# UNA ALTERNATIVA DIDACTICA EN LA ENSEÑANZA DE LA LOGICA PARA ESTUDIANTES DE PRIMERO DE BACHILLERATO EN COMPUTACION

# UNIVERSIDAD PEDAGÓGICA NACIONAL FRANCISCO MORAZÁN

VICERRECTORÍA ACADÉMICA
DIRECCIÓN DE POSTGRADO

UNA ALTERNATIVA DIDACTICA EN LA ENSEÑANZA DE LA LOGICA PARA ESTUDIANTES DE PRIMERO DE BACHILLERATO EN COMPUTACION

**Tesis**
**Para obtener el título de Master en Educación Matemática**

Presentada por:

EDGAR VASQUEZ ALBERTO

ASESOR

DR. MAURO GARCIA PUPO

SAN PEDRO SULA, FEBRERO DE 2007

# *DEDICATORIA*

Este trabajo se lo dedico a toda mi familia.

**A mi Papá y mi Mamá** porque siempre han estado pendientes de mi en todo momento y han sido un apoyo importante para mi.

**A Evelin mi esposa**, porque siempre ha sabido comprenderme y siempre me ha apoyado en todo momento.

**A mis hijos Edgar y Evelin**, porque están entre lo que mas amo en esta vida y de una u otra forma hubieron momentos que tomé para el desarrollo de este trabajo siendo un tiempo que ellos necesitaron.

# *AGRADECIMIENTO*

A las autoridades de la Universidad Pedagógica Nacional “Francisco Morazán” Centro Universitario Regional, por el apoyo incondicional para culminar mis estudios de Postgrado.

Al Dr. Mauro García Pupo por todo el tiempo que ha dedicado para mi, así como la motivación dada en los momentos de desanimo.

Al Dr. Adalid Gutiérrez por los valiosos aportes que me brindaron en las clases de la especialidad, así como el apoyo moral en los momentos difíciles de este proceso.

Al MSC. Hermes Díaz por brindarme el apoyo en la revisión final de este trabajo y a lo largo de mi carrera profesional.

A todos mis amigos, siempre son las personas que me apoyan en forma incondicional y que sin su ayuda no hubiese sido posible este trabajo.

**EDGAR VAS QUEZ ALBERTO**
**TESISTA**

…Una buena notación tiene una tal sutileza y ejerce una sugestión
tal que a veces parece un maestro vivo

BERTRAND RUSELL

*"Hay partes enteras de la matemática*
*sin apenas simbolismo,*
*y páginas enteras de símbolos*
*sin apenas matemáticas"*
E.T. Bell

**INDICE GENERAL**

# *1. INTRODUCCION*

En contra de lo que pudiera parecer, la ciencia de la computación y las teorías sobre la computabilidad no pertenecen a la disciplina que hoy conocemos como “informática”, sino, a las Ciencias Matemáticas que es muy anterior a ella y en la mayoría de los novedosos descubrimientos y aplicaciones en su fundamento. La programación es un problema de aplicación de las matemáticas, tal como lo resume Ojeda (2000, Pág. 5) “la lógica es a la computación como el calculo infinitesimal es a la física”. Al observar los cuadros de calificaciones se tienen estudiantes que aprueban la asignatura de matemáticas pero no la de programación, esto permite preguntarse: ¿porque si un estudiante aprueba las clases de matemáticas en los cursos de secundaria, como es posible que puedan reprobar la clase de programación? Pareciera como si es una clase común y cualquiera.

La teoría de la computación puede separarse en dos tradiciones: por un lado la tradición teórica de la computación, arraigada en los estudios de Allan Türing que basa sus principios fundamentalmente en los ceros y unos, y la moderna computación científica en la que muchos algoritmos son algoritmos de números reales (Blue, 2004, Pág. 1024). Estos algoritmos se originan con Newton, Euler, Gauss, entre otros. En los cursos de computación estas dos tradiciones son tratadas como cursos independientes. Sin embargo, ambas tradiciones comparten la noción de algoritmo. El método de newton es el ejemplo mas citado en los textos de análisis numérico. La máquina de Türing es el modelo de computación que mas se da en los textos de ciencias de la computación sobre algoritmos.

La teoría de Türing surgió para esclarecer el concepto de algoritmo computacional y clasificar los problemas en decidibles o no decidibles. Su ulterior desarrollo ha fundamentado la Teoría de la Computabilidad y las Máquinas de Türing se han convertido en el paradigma para comparar diferentes modelos que sustentan las bases de esta teoría y permiten incursionar en el mundo de las técnicas de diseño de algoritmos y explicar científicamente los mismos. Lejos estaba Türing de imaginarse todo esto. Además los símbolos que manipula una máquina de Türing no necesariamente tienen que ser ceros y unos, sino que pertenezcan a un alfabeto que

permitan formar palabras libre de ambigüedades, mas notoriamente pueden representarse situaciones en las que los enunciados que forman el algoritmo solución únicamente se preste para dos interpretaciones mutuamente excluyentes.

En el caso de la programación para estudiantes de secundaria es una temática que puede situarse en el medio de las dos tradiciones. Los problemas que los estudiantes resuelven tienen mucho que ver con los algoritmos que generalmente son definidos sobre números reales, aunque en el análisis numérico y en la computación científica los algoritmos son definidos sobre los números reales o los números complejos. En otro sentido se debe distinguir que cuando se habla de la maquina de Türing, se focaliza principalmente en el funcionamiento de la computadora desde la perspectiva del hardware en el sentido inicial que proveerá de una clara comprensión del papel que juega el lenguaje de programación y su papel de interprete con el equipo físico de la computadora. Mientras que cuando se habla de la computación científica se refiere más a las aplicaciones que requieren la ayuda de una computadora. Se necesita conocer de ambas tradiciones para poder entender el porque de la programación de computadoras, la lógica de su construcción y de las aplicaciones que son posibles en cada una de ellas.

Cabe hacer notar que las “divergencias”, si es que conviene llamarlas así, entre las dos tradiciones lejos de traer conflictos generan una serie de resultados en beneficio de la ciencia en general. A través de la historia se ha notado que mas bien se trata de “reconciliar la disonancia entre las dos tradiciones, talvez unificar, pero lo mas importante, ver como perspectivas y herramientas de cada una pueden informar la otra” (Blue, 2004, Pág. 1025).

Esta reflexión lleva a reseñar uno de los problemas fundamentales que afrontó la teoría clásica de la computación, la axiomatización de toda la matemática propuesta por David Hilbert. Hilbert el generalista más grande del siglo XX intento demostrar que todas las estructuras matemáticas podrían escribirse en función de ciertos axiomas iniciales. Sin embargo, encontró enunciados en los cuales no pudo definir su valor de verdad. Para su sistema axiomático necesitaba enunciados que fueran verdaderos o falsos. En este punto, David Hilbert se encontraba con el problema mas grave, al tratar de demostrar la

no contradicción. Hilbert intentaba axiomatizar a toda la matemática. Quizás el problema que revolucionó las ciencias matemáticas y que por una parte obligó a los matemáticos a resolver un problema lógico, y por otra parte, esclareció el alcance de los algoritmos computacionales en cuanto a su solubilidad. Claramente una situación con carácter muy matemática.

Pero en 1931, Kurt Gödel equilibra las cargas con su articulo “Sobre proposiciones formalmente no decidibles en Principia Matemáticas y sistemas relacionados” mas conocido como el Teorema de la Incompletitud de Gödel, en este articulo esencialmente demuestra que ningún sistema axiomático es completo, pues siempre existirán enunciados del sistema que no podrán ser demostrados con los axiomas del sistema propio, por lo que será necesario recurrir a otro sistema exterior para poder resolver el problema. Son muchos los significados que toman los resultados de Gödel, Careaga interpreta por una parte el teorema de Gödel en el sentido de que ningún sistema basado en un número finito de axiomas está completo, ya que siempre existirán proposiciones cuya verdad o falsedad será imposible de decidir (Careaga, 2002).

En el campo computacional esto tiene muchas implicaciones. Primeramente, los problemas que tienen una solución computacional, pertenecen a la clase de problemas decidibles. Problemas decidibles son aquellos que pueden solucionarse mediante un algoritmo, “se dice que un algoritmo resuelve un problema si el algoritmo puede aplicarse a cualquier instancia del problema y se pueda garantizar que siempre produce una solución” (Solís, 2000, p.34) por lo que la dificultad radica en que hay problemas para los que no se puede escribir o no existe un algoritmo que para cualquier instancia produzca una solución. La aclaración de Solís es válida, porque existen problemas para los que se puede escribir un algoritmo que lo resuelve para determinadas instancias, pero no para la totalidad de las instancias de sustitución.

Después de este primer escollo, otra dificultad que se puede presentar, es la limitante en los recursos de hardware que se dispone o también las condiciones de software en las que se ejecutara el algoritmo. Si un algoritmo no es eficiente tardaría mucho tiempo en resolver el problema, lo que volvería inútil el equipo de cómputo. En este caso hay

que valorar la cantidad de memoria RAM de la PC y la velocidad del procesador de la computadora.

Un aspecto muy importante en un algoritmo es la eficiencia. Se busca por lo tanto, un algoritmo que resuelva un problema ocupando el mínimo de recursos posible y en el menor tiempo posible. En general, el algoritmo más eficiente debe ser el más rápido.

En conclusión, para el caso computacional, podemos utilizar la computadora para ayudar a resolver un problema a través de la programación siempre y cuando ese problema puede ser resuelto mediante un algoritmo.

Después de la revisión panorámica del estado de la cuestión se ve que el aprendizaje de la programación puede involucrar muchos aspectos adyacentes al conocimiento del código del lenguaje de programación. Estos aspectos pueden ser inherentes en el estudiante, el profesor o en ambos. Entre otros aspectos pueden mencionarse: tener conocimientos generales tales como el funcionamiento de la computadora moderna en cuanto al procesamiento de la información, relación entre el software y el hardware, algoritmos usados por los procesadores, unidades de almacenamiento, funcionamiento de periféricos; sistemas de numeración principalmente base dos, base 10, octal y hexadecimal.

Agregando a los elementos antes mencionados las dificultades de aprendizaje de la programación en alumnos, son muchas las interrogantes que pueden surgir tratando de explicar porque los estudiantes no pueden aprender a programar. Así, para el caso, ¿el alumno que tiene dificultades en programación desconoce el funcionamiento de la computadora?, está interrogante puede surgir cuando no logra restringir el alcance de un algoritmo a las limitantes de hardware de la computadora, como en el caso de una aproximación para la raíz irracional de una ecuación. En este punto se descarta el conocimiento o desconocimiento acerca de las ecuaciones. Un obstáculo que puede sumarse con esta interrogante es la no comprensión del concepto de variable asociado con el de valor presente en la memoria de la computadora, el concepto de valor presente en muchos estudiantes puede resultar un tanto sorprendente pues no se concibe la idea de que el valor de una variable pueda cambiar de un momento a otro en

el desarrollo de un algoritmo, contrario a lo que sucede cuando se resuelve una ecuación donde las variables tienen asociados valores fijos cuando no se utilizan métodos interactivos. El currículo de Matemáticas para el ciclo común de cultura general no contempla resolver ecuaciones con métodos numéricos por lo que este aspecto podría ayudar en la comprensión del concepto de valor presente.

Otra interrogante que puede surgir después de que el alumno ha comenzado a escribir programas que ayudan a resolver problemas con una dificultad elemental es: ¿el alumno comprende para que sirve un programa? Y todavía más, ¿el alumno sabe cuál es el objeto de la creación de las computadoras? No se sabe si aclarando estas incógnitas los estudiantes pueden mejorar su comprensión en el desarrollo de programas computacionales. Así mismo puede agregarse otra interrogante: ¿sabe el estudiante como una computadora ejecuta un programa?, aunque en general se trata de nociones básicas y generales, su desconocimiento puede suponer conflictos al momento de escribir programas o ventajas en caso contrario. Son muchas las interrogantes que pueden surgir tratando de encontrar una explicación para las dificultades que presentan los estudiantes en el proceso de aprendizaje de la programación de computadoras, por lo que el presente trabajo y ningún otro podrían responder todas esas inquietudes, por lo que tendrá que focalizar en un aspecto en especial.

# *2. PLANTEAMIENTO DEL PROBLEMA*

Los bachilleratos en computación son relativamente nuevos en la reciente oferta educativa, y aparecen en el segundo lustro de la década de los años noventa. Sin embargo, el movimiento latinoamericano por la privatización de la educación ha ocasionado el surgimiento de un sin número de instituciones privadas en el que utilizan, como bandera de publicidad, las carreras computacionales. Esto ha ocasionado una alta demanda de profesores en el área de computación. Por otro lado, la oferta de personal calificado es casi nula. En los últimos años en la Universidad Pedagógica Nacional Francisco Morazán, encargada de preparar los cuadros docentes para el nivel medio del sistema educativo nacional, no ha existido la carrera de informática. Solamente las universidades privadas son las únicas que han graduado personal en el área

informática, pero el hecho de dominar un área del conocimiento no significa que se pueda desarrollar una cátedra de forma acertada. Sin embargo, las clases de programación en todos los niveles son impartidas por catedráticos que simplemente son egresados de bachilleratos en computación del nivel medio (anexo11). Una de las razones por las que se emplean ex alumnos para impartir las clases de programación es la parte salarial, que es un pago inferior al que perciben los docentes graduados en una universidad. En los centros educativos no es común encontrar egresados de las universidades privadas, estos normalmente trabajan en oficinas y empresas con una mejor remuneración salarial. En el momento de contratación laboral en el campo educativo, los egresados de universidades sin un perfil pedagógico en el área computacional deberían ser la última opción por considerar, tomando en cuenta que son las personas que a pesar de contar con una mejor preparación científica en el área computación, no así en el área de la pedagogía y la didáctica de la enseñanza de las ciencias computacionales. Estas condiciones iniciales dificultan un óptimo desarrollo de las clases de informática en las carreras con orientación computacional.

El campo computacional en estos momentos es ya considerado de alcances inconmensurables, cada una de sus áreas contiene una cantidad importante de líneas de expansión entre otras las redes de computadoras, tecnologías inalámbricas, informática en general, el diseño asistido por computadora (CAD), entre otros. Pero también el área de la programación y los lenguajes de programación que representa una de las áreas de mayor interés y crecimiento.

Focalizando en los bachilleratos del nivel medio con orientación en computación se tiene que contemplan en su pensum académico la asignatura de programación, es una de las clases medulares del plan de estudios y de una importancia vital para el cumplimiento del perfil del egresado en el área informática.

Por otra parte, reformas futuras en los planes de estudios del ministerio de educación Pública contemplan la inclusión de los bachilleratos Técnico Profesionales en los que se incluye las tecnologías de la información como uno de los campos ocupacionales donde se insertaran laboralmente. En este campo se proponen bachilleratos técnicos profesionales en informática y bachillerato técnico en profesional en diseño multimedia.

En todos los casos “se visualiza como el ámbito donde los egresados deben poseerán las competencias requeridas para asistir al usuario de productos o servicios informáticos brindándoles servicios instalación, capacitación, sistematización, mantenimiento primario, resolución de problemas derivados de la operatoria” (Alas Solís, Hernández Rodríguez, Moncada Godoy, 2004, p.75). Esta situación pone de manifiesto la vigencia y necesidad de mejorar los métodos de enseñanza aprendizaje de la programación que cada vez necesitará mayor calidad y eficiencia en los futuros egresados de las carreras computacionales.

Pensando en el futuro de la educación en el campo de la informática, preocupa el hecho que en todas las instituciones con este tipo de bachillerato se observa un alto índice de reprobación (40% de reprobados) en la asignatura de programación, especialmente en el primero y segundo año de carrera. En ese sentido, el Instituto Oficial Primero de Mayo no escapa a esta problemática que afecta a gran número de estudiantes que ven a esta asignatura como el obstáculo para lograr las más altas aspiraciones académicas del nivel medio.

Esta dificultad es señalada por algunos investigadores en el campo de la lógica computacional, donde identifican que uno de los principales problemas a los que se enfrentan las personas que desean aprender a programar es crear secuencias lógicas de instrucciones que permitan solucionar un problema dado (Gutiérrez y Vargas, 2000), este mismo autor destaca el hecho que la bibliografía empleada por los profesores en la enseñanza de programación focalizan en la semántica y la sintaxis de los lenguajes de programación ignorando la lógica de la programación que se adquiere mas por el proceso de maduración de los alumnos que por los métodos lógicos empleados en el proceso enseñanza aprendizaje. En ese sentido se pueden citar muchos libros que se utilizan como textos de programación para principiantes con ese enfoque que lejos de facilitar el aprendizaje de la lógica de la programación más bien frustra los anhelos y deseos de muchos estudiantes que podrían tener un óptimo desarrollo en el campo de la programación. Al respecto Tucker y Noonan (2003) amplían el alcance del estudio de la programación computacional e implícitamente la lógica computacional al declarar que

> *El estudio de los lenguajes de programación no debería reducirse al estudio de los lenguajes del presente; siempre podemos aprender características nuevas, pero esto por si mismo nos proporciona poca inspiración o experiencia sobre las posibilidades futuras del diseño de lenguajes*. (Pág. 4)

La mayoría de los libros de lógica en la actualidad están dirigidos, principalmente, al desarrollo del pensamiento matemático. Esta condición ubica en desventaja a los estudiantes de informática, pues además de la lógica matemática necesitan la lógica computacional que es uno de los pilares en el desarrollo de la programación y del estudio de las estructuras de datos que se erige como el paso inmediato en el estudio de la programación.

Sin embargo, los contenidos de la lógica computacional se encuentran plasmados en los libros de matemáticas discreta, y que, en ningún momento han sido diseñados pensando en el desarrollo de la lógica y de la lógica computacional de los estudiantes de informática.

Por otro lado los alumnos del Bachillerato en computación consignan en su plan de estudios dos cursos de matemáticas: uno en primero de computación y el otro en segundo de computación. En ninguno de los casos se considera la enseñanza de la lógica, mucho menos el caso particular los métodos lógicos que conlleven al aprendizaje del diseño algorítmico en la resolución de problemas computacionales.

Esto permite plantear el siguiente **problema de investigación**:

¿Cómo enseñar lógica a los alumnos de primero de bachillerato de computación para facilitar el diseño de algoritmos que permitan resolver problemas computacionales?

## 2.1 Objetivo de la investigación.

Una alternativa didáctica para la enseñanza de la lógica a los alumnos de primero de bachillerato en computación del Instituto Oficial Primero de Mayo que conlleve al desarrollo del pensamiento algorítmico. Es por ello que se puede identificar el siguiente:

## 2.2 Objeto de investigación.

El proceso enseñanza-aprendizaje de la lógica matemática; que al particularizarlo y en consecuencia con todo lo que se ha explicitado permite establecer como:

## 2.3 Campo de acción

El proceso de enseñanza-aprendizaje para la asignatura de programación en aquellas carreras con un perfil en la informática o en ciencias de la computación, y en particular el bachillerato en computación del Instituto Oficial Primero de Mayo.

Todo lo expuesto hasta ahora permite formular las siguientes:

**Preguntas de investigación**:

1. ¿Cuáles son los métodos lógicos que se necesitan para el desarrollo de un programa?

2. ¿Qué procedimientos utilizan los alumnos que al estructurar un programa fallan y cuales los que le permiten estructurar correctamente un programa?

3. ¿Como enseñar lógica a los alumnos del bachillerato en computación para favorecer el desarrollo del pensamiento algorítmico?

## 2.4 TAREAS CIENTIFICAS

**1.** Determinar las condiciones bajo las que se desarrolla un programa de computadora para entender cuáles son los conocimientos básicos necesarios para su construcción.

**2.** Diagnosticar el nivel académico de los profesores que imparten la clase de programación a los estudiantes de Primero de Bachillerato en Computación para visualizar algunas condiciones bajo las que se desarrollan las clases de programación.

**3.** Estudiar los algoritmos computacionales en cuanto a su desarrollo, evolución histórica y complejidad, para entender las posibilidades y limitaciones de ejecución de la computadora.

**4.** Estudiar el desarrollo de la computadora y los lenguajes de programación para organizar de una mejor manera los paradigmas de la programación de computadoras.

**5.** Estudiar los métodos lógicos utilizados en el diseño y desarrollo de los programas de computadora.

**6.** Estudiar la relación existente entre la lógica, lógica computacional, lógica matemática, resolución de problemas y programación.

**7.** Analizar los estilos de aprendizaje y la forma que pueden contribuir en la enseñanza aprendizaje de la programación.

**8.** Precisar las diferentes teorías psicológicas existentes respecto a la enseñanza aprendizaje de la programación.

**9.** Elaborar una alternativa didáctica para la enseñanza de la lógica a los alumnos de primero de bachillerato en computación.

**10.** Aplicar el método de expertos para evaluar la valoración de la solución que se da al problema de investigación

# *3. MÉTODOS UTILIZADOS*

Para el desarrollo de la investigación se emplearon los **métodos** siguientes:

## 3.1 Del nivel teórico

- Análisis-síntesis e inducción-deducción para el estudio de las fuentes de información, obtener de ellas regularidades y tendencias relacionadas con la enseñanza de la programación, así como los métodos lógicos empleados en el desarrollo de programas.
- Método de análisis histórico y lógico para determinar la evolución en el diseño de programas y el aprendizaje de la programación.
- Método de enfoque sistémico para estudiar el proceso enseñanza aprendizaje de la programación.
- Método de modelación para elaborar el modelo teórico en que se sustenta la solución del problema de investigación.

## 3.2 Del nivel empírico

- La encuesta para:
    - determinar las características de los estilos de aprendizaje del grupo de los alumnos en estudio y los métodos lógicos que utilizan para diseñar un programa que resuelve un problema.
    - Determinar el nivel académico de los profesores que imparten la clase de programación en el primero de bachillerato en computación en los institutos seleccionados para el estudio.

- Conocer la opinión de los profesores acerca de la enseñanza de la programación desde la perspectiva de la resolución de problemas.

- El método de expertos para fundamentar la validez de la propuesta.

## 3.3 Del nivel estadístico.

Técnicas de la Estadística Descriptiva para realizar el procesamiento de la información recolectada con los instrumentos asociados con la investigación de las características de los estilos de aprendizaje y métodos lógicos empleados por los alumnos en el desarrollo de programas, así como el nivel académico de los profesores que imparten las clases de programación a los estudiantes de primero de bachillerato en computación.

## 3.4 Del nivel cualitativo.

- El muestreo de casos extremos, para focalizar sobre aquellos casos que son ricos en información a causa de que son inusuales o especiales en alguna forma. La lógica de este tipo de muestreo está en que del análisis de las condiciones inusuales se pueden derivar aprendizajes útiles para comprender aspectos ocultos en las situaciones regulares.

- El muestreo de variación máxima tiene como propósito capturar y describir los temas centrales o las principales características que tipifican una realidad humana relativamente estable. Ésta última es la principal diferencia con el anterior.

- El muestreo de casos homogéneos busca describir algún subgrupo en profundidad. Es la estrategia empleada para la conformación de grupos focales. El punto de referencia más común para elegir los participantes de un grupo focal es que estos posean algún tipo de experiencia común en relación con el núcleo temático al que apunta la investigación.

- El muestreo de caso típico pretende mostrar a quién no está familiarizado con la realidad objeto de análisis los rasgos más comunes de dicha realidad. La definición de “típico” cualitativamente se construye a partir del consenso de opiniones entre informantes clave, buenos conocedores de la realidad bajo estudio

# CAPÍTULO I

# *4. MARCO TEORICO*

En este capitulo se presentan las teorías que sirven de fundamento en la propuesta didáctica para el mejoramiento en la enseñanza de la lógica de la programación. Para este fin se sigue la estructura siguiente: Considerando que la computadora es el entorno bajo el cuál se desarrolla el proceso educativo de la solución del problema científico se describen primeramente algunos apuntes relacionados el funcionamiento de la computadora y la historia de la programación de computadoras; seguidamente se introduce el concepto de algoritmo computacional que sirve para explicitar los principios de la lógica matemática que fundamentan la lógica computacional y en consecuencia la lógica de la programación, esto incluye una descripción de las leyes de la lógica matemática que se utilizan en la elaboración y desarrollo de programas de computadora; para afianzar el aprendizaje de la lógica de la programación se mencionan algunas teorías psicológicas relacionadas con las ciencias computacionales y los estilos de aprendizaje que los alumnos utilizan en el proceso de aprendizaje; Así, por último, se mencionan los principales conceptos involucrados en la resolución de problemas con una breve descripción de la evolución histórica de esta teoría, por lo que se propone enseñar la programación de computadoras desde el paradigma de la resolución de problemas con algunos ajustes especiales que se describen en el siguiente capitulo destinado a la descripción de la propuesta didáctica para la enseñanza de la lógica de la programación.

## 4.1 APUNTES SOBRE EL FUNCIONAMIENTO DE LA COMPUTADORA Y LA HISTORIA DE LA PROGRAMACIÓN

Un aspecto muy importante en la programación de las computadoras, es tener algunos conocimientos mínimos sobre el funcionamiento de la computadora a nivel de arquitectura y a nivel lógico, porque los programas de computadora se ejecutan bajo esas condiciones. Se sugiere primeramente, que los profesores que impartan las clases de programación para novatos en el primer año de computación tengan los conocimientos básicos necesarios en los campos anteriormente mencionados. Además, los estudiantes deben tener conocimientos básicos sobre el funcionamiento de la

computadora en general. Se aclara que para desarrollar un algoritmo no es necesario tener conocimientos computacionales, pero no es el caso de un algoritmo computacional.

Entre otros aspectos debe tenerse muy en cuenta que la historia y origen de la computadoras se remonta a los trabajos realizados por Lady Ada Lovelace considerada como la primera programadora de la historia por sus contribuciones en el mejoramiento del telar de Jacquard y a la notable contribución de Herman Hollerith en la construcción de las maquinas internacionales de negocios, con las que logró procesar los censos de Estados Unidos de América a partir de 1890, y en las que se comenzó la automatización del procesamiento de la información, es de suma importancia destacar que uno de los aportes teóricos claves para el surgimiento de la programación fueron los trabajos realizados por Alan Türing, quien usó un dispositivo para definir matemáticamente lo que es un algoritmo y lo denominó Máquina de computación lógica que en su honor se conoce como máquina de Türing. Existen diversas variedades de una maquina de Türing, pero la más simple se puede describir como un dispositivo que cumple las siguientes condiciones (Hodges, 2002):

1. Tiene una cinta sobre la que puede desplazarse de izquierda a derecha un cabezal de lectura/escritura. La cinta contiene una serie de celdas, y en cada una de ellas puede escribirse un símbolo de un conjunto finito.

2. El cabezal puede moverse a derecha a izquierda de su posición actual, así como leer el contenido de una celda o escribir en ella cualquier carácter de su alfabeto.

3. Existe un registro de su estado que almacena el estado de la máquina. El número de estados posibles es finito.

4. Existe una tabla de acción, que contiene las instrucciones de lo que hará el autómata. Estas instrucciones representan en cierta forma el “**programa**” de la máquina. La ejecución de cada instrucción de la tabla de acción incluye cuatro pasos:

a. Leer un carácter en la posición actual.
b. Escribir un nuevo símbolo en esta posición (puede ser el mismo que había)
c. Desplazar el cabezal una celda a derecha o izquierda, en algunos modelos de desplazamiento puede ser nulo.
d. Decidir cuál será el nuevo estado en función del carácter que se acaba de leer y del estado actual. Si la tabla de acción no contiene ninguna correspondencia con el estado actual y el símbolo leído, entonces la máquina detiene su funcionamiento.

De allí se infirió la idea de programar (Gardner, 1996) utilizando un código binario. Estas ideas fueron aprovechadas por John Von Newman para crear un programa que permitiera instruir la maquina de Türing en 1950. Gardner afirma que por primera vez, se vislumbro la posibilidad de que una computadora preparase y ejecutase sus propios programas.

Desde que Newman comenzó a instruir las maquinas de Türing hasta nuestros días, la tarea de escribir programas ha ido creciendo y mejorándose cada vez mas, en armonía con los avances tecnológicos a nivel computacional y con las necesidades de uso de la computadora en la resolución de problemas. Una de las tareas en el actual mundo de la computación es el arte de escribir programas, considerado de esta forma desde sus momentos iniciales, en ese sentido, explica Knuth: “Tenemos que ver que la programación de computadoras es un arte, porque aplica el conocimiento acumulado al mundo, porque requiere habilidad e ingenio, y especialmente porque produce objetos de belleza” (Knuth, 1974, p.673). Este adjetivo ha sido dado a la programación de computadoras desde sus inicios, y actualmente se reconoce la belleza en el diseño tanto visual como técnico al momento de escribir programas, además del ingenio en la optimización de los algoritmos codificados en cualquiera de los lenguajes de programación.

El arte de escribir programas convertido casi en una ciencia trae como consecuencia el aparecimiento de una diversidad de lenguajes de programación de acuerdo a las

necesidades de la situación problemática. A tal punto que en los principales rotativos del mundo se anuncia con mucha preocupación la proliferación de los lenguajes de programación, entre los que se pueden mencionar: De los mas antiguos Fortran y Cobol, el mas popular el lenguaje C, el sucesor C++, también el lenguaje Pascal; La lista continúa con lenguajes como Perl, Python, PHP, TLC, PROLOG, Ada, JAVA; entre los de mas reciente creación el lenguaje Ruby (Gomes, 2005).

Es muy probable que la versatilidad de los lenguajes de programación hace que muchos de los profesores que titulares de las clases de programación se atrevan a impartir la cátedra sin tener los conocimientos necesarios para el desarrollo óptimo de la cátedra, así como las herramientas pedagógicas y didácticas que permitan la adecuada realización del proceso enseñanza aprendizaje de la programación.

Por una parte se pueden abordar los lenguajes de programación y por otra parte los métodos de enseñanza aprendizaje utilizados por los profesores en el proceso educativo. Según el anexo (métodos usados en la enseñanza de la lógica de la programación), puede verse que los profesores utilizan el mismo método de enseñanza independientemente del lenguaje de programación utilizado para codificar los programas. En opinión de Salvat (1999) los modelos cognitivos de aprendizaje se recomiendan más que los conductuales en la enseñanza de la informática y de las tecnologías computacionales, sin embargo los modelos conductuales pueden servir en situaciones dentro de la informática tales como la iniciación en el manejo del teclado. Pero considerando las tendencias actuales en programación se ve que los paradigmas orientados a las resolución de problemas se adaptan mejor a los paradigmas computacionales, en especial a la programación lógica, que una de sus aplicaciones pertenecen al campo de la Inteligencia Artificial.

Debido a que un programa puede ser codificado en una diversidad de lenguajes de programación y que de alguna forma todos los lenguajes de programación son equivalentes, en la actualidad existe también una diversidad en los paradigmas de la programación.

A pesar de que los lenguajes programación están diseñados para comunicar ideas sobre algoritmos entre personas y las computadoras, tienen limitadas características expresivas. Sin embargo dichos lenguajes permiten un sorprendente amplio margen de expresión algorítmica, que soporta una gran variedad de aplicaciones de computación a través de varios dominios de aplicación. Para conseguir semejante versatilidad, las distintas comunidades de programadores de estos dominios han desarrollado caminos especiales y diferentes, o paradigmas para expresar algoritmos que se ajustan especialmente bien a sus propias áreas de aplicación. Por tanto en el área computacional, un paradigma es una forma de representar y manipular el conocimiento. Representan un enfoque particular o filosofía para la construcción del software.

Los paradigmas en programación aparecen en función de los dominios de aplicación como puede ser la computación científica, los sistemas de gestión de la información, la inteligencia artificial, los sistemas, y las aplicaciones centradas en la Web. (Tucker y Noonan, 2003).

Los principales paradigmas de aplicación que han surgido son los siguientes: La programación imperativa, programación orientada a objetos, programación funcional, programación lógica (declarativa), programación guiada por eventos, programación concurrente, etc.

Se da una pequeña descripción de cada uno de los paradigmas de la programación de computadoras con la idea de dar un panorama general acerca de los factores que involucran el desarrollo de la programación a través de su historia y las tendencias actuales.

En el paradigma funcional los programas forman una colección de funciones matemáticas, cada una de ellas con su entrada (dominio) y su resultado (intervalo). Las funciones interactúan y se combinan unas con otras utilizando condicionales, recursividad y composición funcional. Otras características propias de estos lenguajes son la no existencia de asignaciones de variables y la falta de construcciones estructuradas como la secuencia o la iteración (lo que obliga en la práctica a que todas las repeticiones de instrucciones se lleven a cabo por medio de funciones

recursivas).Entre los lenguajes de programación funcional se mencionan: LISP, SCHEME, OCAML, HASKELL, MIRANDA, ML, etc.

En la programación lógica, los programas son una colección de declaraciones lógicas sobre el resultado que debería conseguir una función, en lugar de cómo debería conseguirse dicho resultado. La ejecución del programa aplica estas declaraciones para obtener una serie de soluciones posibles a un problema. La programación lógica ofrece una alternativa para los problemas con múltiples soluciones, entre los que más se aprovechan esta la prueba de Teoremas. El lenguaje más importante en la programación lógica es PROLOG.

En la programación guiada por eventos, los programas se conforman por un bucle continuo que responde a los eventos generados en un orden no predecible, son generados a partir de acciones del usuario en la pantalla otras fuentes. Entre los principales lenguajes para este paradigma se tienen: VISUAL BASIC y JAVA.

La programación concurrente es una colección de procesos cooperativos que comparten información unos con otros de vez en cuando, pero que operan generalmente de forma asíncrona. El paralelismo puede producirse dentro de un solo proceso. Entre los principales lenguajes de programación en este paradigma se tienen: SR, LINDA Y FORTRAN de alto rendimiento.

En el Paradigma Imperativo el programa es una serie de pasos, cada uno de los cuales realizan un cálculo, recupera una entrada o tiene como resultado una salida. Los programas contienen también procedimientos, sentencias condicionales, asignaciones, bucles y secuencias. Considerando que es el paradigma mas utilizado y el mas desarrollado, la solución al problema de investigación estará dentro de este paradigma de la programación, específicamente utilizando un lenguaje de alto nivel, pues prácticamente todo el hardware de los computadores está diseñado para ejecutar código de máquina, que es nativo al computador, escrito en una forma imperativa. La arquitectura de estos equipos, llamados equipos Von Newman, tiene una memoria que contiene tanto las instrucciones del programa como los valores de los datos. Los lenguajes imperativos tienen declaraciones de variables, expresiones y comandos. Las

declaraciones asignan nombres a ubicaciones de memoria y asocian tipos con valores almacenados. Las expresiones se interpretan en términos de valores actuales las instrucciones del programa; dado un nombre x, las instrucciones del programa devuelven el valor actual en los valores de los datos que esta asociado con x. Los lenguajes de programación imperativos de alto nivel, adicionalmente, permiten la evaluación de expresiones complejas, que pueden consistir de operaciones aritméticas y evaluaciones de funciones y la asignación del valor resultante a la memoria. Los primeros lenguajes imperativos fueron los lenguajes de máquina de los computadores originales. En estos lenguajes, las instrucciones fueron muy simples, lo cuál hizo la implementación de hardware[1] fácil, pero obstruyendo la creación de programas complejos.

Un lenguaje de programación es un lenguaje de recorrido completo que además soporta un cierto número de características fundamentales que han nacido con la evolución del paradigma de la programación imperativa desde los años cuarenta, estas características incluyen:

1. Tipos de datos para números reales, caracteres, cadenas, booleanos y sus operadores.

2. Estructuras de control, bucles for y while, instrucciones case.

3. Asignación de elementos y arrays.

4. Asignación de elementos y estructuras de grabación.

5. Comandos de entrada y salida.

6. Punteros.

7. Procedimientos y funciones.

[1] Hardware. Con este término se reconoce a la parte física de una computadora.

A este tipo de paradigma de programación se le suele llamar algorítmico, dado que el significado de algoritmo es análogo al de receta, método, técnica, procedimiento o rutina, y se define como "un conjunto finito de reglas diseñadas para crear una secuencia de operaciones para resolver un tipo específico de problemas". De esta forma para N. Wirth, un programa viene definido por la ecuación:

Algoritmos + Estructura de Datos = Programas

No obstante, debe considerarse que el concepto de algoritmo encaja en otros tipos de paradigmas, es privativo del tipo de programación procedimental en el que su característica fundamental es la secuencia computacional. Dijkstra destaca que uno de los principales avances en la programación imperativa se dio cuando se suprimió la instrucción GO TO de los lenguajes de programación de alto nivel (Dijkstra, 1968).

Entre algunos de los lenguajes de programación imperativa se tienen: Fortran iniciado en 1954 por John Backus en IBM, Basic, C, C++, Cobol, Pascal, Perl, entre otros.

El Paradigma de la programación orientada a objetos define los programas en términos de clases de objetos, objetos que son entidades que combinan estado (datos) y comportamiento (métodos). Surge en los años ochenta y noventa con la intensión de remediar algunos vacíos dejados por el paradigma imperativo como ser la extensión de los códigos de los programas que al momento de realizar modificaciones o correcciones representan un dificultad enorme, por otra parte se amplia la abstracción de los datos de una forma sencilla agregando operaciones nuevas, con la idea de clase, el principal bloque de construcción de los lenguajes orientados a objeto. La programación orientada a objetos contempla las siguientes características:

Abstracción

Cada objeto en el sistema sirve como modelo de un "agente" abstracto que puede realizar trabajo, informar y cambiar su estado, y "comunicarse" con otros objetos en el sistema sin revelar cómo se implementan estas características. Los procesos, las funciones o los métodos pueden también ser abstraídos y

cuando los están, una variedad de técnicas son requeridas para ampliar una abstracción.

Encapsulamiento

También llamada "ocultación de la información", esto asegura que los objetos no pueden cambiar el estado interno de otros objetos de maneras inesperadas; solamente los propios métodos internos del objeto pueden acceder a su estado. Cada tipo de objeto expone una interfaz a otros objetos que especifica cómo otros objetos pueden interactuar con él. Algunos lenguajes relajan esto, permitiendo un acceso directo a los datos internos del objeto de una manera controlada y limitando el grado de abstracción.

Polimorfismo

Las referencias y las colecciones de objetos pueden contener objetos de diferentes tipos, y la invocación de un comportamiento en una referencia producirá el comportamiento correcto para el tipo real del objeto referenciado. Cuando esto ocurre en "tiempo de ejecución", esta última característica se llama asignación tardía o asignación dinámica. Algunos lenguajes proporcionan medios más estáticos (en "tiempo de compilación") de polimorfismo, tales como las plantillas y la sobrecarga de operadores de C++.

Herencia

Organiza y facilita el polimorfismo y la encapsulación permitiendo a los objetos ser definidos y creados como tipos especializados de objetos preexistentes. Estos pueden compartir (y extender) su comportamiento sin tener que reimplementar su comportamiento. Esto suele hacerse habitualmente agrupando los objetos en clases y las clases en árboles o enrejados que reflejan un comportamiento común.

La programación orientada a objetos introduce nuevos conceptos, que a veces no son más que nombres nuevos aplicados a conceptos antiguos, ya conocidos. Entre ellos destacan los siguientes:

Método

Es un programa asociado a un objeto (o a una clase de objetos), cuya ejecución se desencadena mediante un "mensaje". En analogía con un lenguaje procedural se le llamaría "función".

Mensaje

Una comunicación dirigida a un objeto, que le ordena que ejecute uno de sus métodos con ciertos parámetros.

Propiedad

Atributo o variable: datos asociados a un objeto o a una clase de objetos.

Entre los lenguajes orientados a objetos destacan los siguientes:
Smalltalk, Objective-C, C++, Ada 95, Java, Ocaml, Python, Delphi, Lexico (en español), C Sharp, Eiffel, Ruby, ActionScript, Visual Basic.NET, PHP, PowerBuilder, Clarion.

Como se ve, este paradigma es relativamente nuevo en el ámbito escolar, considerando que cuando una herramienta informática nace, primeramente es conocido en el ámbito de los desarrolladores que no necesariamente son los profesores que imparten las clases de programación.

Estas consideraciones conllevan a tomar en cuenta que antes de escribir un programa y de escoger un lenguaje de programación, debe considerarse el paradigma bajo el cuál se desarrollará el programa. En diferentes paradigmas son diferentes los conocimientos previos necesarios para codificación correcta de los algoritmos computacionales, por tanto la preparación de los alumnos en cada caso debe considerar aspectos especiales de cuidadosa consideración. Consecuentemente es recomendable que los profesores que imparten la cátedra de programación conozcan lenguajes de programación pertenecientes a diferentes paradigmas de programación. Este es sólo uno de los requisitos que debe tener un profesor, pues de acuerdo al National Council for Accreditation of Teacher Education (NCATE) citado en el informe de 2003 por el CSTA (CSTA, 2003), la preparación de los profesores que trabajan en el área computacional deben incluir programación y diseño de algoritmos, organización y operación de

sistemas de computo, representación de datos y organización de la información y aspectos sociales de la computación.

Algunos de los aspectos que considera el NCATE para certificar profesores del nivel secundario se relacionan con conocimientos específicos en ciencias de la computación y capacidades. Entre los conocimientos relacionados con la programación y el pensamiento algorítmico que un profesor debe poseer se detallan:

- Conocimiento y habilidad respecto a la sintaxis y la semántica de un lenguaje de programación de alto nivel.
- Conocimiento y habilidad respecto a mecanismos comunes de abstracción de datos (tipos de datos o clases tales como pilas, árboles, etc.)
- Conocimiento y habilidad respecto a corrección de programas y prácticas (probar resultados de programas, probar diseño de datos, ciclos invariantes, etc.)
- Diseño e implementación de programas de suficiente complejidad demostrativo de conocimiento y habilidades.
- Diseño, implementación y prueba de programas en lenguajes de dos diferentes paradigmas de programación en una manera apropiada a cada paradigma.
- Aplicaciones de varias estructuras de datos y archivos que las provea un lenguaje de programación (objetos, colecciones, archivos, etc.)

De igual manera el profesor debe demostrar las siguientes capacidades:

- Identificar recursos, estrategias, actividades y manipulaciones apropiadas que enseñen la ciencia de la computadora en secundaria
- Planificar lecciones, módulos y cursos relacionados con: Proceso de la programación y conocimientos además de conceptos, y edición de la examinación.
- Desarrollar estrategias apropiadas de evaluación a los objetivos de las lecciones y la necesidad de proveer retroalimentación al estudiante.
- Observar y discutir la enseñanza de la ciencia computacional en secundaria.

- Planear instrucción que involucre la independencia de los estudiantes usando las facilidades computacionales.

En nuestro país el paradigma de la programación orientada a objetos es utilizado principalmente en el sector de los Institutos de enseñanza secundaria Privada (Anexo 1), a diferencia del paradigma imperativo que no se utiliza en muchos casos por falta de conocimiento y que según el NCATE deberían desarrollarse programas en por lo menos dos paradigmas computacionales.

La propuesta en la enseñanza de la programación sugiere comenzar la enseñanza de la programación utilizando el paradigma imperativo. La experiencia en el Instituto Primero de Mayo indica que los alumnos que tuvieron éxito en el desarrollo de programas utilizando el lenguaje de programación Pascal del paradigma imperativo tuvieron una relativa facilidad para comprender el diseño de aplicaciones de sistemas comerciales desarrollados en manejadores de bases de datos como el Visual FoxPro del paradigma de la programación orientada a objetos.

A diferencia de alumnos que llegan de traslado de otros institutos oficiales o privados en los que "conociendo" el paradigma de la programación orientada a objetos tienen dificultades en el diseño y estructuración del código necesario para la implementación de rutinas para objetos determinados. Esto refleja empíricamente la adolescencia de un sólido pensamiento algorítmico, más específicamente el pensamiento algorítmico computacional que debe estar asociado, de igual forma orientado a resolver problemas con el auxilio de la computadora mediante el diseño y desarrollo de programas. Consecuentemente no se debe olvidar que "El lenguaje Pascal fue diseñado por Nicklaus Wirth a finales de los años 60 como un lenguaje docente, y a finales de los setenta, principios de los ochenta muchas facultades y universidades lo habían adoptado para sus planes de estudio" (Tucker y Noonan, 2003, Pág.5).

Aunque el currículo de las carreras computacionales y los textos actuales utilizados en los Institutos de enseñanza secundaria no particularizan en el desarrollo de la lógica de la programación, sino en el conocimiento de un lenguaje de programación, se han

construido muchas herramientas a nivel de software para aprender la deducción lógica (Faraón, Sergio, 2004).

No obstante, la solución propuesta contempla enseñar la lógica de la programación independiente del lenguaje de programación, acercándose más al desarrollo del pensamiento algorítmico. Gutiérrez y Vargas (2000) lo confirman cuando explican que

> *La lógica de programación basada en lenguajes imperativos tuvo su máxima expresión en el paradigma estructurado. Hoy por hoy, la programación Estructurada se considera superada por nuevos paradigmas tales como el Orientado a Objetos. Sin embargo, no debemos perder de vista que la implementación del método de una clase no es mas que código basado en programación estructurada*. (Pág.5)

Los mismos autores confirman que no importa el paradigma de la programación que se utilice, en el fondo siempre se depende de la programación estructurada, de allí que "La lógica de la programación y sus reglas tienen vigencia actual y futura" (Gutiérrez y Vargas, 2000, Pág.5).

Estos rasgos pueden seguirse en los principales ámbitos dedicados al desarrollo de la programación, para el caso, la Computer Science Teachers Association (CSTA, 2003) en el modelo curricular propuesto para las ciencias de la computación en el K-12 propone en términos generales que un estudiante graduado de secundaria debe tener fluidez en la información tecnológica sobre tres ejes ortogonales como lo son: Conceptos, capacidades y habilidades.

Los conceptos que son diez que delinean las computadoras modernas, redes e información: Organización de computadoras, sistemas de información , redes, representación digital de información, organización de información, modelado y abstracción, **pensamiento algorítmico y programación**, universalidad, limitaciones de información tecnológica e impacto social de la información tecnológica.

Las capacidades que son las 10 habilidades fundamentales para el uso de la información tecnológica al resolver un problema: Engranar un razonamiento sustentado, manejar la complejidad, probar una solución, manejar programas y sistemas defectuosos, organizar y navegar estructuras de información y evaluación de la información, colaborar, comunicar audiencias a otros, esperar lo inesperado, anticipar cambios tecnológicos y pensar abstractamente acerca de la información tecnológica.

Las destrezas son 10 habilidades que usa hoy en día las aplicaciones de computadora para uno de sus trabajos: Instalar una computadora personal, usar las características básicas de un sistema operativo, usar un procesador de texto y crear un documento, usar gráficos o paquetes artísticos para crear ilustraciones, diapositivas e imágenes; conectar una computadora a una red, usar Internet para encontrar información y recursos, usar la computadora para comunicarse con otras personas, usar hojas de calculo para modelar procesos simples o tablas financieras, usar manejadores de bases de datos para instalar o acceder a la información, usar materiales instruccionales para aprender acerca de nuevas aplicaciones o características.

## 4.2 EL ALGORITMO Y SU LOGICA

La lógica se ha convertido en uno de los fundamentos matemáticos y en una base formal indispensable en todo informático. La formalización del conocimiento y la automatización de las formas de razonamiento son primordiales en muchas áreas de la informática. La importancia de la lógica en los currículos de Informática va tomando cuerpo propio debido a sus aplicaciones en contextos específicos tales como la programación, la ingeniería del software, el diseño de sistemas de bases de datos y la inteligencia artificial. En los últimos años han ido surgiendo libros de texto de lógica escritos específicamente para estudiantes de Ingeniería Informática, que abordan la lógica desde una perspectiva de aplicación a la computación.

El concepto de Algoritmo en su etimología tiene que ver con *Abu Yafar Mohammed ibn Musa al-Jowarizmi (825)* el autor de *Kitab al jabr walal-muqabala* primer libro que se conoce en la historia de las matemáticas que presenta un equivalente al algebra de nuestros días. Respecto a los nombres, E.T. Bell destaca que, el nombre completo del autor es Mohammed ibn Musa Al-Khowarizmi de Bagdad y Damasco; el tratado que presenta lo llama al-jebr w'almuquabal que significa restauración y reducción.

Haciendo un salto en el tiempo, del año 825 hasta 1545, año en que Cardano publicó su Ars Magna (Bell, 1995), la suma de conocimientos en álgebra de aquella época. Se dice que Cardano fue influido por Raimundo Lulio, 1300, desarrolla su Ars Magna, como un procedimiento general de base combinatoria para hallar todas las verdades. Considerados desde un punto de vista sensato, los procedimientos aducidos por Lulio no tiene mucho valor. Lo importante sin embargo, es que concibió una idea ciertamente espléndida.

El mismo objetivo de reducir a cálculos la determinación de todas las verdades lógicas fue perseguido por Leibniz (1646-1716). Determinó que había dos conceptos que se deberían considerar por separado: El ars inveniendi o procedimiento de generación y el ars iudicandi o procedimiento de decisión. (Nótese que, al menos informalmente, estos dos conceptos se pueden asociar con la pareja recursivamente enumerable y recursivamente decidible). Leibniz vio con claridad la posibilidad de confiar a una máquina la ejecución de un procedimiento general, siendo el primero en construir una máquina de cálculo con las cuatro operaciones básicas. Sin embargo su sueño era tener un calculus ratiocinator para decidir las verdades lógicas. Pensaba que se podría llegar a un procedimiento tal que si una persona opina que A es verdad y otra que no-A es verdad, para ponerse de acuerdo solo necesitarían papel y lápiz y realizar ciertos cálculos que darían la razón a uno de los dos.

## 4.3 DE LA LÓGICA A LA LÓGICA DE LA PROGRAMACIÓN

Se puede situar el comienzo de la aportación de la Ingeniería a la Lógica en 1938, cuando Claude E. Shannon (mas tarde famoso por su Teoría de la Información) observó que las funciones realizadas por circuitos combinatorios, inicialmente construidos con relees, se podrían representar con la notación simbólica del algebra de Boole (Shannon, 1938). A mediados de la década de los 50, D.A. Huffman extendió este trabajo a los circuitos secuenciales, lo cual dio origen al desarrollo de la teoría de maquinas de estados finitos.

La contribución de la lingüística llega a finales de los 50. Noam Chomsky, con su teoría de las gramáticas formales, establece las bases de la lingüística matemática e inicia el camino hacia la formalización en la descripción de los lenguajes naturales (Chomsky, 1959). Al mismo tiempo, se estaba trabajando en la especificación de la sintaxis de lenguajes de programación de ordenadores: Backus adaptó algunos trabajos de E. Post (Post, 1921) a tales especificaciones, y obtuvo una notación que era una variante de las gramáticas libres de contexto de Chomsky. Por otra parte, el estudio de las clases de lenguajes generados por las gramáticas formales y el estudio de las máquinas de estados finitos llevó al establecimiento de una relación inmediata y sorprendente: los mismos fenómenos aparecían de forma independiente en ambos campos, de manera que se podrían establecer isomorfismos entre ambos modelos.

En otro orden de sucesos, Leibniz fue el primero en afirmar la posible existencia de algo equivalente a una lógica formal completa para describir el razonamiento (Parkinson, 1965). No estaba satisfecho con la lógica aristotélica y desarrolló sus propias ideas para mejorarla. Estaba convencido de que podría desarrollar un lenguaje para, y un calculo de, los razonamientos que seria tan importante como el cálculo desarrollado por el mismo Newton para las derivadas y las integrales. Leibniz denominó *lingua characteristica* a este nuevo lenguaje y calculus ratiocinator al esperado cálculo, con el cual, la mente (Sería liberada de tener que pensar en las cosas en si mismas, y aún así todo funcionaría perfectamente)). Leibniz esperaba que estas nuevas ideas expandirían la capacidad de razonamiento mediante la reducción a un cálculo simbólico de mucha de la labor necesaria para descubrir como obtener determinada conclusión a partir de

unas premisas dadas, y como comprobar la corrección de una deducción. Dicho de otro modo, esperaba la existencia de un cálculo análogo al cálculo infinitesimal, pero un cálculo de razonamientos para tratar las deducciones con proposiciones. Pese al sueño de Leibniz, la lógica Matemática se iba gestando más lentamente; el desarrollo de las ideas, notación y formalismos adecuados para la obtención de conceptos similares al cálculo diferencial, en cuanto a potencia, para la lógica necesitó varios siglos más. Se puede hacer un símil con la división habitual de periodos históricos, aunque es preciso notar que no coinciden temporalmente con los periodos homónimos de la Historia Universal, y así dividir el desarrollo de la lógica en una Edad Antigua, que se corresponde con la lógica Tradicional (500 a.C–1847); una Edad Media, con el desarrollo de la lógica Simbólica (1847–1880); una Edad Moderna, en la que se introduce la lógica Matemática de manera formal (1880–1960); y una Edad Contemporánea, en la que surge la lógica Computacional (desde 1960 a la actualidad).

## 4.4 LOGICA MATEMATICA

Desde el surgimiento de la lógica hasta los tiempos actuales se le ha dado varios usos e interpretaciones de acuerdo al campo de su aplicación. En sus comienzos la lógica es caracterizada por Aristóteles con el término *organon* refiriéndose al carácter instrumental de la lógica en la filosofía. En su evolución, Leibniz trata de usarla en la interpretación de nuestros razonamientos hasta el desarrollo simbólico moderno con las obras de George Boole y Augustus De Morgan quienes plantearon casi simultáneamente los fundamentos de la llamada álgebra de la lógica.

En esta sección se describen las propiedades y leyes empleadas en la lógica proposicional, que derivan posteriormente en los fundamentos de la lógica de la programación. Este sentido explica la razón por la que la lógica ha servido también como base para el desarrollo de un nuevo paradigma de programación: Programación Lógica. La idea central de la programación lógica la podemos expresar con la conocida ecuación de Kowalski: Algoritmo = Lógica + Control de manera que el control (estrategia

para encontrar la solución) la dejamos en manos de la máquina, y el programador sólo se preocupa de la lógica (información acerca del problema que queremos resolver).

Hasta el momento no se ha definido que es la lógica, en ese sentido son muchas las definiciones de la lógica que pueden citarse y discutirse, pero es claro que todos los matemáticos, filósofos y diversos autores coinciden en que la lógica es el estudio de los métodos y principios que se usan para distinguir el razonamiento correcto del incorrecto.

Considerando los aportes de Claude Shannon y Chomsky se puede agregar que una computadora puede ser programada para tomar decisiones basadas en ciertas proposiciones.

Se distinguen varios tipos de proposiciones, existen proposiciones gramaticales y proposiciones lógicas (Copi, 1987). Las lógicas son aquellas en las que se puede determinar su valor de verdad y por tanto podemos decidir si son verdaderas o falsas, que en el ámbito computacional pueden interpretarse como ceros y unos en relación a la correspondencia biunívoca con el encendido o apagado de los estados básicos y fundamentales de los millones de circuitos que conforman la computadora. En consecuencia, el estado encendido para un circuito se representa por el número **uno**, y cuando está apagado se representa con el **cero**.

Las proposiciones lógicas pueden ser proposiciones simples o proposiciones compuestas. Las proposiciones compuestas están formadas por dos o más proposiciones simples unidas por elementos lógicos llamados conectores, conectivos u operadores lógicos. También se les denominan compuertas lógicas en la lógica digital.

Los operadores lógicos son de dos tipos: Los unarios, como el caso de la negación (~) y los binarios.

Los operadores binarios más usuales son:

La conjunción : Se representa con símbolos tales como: ^, &, and, “,” (es decir, la coma)

La disyunción : Utilizada con los siguientes símbolos: “o”, OR, v.

La implicación : Usa representaciones como las siguientes: →, ⊃

La bicondicional: Matemáticamente se usa como si y solo si, se representa con estos símbolos: ↔, ≡.

Precedencia: Los símbolos de los conectivos en la lógica tienen un orden de precedencia, se usa de la siguiente manera: ~, ^, v, →, ↔

Uno de los fines de la lógica computacional es la aplicación de lógica formal para la representación formal de argumentos, las técnicas de deducción automática o asistida por computadora (Solís, 2000) que conlleva la prueba de teoremas, tema con mayor interés entre 1960 y 1970. Este método fue desarrollado por Herbrand en 1930, pero no fue hasta 1960 cuando Gilmore lo implementó en una computadora digital. En 1965 Robinson mejoró la prueba mecánica de teoremas desarrollando una regla de inferencia simple, conocida actualmente como el principio de resolución. Esta prueba ha sido aplicada en muchas área, tales como el análisis de programas, síntesis de programas, sistemas deductivos de preguntas y respuestas, sistemas para la resolución de problemas y tecnología robótica (Solís, 2000).

Aunque Church y Türing mostraron que no hay procedimiento de decisión general para probar la validez de formulas de lógica de primer orden, existen procedimientos para probar que una fórmula es válida cuando verdaderamente lo es.

En el caso de la programación de computadoras para estudiantes de secundaria especialmente para los principiantes, básicamente necesitan saber la lógica clásica referente al valor de verdad de los enunciados compuestos. Los enunciados compuestos en los programas representan las decisiones que el programa tomará de acuerdo al flujo de los datos. Por lo que el estudiante debe tener muy claro los valores de verdad de los enunciados compuestos que se forman utilizando los principales conectivos lógicos como ser la negación, la disyunción, la conjunción y la implicación.

Muy importante es que estos valores sean interpretados como Ceros y unos en vez de V y F como en la mayoría de los libros de lógica se describen. En el caso computacional tienen mucho mas sentido las interpretaciones digitales, estas interpretaciones se realizan representando únicamente dos estados: encendido y apagado que se asocian con el uno y el cero respectivamente. Las tablas de verdad para cada uno de los operadores lógicos anteriormente mencionados pueden ser:

La negación

| **P** | **~P** |
|---|---|
| 0 | 1 |
| 1 | 0 |

La disyunción

| **P** | **Q** | P v Q |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 0 | 0 | 0 |

La conjunción

| **P** | **Q** | P and Q |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |

La implicación

| **P** | **Q** | P → Q |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 1 |
| 0 | 0 | 1 |

La bicondicional

| **P** | **Q** | $P \leftrightarrow Q$ |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 1 |

Aunque el paradigma imperativo de la programación necesite algunos fundamentos de la lógica clásica, al menos en los inicios de la programación, tal como se describió en los párrafos anteriores, se usan principalmente en las estructuras secuenciales, condicionales y cíclicas, llamadas estructuras de control. Para Gutiérrez y Vargas, “las estructuras de control son los componentes mas elementales de la Lógica de Programación. Equivalen a las premisas de la lógica formal” (Gutiérrez y Vargas, 2000, p.7). Los programas bajo este paradigma se ejecutan en forma secuencial, por lo que las estructuras de control son las encargadas de guiar el orden de ejecución de las tareas o instrucciones, como puede verse el conocimiento de estas estructuras es fundamental. Al respecto Gutiérrez y Vargas agregan que lo más importante es saber como colocar las estructuras de control, por lo que proponen las siguientes reglas básicas de la lógica de la programación:

1. Regla de orden de ejecución. Todas las tareas requieren de al menos una salida y cero, una o mas entradas. Primero se ejecutan las acciones de entrada, luego las de proceso y finalmente las de salida.

2. Regla de descomposición. Un Proceso complejo puede dividirse en subprocesos más simples colaborando en la solución.

3. Regla de colaboración. La(s) salida(s) de un proceso puede(n) usarse como entrada(s) de otro.

4. Regla de cerradura. En todo lugar que haya un proceso puede colocarse una estructura de control completa.

5. Regla de dependencia. Los procesos son parte integral de las estructuras de control, no son estructuras independientes.

6. Regla de integridad. Las estructuras de control no se deben romper ni alterar (a menos que se sepa que se esta haciendo).

7. Regla de existencia. Todos los datos antes de usarse deben existir y si son parte de un proceso de cálculo debe tener un valor válido.

## 4.5 APUNTES DE PSICOLOGÍA Y LA NUEVA CIENCIA COGNITIVA

Cualquier modelo que intente mejorar la enseñanza de la programación de computadoras debe referenciar las fuentes que sustentan el desarrollo del pensamiento y sus aplicaciones en la enseñanza aprendizaje, dichas teorías provienen de la psicología y la ciencia cognitiva (Gardner, 1986; Glass y Holyoak, 1986; Jones e Idol 1990), de los modelos actuales que explican la inteligencia humana (Stenberg, 1985; y 1987; Gardner 1983; Goleman, 1986) y el paradigma de procesos (Sánchez 1985 y 1992).

Al hablar de computadoras y en consecuencia de programación de computadoras no se puede dejar de mencionar la ciencia cognitiva que por un lado trata de explicar el funcionamiento de los procesos mentales y por otro trata de crear una teoría que explique el funcionamiento del cerebro, y que a la larga permita construir modelos que simulen una réplica del mismo. En la búsqueda de un modelo que permita hacer tal descripción y explicación surge como una producción de primera, las computadoras y sus lenguajes de programación. Así, para 1956 se reúnen en el Dartmouth Collage, en Hannover, diez especialistas en matemática y lógica con el propósito de crear programas capaces de comportarse o pensar inteligentemente. Aunque a John Von

Newman suele acreditársele la noción de un programa almacenado, demostró que la lógica binaria y la aritmética podrían conjugarse en la conformación de estos programas almacenados (Gardner, 1996), de aquí se derivan otras ideas como la de los compiladores y los interpretes. Los compiladores son programas que leen todo el código y posteriormente lo ejecutan, mientras los intérpretes ejecutan cada línea inmediatamente después de su lectura. Los cimientos de la ciencia cognitiva comienzan desde septiembre de 1948, cuando científicos eminentes de diversas disciplinas se reunieron en los predios del Instituto de Tecnología de California donde se desarrolló el famoso simposio Hixson (Gardner, 1996). A partir de allí se hicieron intentos por crear lenguajes que permitieran hacer demostraciones completas de teoremas. Esta última idea es la moda en la programación lógica a nivel de ingeniería computacional que al final despunta en la inteligencia artificial.

## 4.6 LOS ESTILOS DE APRENDIZAJE EN LOS ALUMNOS

Para lograr con éxito los objetivos propuestos se debe tomar como aliado principal los métodos lógicos que contribuyen al desarrollo del pensamiento algorítmico de los alumnos, en consecuencia realizar en forma eficiente el trabajo que se deja a la madurez mental de los estudiantes.

Hay que reconocer, las clases que normalmente se imparten están dirigidas a la reproducción de conocimientos que se suponen todos los alumnos o la mayoría aprenden de la misma forma y manera. Por eso cuando un alumno pregunta, ¿que hago primero?, o dice: “voy bien”, sorprende y muchas veces hasta se cree que los estudiantes son retrasados o que no pueden entender lo que se les esta diciendo, o como si la culpa fuera de ellos. Se debe revisar muchos de los factores que influyen en el entendimiento de los alumnos, cuáles actividades de las que se desarrollan contribuyen al desarrollo del pensamiento algorítmico en función del desarrollo de las habilidades necesarias para la elaboración de programas. Para hacer efectiva la faena educativa es necesario familiarizarse con el ambiente, características físicas, psicológicas y genéticas de los estudiantes. Y en este caso los estudiantes de primero

de bachillerato en computación que tienen una edad comprendida entre los 15 y 16 años.

El conocimiento de las interioridades de la lógica de la programación y la madurez mental de los alumnos en ese nivel, facilita llevar a cabo eficientemente el proceso de la enseñanza-aprendizaje de la programación, en ese sentido es indispensable tomar muy en cuenta los estilos de aprendizaje (Ulrich, 2000) que los estudiantes poseen y emplean en su vida cotidiana, y también en las clases de programación cuando necesitan diseñar un programa para resolver un problema. Considerar que normalmente en los cursos de lógica y programación se realizan exámenes para diagnosticar la cantidad de conocimientos que el estudiante posee o mas precisamente lo que el estudiante logra recordar en el momento de las pruebas, elementos hasta cierto punto intrascendentes comparado con las habilidades de aprendizaje que el estudiante adquirió a lo largo de su recorrido académico en los curso y niveles anteriores. ¿Como el alumno aprende?, es más importante. Y mas importante todavía ¿Cómo el alumno aprende a programar mas fácilmente?

En el aprendizaje de la lógica de la programación no se puede prescindir de las estrategias de aprendizaje empleadas tanto por los estudiantes como por los maestros en el proceso enseñanza aprendizaje. Este aspecto es señalado por muchos investigadores,

> *El estilo personal de aprendizaje describe el camino que una persona recorre habitualmente para responder a una tarea de aprendizaje, suma el estilo cognitivo, que refleja el modo con el que el individuo piensa y la estrategia de aprendizaje, que refleja el proceso utilizado para responder a las demandas de la actividad de aprendizaje* ( Alonso, Catalina y Gallego, Domingo, 2003, Pág. 7).

Todos los esfuerzos que los docentes realicen por enseñar la lógica de la programación van a depender en parte del estilo de aprendizaje que el alumno utilice. Así, lo que es

lógico para una persona no lo es para otra, lo que es divertido para una persona es aburrido para otra o “lo que es aburrido para una persona, para otra es seguridad” (Ulrich, 2000).

Se debe conocer las características del grupo de alumnos de la que se es responsable, principalmente, el estilo de aprendizaje que utilizan en sus actividades diarias y académicas.

En muchas ocasiones ocurre que cuando el profesor explica el tema solamente es entendido por un pequeño grupo de alumnos, estos coincidentemente pertenecen al mismo grupo de estilo de aprendizaje que el profesor, por lo que el resto de los estudiantes de otros grupos de estilos de aprendizaje quedan excluidos de los objetivos de la clase. En ese sentido Alonso trata de explicar la relación existente entre el estilo de aprender de los alumnos y el estilo de aprender de los profesores, afirma que “es frecuente que un profesor tiende a enseñar como le gustaría que le enseñaran a él, es decir, enseña como a él le gustaría aprender, en definitiva enseña según su propio estilo de aprendizaje” (Alonso, 2002, p.44).

No se puede ofrecer una definición de estilo de aprendizaje, pero son muchos los investigadores que se refieren al tema. Para Alonso “Los estilos de aprendizaje son los rasgos cognitivos, afectivos y fisiológicos que sirven como indicadores relativamente estables, de cómo los discentes perciben, interaccionan y responden a sus ambientes de aprendizaje” (Alonso, 2002, p.45).

> *Los rasgos cognitivos que tienen que ver con la forma en que los estudiantes estructuran los contenidos, forman y utilizan conceptos, interpretan la información, resuelven problemas, seleccionan medios de representación (visual, auditivo, kinestésico), etc. Los rasgos afectivos se vinculan con las motivaciones y expectativas que influyen en el aprendizaje, mientras que los rasgos fisiológicos están relacionados con el biotipo y biorritmo del estudiante* (Cazau, 2003, Pág. 1)

Por otra parte, Felder y Henríquez afirman que los modos en que el individuo característicamente adquiere, retiene y recupera información se denominan colectivamente su estilo de aprendizaje (Felder y Henríquez, 2004). En términos más sencillos son los dones o tendencias individuales (Ulrich, 2000).

Actualmente los estilos de aprendizaje son utilizados en diversas disciplinas: en el desarrollo de software de ayuda en el aprendizaje de idiomas (Martín, 2004), investigaciones para diseñar estrategias de aprendizaje para cursos de aplicaciones de terapia ocupacional a disfunción física (Marrero, 2004)

Existen muchas clasificaciones respecto a los estilos de aprendizaje, algunas clasificaciones han sido realizadas en función de otras, en otros casos se han fusionado tomando posiciones eclécticas, al respecto Cazau realiza un resumen de la clasificación de los principales estilos de aprendizaje (Cazau, 2003). La clasificación con respecto al hemisferio cerebral que procesa la información puede ser: lógico u holístico. Según el cuadrante cerebral, Cazau cita a Herrmann, los estilos de aprendizaje pueden ser: cortical izquierdo, límbico izquierdo, límbico derecho y cortical derecho. De acuerdo al sistema de representación (PNL) tenemos el estilo visual, el auditivo y el kinestésico. Según el modo de procesar la información, Cazau cita a Kolb, pueden ser: Activo, reflexivo, pragmático y teórico. Para Felder y Silverman existen las categorías siguientes: Activo/reflexivo, sensorial/intuitivo, visual/verbal y secuencial/global. En cambio Gardner, quien las denomina inteligencias múltiples, encuentra que hay individuos con inteligencias: Lógico-matemático, Lingüístico-verbal, Corporal-kinestésico, Espacial, Música, Interpersonal, Intrapersonal y Naturalista.

Gregory, citado por Ulrich, propone un modelo para determinar el estilo de aprendizaje basado en cómo las mentes perciben y comprenden la información. En este modelo se distinguen dos cualidades perceptivas que corresponden a nuestro modo de ver el mundo, por lo que podemos percibirlo en forma concreta o en forma abstracta; pero el modo que se usa la información percibida, llamado orden, puede ser cualquiera de los dos métodos: secuencial o aleatorio. Tomando las definiciones anteriores se puede

formar cuatro combinaciones: Secuencial concreto, secuencial abstracto, aleatorio abstracto y aleatorio concreto (Ulrich, 2000).

Considerando los estudios de uno de los principales exponentes de los estilos de aprendizaje David A. Kolb (KOLB, A. y KOLB, D. A. 2001) se deriva una de las clasificaciones más conocidas, realizada por P. Honey y A. Mumford conservan la idea de un modelo de aprendizaje experiencial (Kolb, 1984) en cuatro fases que llaman: la experiencia, el regreso sobre la experiencia, la formulación de conclusiones y la planificación. En la medida donde estas fases son privilegiadas por los individuos, definen cuatro estilos de aprendizaje que corresponden respectivamente a "una descripción de las actitudes y conductas que determinan una manera de aprender preferida por un individuo" (Honey y Mumford, 1992). En los trabajos iniciales y en conjunto de Kolb y Fry identifican cuatro tipos estudiantes: convergentes, divergentes, asimiladores y acomodadores (Kolb, D.A. y Fry, R., 1975). De esta forma postulan la existencia de cuatro dimensiones unipolares, en vez de dos dimensiones bipolares como lo hace Kolb. En consecuencia los cuatro estilos de aprendizaje son: el estilo activo, el estilo reflexivo, el teórico y el pragmático.

El estilo de aprendizaje activo. Busca experiencias nuevas, son de mente abierta, nada de escépticos y acometen con entusiasmo las tareas nuevas. Piensan que hay que intentarlo todo por lo menos una vez. Tan pronto como desciende la excitación de una actividad, comienzan a buscar la próxima. Se crecen ante los desafíos que suponen nuevas experiencias, y se aburren con largos plazos. Son personas muy de grupo que se involucran en los asuntos de los demás y centran a su alrededor todas las actividades. Se destaca por ser: animador, improvisador, descubridor, arriesgado y espontáneo.

El estilo de aprendizaje reflexivo. Le gusta considerar las experiencias y observarlas desde diferentes perspectivas. Recogen datos, analizándolos con detenimiento antes de llegar a una conclusión. Son prudentes les gusta considerar todas las alternativas posibles antes de realizar un movimiento. Disfrutan observando la actuación de los demás y no intervienen hasta que se han adueñado de la situación. Crean a su

alrededor un aire ligeramente distante y condescendiente. Se destaca por ser: Ponderado, concienzudo, receptivo, analítico y exhaustivo.

El estilo de aprendizaje teórico. Enfoca los problemas de forma vertical escalonada, por etapas lógicas. Tienden a ser perfeccionistas integran los hechos en teorías coherentes. Les gusta analizar y sintetizar. Son profundos en su sistema de pensamiento, a la hora de establecer principios, teorías y modelos. Para ellos si es lógico es bueno. Buscan la racionalidad y la objetividad huyendo de lo subjetivo y lo ambiguo. Se destaca por ser: Metódico, lógico, objetivo, crítico y estructurado.

El estilo de aprendizaje pragmático. Su punto fuerte es la experimentación y la aplicación de ideas. Descubren el aspecto positivo de las nuevas ideas y aprovechan la primera oportunidad para experimentarlas. Les gusta actuar rápidamente y con seguridad con aquellas ideas y proyectos que les atraen. Tienden a ser impacientes cuando hay personas que teorizan. Pisan la tierra cuando hay que tomar una decisión o resolver un problema. Su filosofía es: “siempre se puede hacer mejor, si funciona es bueno. Se destaca por ser: Experimentador, práctico, directo, eficaz y realista.

Al observar las potencialidades de cada alumno, resulta claro que si logra adaptar la metodología de enseñanza del profesor al estilo predominante del alumno el número de alumnos que tendrá éxito será mayor. En consecuencia el docente tiene la obligación de presentar actividades que involucren estilos de aprendizaje específicos y mixtos que logren involucrar a todos los estudiantes, mejor aún si se logra diseñar actividades de tipo holístico.

Para determinar el estilo de aprendizaje de los estudiantes existen varios instrumentos disponibles para ese propósito. Sin embargo cabe destacar el cuestionario CHAEA o Honey-Alonso de estilos de aprendizaje que comprende ochenta ítems para el auto y heterodiagnóstico de estilos (Honey-Alonso, 2002).

Algunos estudios específicamente señalan que después de aplicar un diagnóstico de los estilos de aprendizaje CHAEA, facilita configurar, proponer y sistematizar diferentes estrategias de enseñanza con diversas actividades para el aprendizaje y la participación

de estos alumnos en determinados momentos del proceso general de enseñanza. Este cambio de perspectiva permitiría superar el modelo tradicional de la homogeneidad de los procesos de enseñanza y de los materiales didácticos, para centrar la atención en como aprende el alumno y con qué estrategias y materiales sugeridos por el docente lo hace mejor (Grau & otros, 2004).

Cuando se obtienen los resultados de aplicar el cuestionario CHAEA no se debe creer que un individuo posea un estilo puro. “Todas las personas destacamos en un estilo de aprendizaje especialmente, pero pueden darse casos de obtener en las que puntuaciones no sean tan claras” (Paule R., Salan, Aitor de la Puente, Pérez, P., y Gonzáles, R., 2002, Pág. 3). Cuando las puntuaciones son tan claras significa que el individuo posee características de varios estilos de aprendizaje predominado en el orden de las puntuaciones. Lo ideal desde el nacimiento es que el individuo desarrolle todas las potencialidades, pero en la mayoría de los casos no es posible por múltiples razones. Sin embargo, hay esperanza para los individuos que desean aprender un estilo de aprendizaje, pues “los estilos en los que un individuos tiene menos preferencia pueden ser aprendidos, desarrollados y mejorados”

No bastaría con identificar las potencialidades de los alumnos después de aplicar un test que determine su estilo de aprendizaje, muy importante también conocer las dificultades mas comunes que pueden enfrentar los estudiantes de cada grupo de aprendizaje.

Para los estudiantes que tengan predominancia alta en el estilo activo, el aprendizaje les resultará más difícil cuando tengan que exponer temas como mucha carga teórica: explicar causas, antecedentes, etc., asimilar o interpretar datos que no están claros, trabajar solos, evaluar de antemano lo que se va a aprender, hacer trabajos que exijan mucho detalle. Estar pasivo, puede ocurrir en actividades tales como: oír conferencias, monólogos, explicaciones de posiciones de cómo deben hacerse las cosas, etc. También les puede resultar difícil asimilar, analizar e interpretar gran cantidad de datos sin herencia.

En cambio, aprenderán mejor cuando puedan intentar cosas nuevas, nuevas experiencias, nuevas oportunidades. Competir en equipo, resolver problemas, representar roles, arriesgarse, encontrar personas de mentalidad semejante con las que se pueda dialogar, poder realizar actividades diversas.

Los estudiantes que tienen predominancia alta en el estilo reflexivo tendrán mayores dificultades en el aprendizaje cuando tengan que: ocupar el primer plano, actuar de líder, presidir reuniones o debates, representar algún rol, participar en situaciones que requieran acción sin planificación, hacer algo sin previo aviso, exponer una idea espontáneamente, verse obligado a pasar de una actividad a otra.

Los estudiantes de este grupo aprenderán mejor cuando se realicen actividades que les permitan: observar, reflexionar sobre las actividades, intercambiar opiniones con otras personas con previo acuerdo, trabajar sin presiones ni plazos obligatorios, revisar lo aprendido o lo sucedido, investigar detenidamente, reunir información, sondear para llegar al fondo de la cuestión, pensar antes de actuar, asimilar entes de comentar, escuchar, observar a un grupo mientras trabaja, tener posibilidad de leer o preparar de antemano algo que le proporcione datos. Tener tiempo suficiente para preparar, asimilar y considerar. Tener la posibilidad de oír los puntos de vista de otras personas, aún mejor, variedad de personas con diversidad de opiniones.

Las posibles dificultades que enfrentan los estudiantes que tengan predominancia en el estilo de aprendizaje teórico son: verse obligados a hacer algo sin un contexto o finalidad clara. Participar en actividades no estructuradas, de finalidades inciertas o ambiguas. Participar en temas abiertos. Verse ante la confusión de métodos o técnicas alternativas, o contradictorias sin poder explorarlos con detenimiento por improvisación. Considerar el tema poco trivial, poco profunda o artificial.

Los estudiantes con estilo de aprendizaje teórico pueden aprender mejor desarrollando actividades en las que puedan sentirse en situaciones estructuradas que tengan una finalidad clara, inscribir todos los datos en un sistema, modelo, concepto o teoría. Tener tiempo para explorar metódicamente las asociaciones y relaciones entre ideas, acontecimientos y situaciones. Participar en sesiones de preguntas y respuestas. Poner

a prueba métodos y lógica que sean la base de algo, sentirse intelectualmente presionado, participar en situaciones complejas, llegar a entender acontecimientos complicados, enseñar a las personas exigentes que hacen preguntas interesantes, estar con personas de igual nivel conceptual.

Para los estudiantes bajo el dominio del estilo de aprendizaje pragmático les resulta difícil aprender lo que está distante de la realidad, aprender teorías y principios generales, trabajar sin instrucciones claras sobre como hacerlo, comprobar que hay obstáculos burocráticos o personales para impedir la aplicación. Sin embargo, los estudiantes de este grupo pueden aprender mejor si se les expone ante modelos que pueden emular, elaborar planes de acción con un resultado evidente, dar indicaciones, sugerir atajos, tener la posibilidad de experimentar y practicar técnicas con asesoramiento o información de retornote alguien experto, ver que hay un nexo evidente entre el tema tratado y un problema u oportunidad que se presenta para aplicarlo, concentrarse en cuestiones prácticas, recibir muchas indicaciones prácticas y técnicas.

## 4.7 LA PROGRAMACIÓN Y LA RESOLUCIÓN DE PROBLEMAS

La generalidad de los docentes que imparten las clases de programación tienen como objetivo principal que los estudiantes puedan escribir programas considerando una sintaxis y una semántica correctas. Con el desarrollo de las diferentes actividades docentes los alumnos deben poder resolver los problemas que se le presenten con una adecuada solución computacional. Constituye una meta de cada profesional de la docencia de la programación contar con estudiantes capaces de escribir programas para cualquier situación que requiera una solución a través de un programa de computadora. Algunos estudios iniciales centran la atención en el desarrollo de una cultura computacional (Riestra, Velásquez, 1996) a través de la incorporación del estudio de los algoritmos en el currículo escolar.

Las ciencias de la computación, tienen entre uno de sus objetos de estudio la resolución de problemas por medio de la computadora. El punto de mayor interés en la didáctica

de la matemática, en el que la computadora ha jugado un papel fundamental en la resolución de muchos problemas insospechados, tales como la demostración del teorema de los cuatro colores que Santos Trigo destaca en el potencial de la tecnología en especial el uso de la computadora en la resolución de problemas (Santos, Trigo, 1997). En este orden se distinguen fundamentalmente dos aspectos: Por un lado el análisis de los problemas, y por otra, el diseño para su solución, dicho de otra manera: como escribir programas que permitan resolver estos usando la computadora. En las ciencias computacionales, a esta asociación entre la temática anterior y los lenguajes de programación se le conoce simplemente como *programación*. Al respecto, Solís afirma: "En el funcionamiento de las ciencias computacionales concurre la lógica computacional, la teoría de la computación y el análisis de algoritmos" (Solís, 2000, p. 4). Resume de esta forma lo que podría llamarse resolución de problemas en las ciencias computacionales.

Si en primer lugar, se focaliza en la resolución de problemas como uno de los componentes importantes de la programación debemos esclarecer que un problema es: Según el diccionario de la lengua española en su acepción matemática "Proposición dirigida a averiguar el modo de obtener un resultado cuando ciertos datos son conocidos" (Océano, 1998, p.907).

El caso anterior, es una definición de lo más general en cuanto al tema se refiere. Por otro lado, si se particulariza en el área matemática, Allan Schoenfeld conceptualiza que "un problema matemático es una tarea: a) en la cuál el alumno está interesado e involucrado y para la cuál desea obtener una resolución; b) para la cuál el alumno no dispone de un medio matemático accesible para lograr esa resolución" (Resnick y Klopfer, 1989, p.148). Schoenfeld distingue a través de esta definición que debemos discernir lo que es un problema y lo que no es un problema. Por lo que "una tarea **no es un problema** para una persona hasta que lo ha hecho propio" (Resnick y Klopfer, 1989, p.148). El problema son tareas que plantean dificultades al estudiante, que puede tener o no tener las herramientas matemáticas que le ayuden a resolverlo.

De esta manera ha existido cierta polémica sobre la diferencia que hay entre un ejercicio o un auténtico problema. Lo que para algunos es un problema por falta de

conocimientos específicos sobre el dominio de métodos o algoritmos de solución, para los que si los tienen es un ejercicio.

En las clases tradicionales, los alumnos están acostumbrados a que el profesor presente un tema, a continuación desarrolle unos ejercicios y finalmente asigne tareas para reafirmar el algoritmo descrito y los trabaje a su propio ritmo en la tranquilidad de su casa. Creyendo que de esta forma, hasta cierto punto válido, deja problemas para que el alumno enriquezca su dominio respecto al tema. Esto nos puede servir cuando necesitamos automatizar procedimientos, pero no para cuando necesitamos resolver problemas. Sin embargo se debe reconocer que en muchos casos la resolución de problemas requiere automatismo de procesos, de esa forma focaliza la atención en la lógica del problema, lo que permite resolver problemas mucho mas completos.

En el área computacional Garey y Jonson, citado por Solís, definen un problema computacional en los términos siguientes: “un problema será una pregunta general que debe ser respondida, la cuál usualmente contiene varios parámetros, o variables libres, cuyos valores se dejan sin especificar” (Solís, 2000, p.34). Aparentemente esta definición es alejada de la de Schoenfeld, pero si se observa detenidamente, hay algunos puntos coincidentes. En el área computacional se resuelve un problema cuando probando con diferentes valores el algoritmo devuelve resultados correctos, y en la búsqueda de un algoritmo el educando puede conocer o no las herramientas necesarias. Muy importante también, que el estudiante se interese por el problema, de lo contrario lejos de ser un momento de aprendizaje será un aprendizaje de calvario, si es que lo hay.

En el área de bachillerato en computación es indispensable afrontar la problemática de la enseñanza de la programación desde el paradigma de la resolución de problemas. Los programas no son más que algoritmos computacionales que resuelven un problema, y resolver un problema implica considerar toda la investigación relacionada con ese campo.

A través de la historia de la matemática son muchos los investigadores que han tratado de explicar como se resuelven problemas. Así George Polya en 1965 realiza uno de los primeros intentos por explicar como se resuelven problemas (Polya, George, 2002).

A grandes rasgos define cuatro fases:

1. Comprender el problema
   ¿Cuál es la incógnita?
   ¿Cuáles son los datos?

2. Concebir un plan
   ¿Se ha encontrado con un problema semejante?
   ¿Conoce algún problema relacionado con este?
   ¿Podría enunciar el problema de otra forma?
   ¿Ha empleado todos los datos?

3. Ejecutar el plan
   ¿Son correctos los pasos dados?

4. Examinar la solución obtenida
   ¿Puede verificar el resultado?
   ¿Puede verificar el razonamiento?

Más recientemente Schoenfeld (1985) propone un marco con cuatro componentes que sirvan para el análisis de la complejidad del comportamiento en la resolución de problemas:

| | |
|---|---|
| Recursos cognitivos | : Conjunto de hechos y procedimientos a disposición del resolutor. |
| Heurísticas | : Reglas para progresar en situaciones dificultosas. |
| Control | : Aquello que permite un uso eficiente de los recursos disponibles. |
| Sistema de Creencias | : Nuestra perspectiva con respecto a la naturaleza de la matemática y como trabajar en ella. |

De acuerdo a este esquema cada uno de tales componentes explica las carencias, y en consecuencia en este planteamiento, uno de los componentes que cabe destacar es el de las heurísticas. Son las operaciones mentales típicamente.

En la clase de programación se debe considerar el esquema de resolución de problemas propuesto por Polya, este plan se adapta mucho a la estructura utilizada por la computadora en el ciclo de procesamiento de los datos, entrada proceso salida, que además complementa con la estructura propuesta por Broda (2000), donde la enseñanza de la programación debe considerar la siguiente estructura:

Especificación + razonamiento + código = programa razonado

En la resolución de problemas se pueden utilizar una variada cantidad de estrategias para resolver problemas, pero también debe considerarse el proceso de la evaluación. Así, para los trabajos grupales o exposiciones se recomienda usar la propuesta de Santos trigo con la propuesta de la evaluación en la resolución de problemas (Santos, Trigo, 1997), valorando el proceso utilizado por los estudiantes al resolver los problemas, en nuestro caso materializado en el desarrollo de los algoritmos, por lo que se deberá considerar las etapas siguientes: El estudiante debe mostrar que entiende el problema, por ejemplo enunciar el problema con sus propias palabras o representar el problema usando diversos caminos, juzgar las condiciones del problema y si es posible estimar alguna solución; La segunda etapa contempla la habilidad para diseñar e implementar un plan; y finalmente, revisar los aspectos relacionados con lo razonable de la solución y la extensión del problema.

## 4.8 APTITUD COMPUTACIONAL

Un aspecto muy importante a considerar es que los grupos de estudiantes de computación son grupos heterogéneos, que en muchos casos se matriculan en el bachillerato en computación sin tener las aptitudes necesarias para el buen desempeño en el área informática especialmente en la programación de computadoras. Por tanto es necesario conocer algunos aspectos de los estudiantes que cursan las clases de programación y saber que tipo de actividades desarrollar para lograr el máximo aprovechamiento y disminuir los índices de reprobación. En términos generales se puede usar un test que identifique el estilo de aprendizaje predominante en el grupo. Lo ideal es aplicar un test aptitudinal para detectar alumnos con aptitudes para la programación de computadoras, para no desperdiciar esfuerzos en alumnos sin aptitudes computacionales. En ese sentido podemos optar por el test CPAB o el COAB. Los estudios realizados en relación a las pruebas anteriores demuestran que tanto el test CPAB como el COAB pueden predecir en forma satisfactoria el desempeño de un estudiante en una carrera de carácter computacional o en un trabajo orientado en la misma dirección (Willoughby, 1980). Pero el medio educativo nacional pretende extender los servicios educativos a la mayor cantidad posible de estudiantes, y es allí donde comienza el reto. Lograr que muchos de los estudiantes desarrollen habilidades mínimas de programación. El país necesita la mayor cantidad de personas profesionales, y el área computacional es una de las pocas oportunidades existentes. Sin embargo para fines de excelencia académica los grupos deberían seleccionarse utilizando el test de habilidades computacionales para tener un mejor desarrollo en las clases relacionadas con la programación de computadoras, para el resto de los alumnos debe utilizarse los resultados de las pruebas de estilos de aprendizaje que deben ser el comienzo de la planeación educativa en las clases de programación.

Así para desarrollar con una mayor eficiencia las habilidades de programación se propone varios aspectos respecto al profesor y respecto a los alumnos. El profesor sin duda alguna debe aplicar un test inicial para detectar el tipos de estudiantes que conformaran su grupo de enseñanza aprendizaje en cuanto al estilo de aprendizaje y la cantidad de conocimientos que el alumno posee, aunque esto último no es tan indispensable hasta cierto punto pues en el desarrollo de un curso de programación

son mucho mas importantes las habilidades lógicas por lo que todos los esfuerzos deben centrarse en el desarrollo de esas habilidades. Teniendo siempre presente que lo mas importante es enseñar a programar y no que los estudiantes aprendan un lenguaje de programación, porque cuando un estudiante aprende a programar puede codificar sus algoritmos en cualquier lenguaje de programación. En los cursos que normalmente se desarrollan los profesores enfatizan en el lenguaje de programación descuidando la lógica de la programación, queriendo enseñar habilidades que deben emanar de un proceso bien estructurado partiendo de lo conocido y fácil como pueden ser los algoritmos con estructuras secuenciales de resolución practica y factible de acuerdo a los conocimientos básicos que un estudiante posee en el nivel académico de un doceavo grado.

A lo anteriormente propuesto también es de suma importancia señalar que siendo un tema de mucha complejidad, entre otros aspectos, involucra: el estilo de aprendizaje, las estrategias de razonamiento, los métodos lógicos que utilizan los alumnos para resolver los problemas, etc. Se puede apreciar en consecuencia una conexión con la lógica que a su vez involucra la lógica matemática y en particular la lógica computacional que es la disciplina que le da sentido a los algoritmos computacionales. Todo esto obliga a conocer mucho sobre el funcionamiento de la computadora, sus precursores, las bases teóricas de su construcción y sobre todo su evolución y aplicación en particular en el desarrollo del software. Teniendo la claridad de todos estos elementos se puede hablar categóricamente de la lógica de la programación. Por lo que se analizará cada aspecto con mucho detenimiento con el fin de tener una mejor comprensión de los factores que afectan la correcta o incorrecta forma, técnica o estilo en el momento del diseño e implementación de programas computacionales.

## 4.9 ANTECEDENTES

En Honduras no existe ningún estudio sobre la enseñanza de la lógica para estudiantes de de nivel medio, mucho menos investigación alguna en la lógica de la programación y como enseñarla en forma efectiva.

Lo mas cerca a nuestra temática es el movimiento impulsado por algunos profesores de la Universidad Nacional Autónoma de México denominado Taller didáctico de la Lógica primera edición (Morado, 1999) y Taller didáctico de la Lógica segunda edición (Campirán, 2003). Sin embargo ninguna de sus publicaciones ha particularizado el caso de la enseñanza y desarrollo de la lógica de la programación. Sus talleres e investigaciones han sido orientados a la enseñanza-aprendizaje de la lógica en general y la didáctica de la lógica jurídica como caso particular de la didáctica de la lógica.

Al considerar que la programación de computadoras es una disciplina relativamente nueva es mucho pedir tener un aparato definido para la didáctica de la lógica de la programación. En nuestro país las computadoras comienzan a popularizarse en el segundo lustro de los años noventas y la privatización educativa obliga a la creación de colegios con orientación computacional. En todas estas instituciones la programación se enseña sin considerar los procedimientos lógicos que intervienen en la programación de computadoras. Los profesores se limitan a conocer únicamente la lógica de los operadores lógicos tales como la negación, conjunción y la disyunción. Pero que a los alumnos se les muestra únicamente la punta del iceberg de la lógica que en la mayoría casos ni se menciona como una rama de las matemáticas. Para enseñar programación se leen los textos que están escritos en función de un lenguaje programación como primer parámetro y no de la lógica de la programación como debería de ser. Estos textos están escritos con el propósito de conocer la sintaxis de los lenguajes de programación y ningún momento en contemplan un diseño didáctico para aprender la lógica de la programación. Para muchos alumnos es imposible comprender la lógica de la programación cuando apenas conocen la operación de algunos programas de computadora, para esto es necesario conocer muchos de los principios básicos sobre los que ha sido construida la computadora y los programas de computadora.

A nivel internacional, hasta en el año 2002 se realizó en Salamanca España el primer congreso internacional sobre la enseñanza de la lógica, por lo que es una disciplina matemática relativamente nueva sin mayor experiencia en la enseñanza de la lógica, mucho menos en la enseñanza de la lógica de la programación.

Sin embargo, existen algunas teorías que explican como la programación es aprendida. Una de las teorías es el modelo de “esquemas de crecimiento”, aquí el estudiante construye “cajas de herramientas“ de cómo resolver clases de problemas en particular. Puede verse en esta teoría, que el alumno comienza conociendo algoritmos que resuelven problemas pequeños que posteriormente sirven para resolver otros mas complejos en los que involucran los esquemas iniciales. Al respecto Davies (Davies, 1994) ofrece evidencia empírica para los cambios en la estructura del conocimiento de la programación que desarrollan los expertos.

Una segunda teoría es que los programadores expertos tienen esquemas mas completos que los novatos, aunque no necesariamente mas sino mejor. En ese sentido los novatos tienden a focalizar en palabras claves en la estructura del problema mas que en la profundidad de la estructura del problema (Davies, 1994).

Otra teoría es la propuesta por Mayer en la que afirma que aprender a programar incluye “adquirir un modelo mental del sistema subyacente” (el modelo de la computadora). Un modelo mental es una metáfora que consiste de componentes y reglas de operación las cuáles son análogas a los componentes y reglas de operación del sistema. (Mayer, 1985). Pero también algunos investigadores han encontrado que un apropiado entrenamiento en modelos de procedimientos mejoran la habilidad de los estudiantes para aprender a programar: “Estudiantes que tuvieron un preentrenamiento en procedimientos aprendieron mucho mas rápido que los que tenían preentrenamiento… una conclusión completa es que la comprensión de procedimientos es un componente de la destreza en el aprendizaje de la programación, y puede ser enseñada a los novatos (Mayer, Dick y Vilberg, 1986).

Gutiérrez y Vargas (2000) hacen una propuesta para la enseñanza de la lógica de la programación planteando un modelo de aprendizaje integral sobre un mapa mental y

una estrategia incremental, pues consideran que la lógica de la programación debe de hacerse desde el punto de vista mas bien semántico de las estructuras, a través de símbolos visuales y acompañado de reglas simples y claras en un proceso de espiral creciente sobre diversos temas involucrados. Son de la convicción que la lógica de la programación es una habilidad mental superior y se desarrolla a través de experiencias que deben ser planeadas en temática y complejidad. Esta propuesta difiere de las teorías anteriores en el sentido de que se realiza en función de la experiencia adquirida a lo largo de la carrera docente, en cambio las teorías anteriores son más resultado de estudios psicológicos que tratan de explicar como el estudiante aprende a programar.

Por otra parte, se hacen esfuerzos para que los lenguajes de programación sean cada vez más fáciles de utilizar, en ese sentido muchos paradigmas de la programación están orientados a los lenguajes naturales, considerando que los lenguajes de programación pueden aprenderse igual que el lenguaje humano, pues presentan las mismas dificultades (Sustrick, 2006).

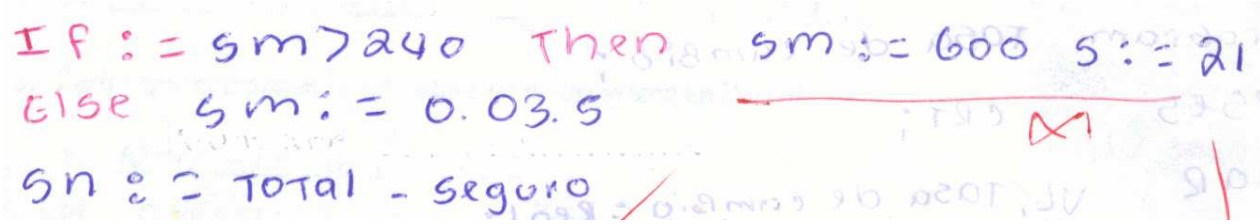
# CAPITULO II

# *5. Análisis e interpretación de resultados en la valoración cualitativa*

Con este capitulo se intenta mostrar evidencia de lo que ocurre cuando los estudiantes fallan al estructurar un algoritmo, que procedimientos utilizan. Así como también tratar de observar los métodos y procedimientos que utilizan los estudiantes que al estructurar un algoritmo y en consecuencia al escribir un programa tienen éxito. Para mostrar algunas evidencias se revisan los exámenes de admisión aplicados en el Instituto Primero de Mayo de 1954 a los aspirantes a ingresar al bachillerato técnico en computación (anexo 21 y anexo 22) después de recibir una semana intensiva de 4 horas clase diarias, el examen ha sido el mismo a lo largo del periodo 2002-2005. en el instrumento de evaluación se considera el problema planteado de la siguiente manera: Calcular el sueldo neto de 10 empleados de una empresa en la que se pide el valor del sueldo base, el valor de la hora extra, numero de horas extras; con esta información calcular el pago por sueldo extra, la deducción por Seguro social, el valor por Fosovi, deducciones, sueldo devengado, sueldo neto y además contar cuantos empleados ganan mas de 600 lempiras; todos los resultados anteriores deben visualizarse en pantalla como salida. Lo principal que se observa es donde los estudiantes muestran mayor dificultad, considerando los puntos neurales en la codificación del algoritmo como establecer la condicional para decidir si el sueldo es mayor que 600, y además observar si establecen una estructura cíclica para realizarlo con los diez empleados de la empresa señalada. En el resumen del anexo 21 y en el gráfico del anexo 22 puede verse que a lo largo de los años se tuvo mayor dificultad en el establecimiento de la condicional en promedio en este periodo un 16% lo hizo bien y únicamente el 13% de los estudiantes en promedio pudieron incluir una estructura cíclica en los cálculos para los 10 empleados. Hay que recordar que de acuerdo al teorema de Jacopini las estructuras necesarias para el desarrollo de un algoritmo son las estructuras condicionales y las cíclicas, con estas herramientas son suficientes para construir cualquier algoritmo. Y si estas estructuras son incomprendidas no se puede escalar hasta la complejidad en el desarrollo de algoritmos.

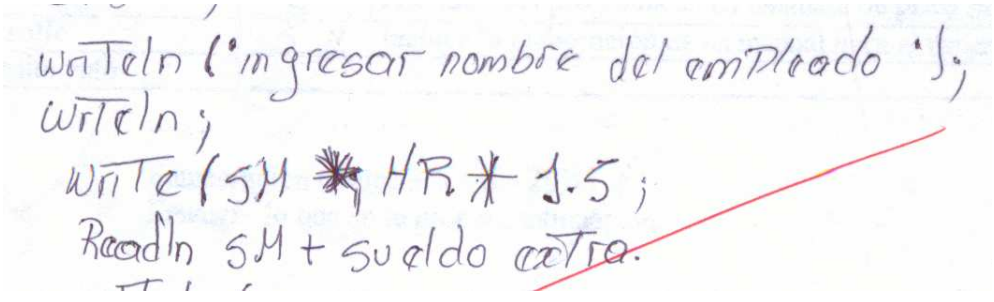
A continuación se mostraran algunos casos extremos del trabajo desarrollado por los estudiantes durante el intento por escribir un programa que resuelva el problema planteado anteriormente.

Figura 1.




En este caso puede verse que el estudiante no distingue la expresión lógica que permite realizar una tarea u otra en el algoritmo al expresar sm>240 para calcular correctamente el seguro social de uno de los empleados, en cuanto a la sintaxis del lenguaje se ve una inclusión incorrecta de := después del condicional IF. Por otra parte se observan dificultades para el cálculo de porcentajes, normalmente los estudiantes escriben el valor del porcentaje como el porcentaje mismo sin especificar la cantidad de la desean extraer la porción. También se observa como hay incongruencias lógicas al tratar de hacer una doble asignación después de la sentencia THEN que puede interpretarse como dos instrucciones diferentes, o como una sola con doble asignación que por sintaxis del lenguaje esta incorrecto. En este caso también trata de calcularse el total como la diferencia entre el valor del seguro social y un total que aun no ha sido calculado, esto evidencia un problema común en situaciones de organizaciones lógicas de secuencias al tratar de establecer el orden de ejecución en las sentencias.

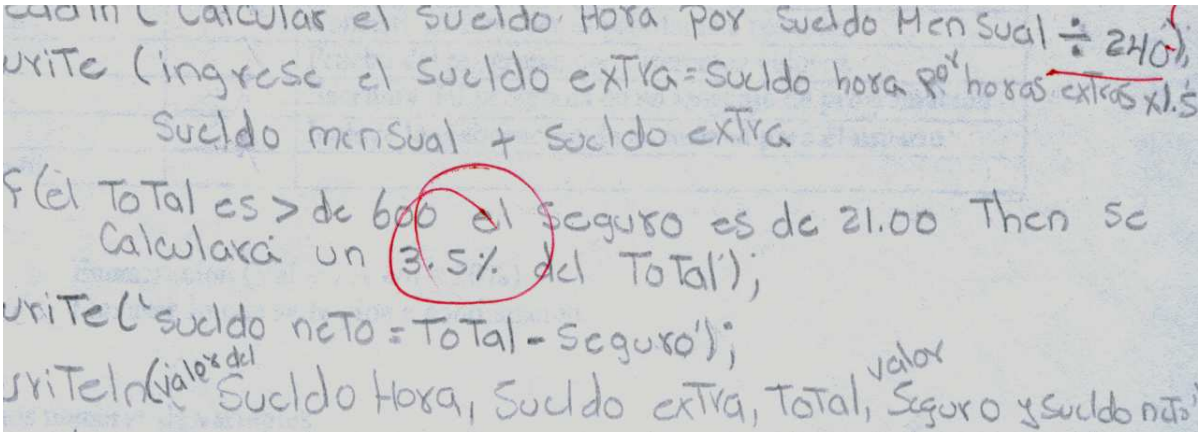
Figura 2.




En este caso se puede observar que el estudiante no distingue las instrucciones propias del lenguaje y que tampoco tiene una organización lógica de la información que en el desarrollo del algoritmo, no pudo separar los datos que forman parte de la entrada

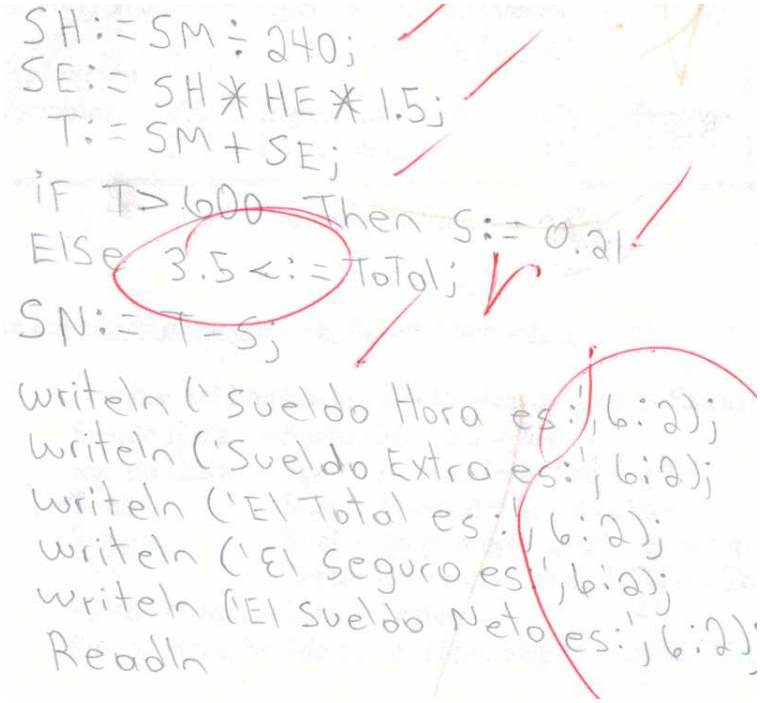
proceso y salida tal como lo establece el ciclo del procesamiento de datos en la solución de problemas computacionales. Otro aspecto importante es que no se aprecia el concepto de variable y mucho menos se comprende la idea de valores actuales de memoria, no se tiene una idea organizada de los cálculos por realizar y de cuales no se deben realizarse.

Figura 3.



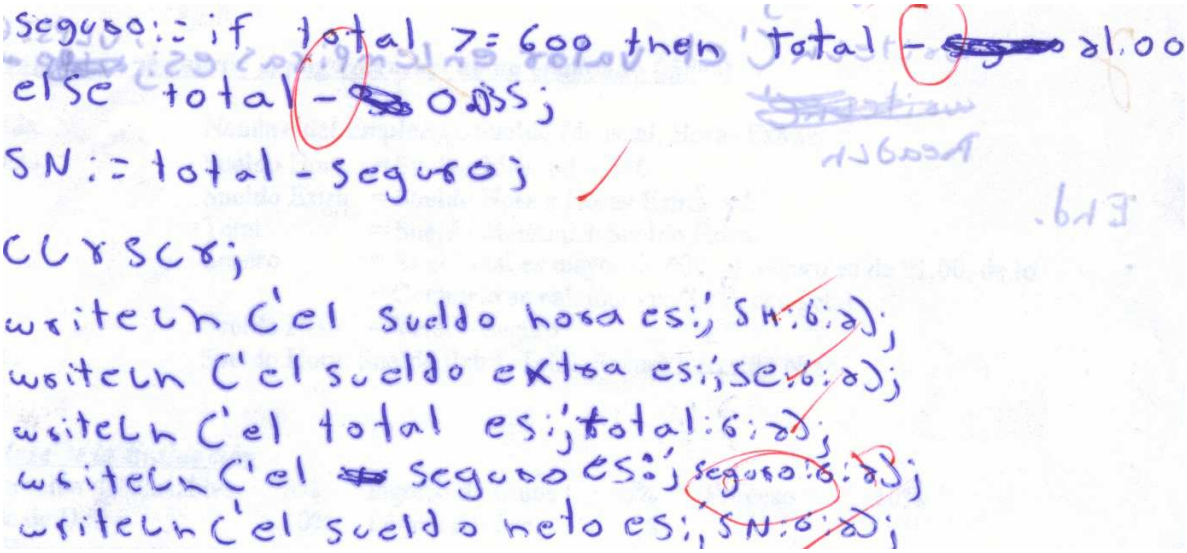

En este caso se evidencian varias dificultades que se pueden destacar: Primeramente se observa que igual que en el caso anterior no existe claridad en la organización de los datos de acuerdo al ciclo del procesamiento de los datos. No distingue lo que puede mostrarse como resultado de lo que debería mostrarse, en ese caso escribe ingrese el sueldo extra=sueldo por hora x 1.5. No comprende el concepto de variables de memoria y consecuentemente la idea de asignación, se puede observar al escribir sueldo mensual + sueldo extra. También se considera que hay dificultades en el establecimiento de la condicional para determinar el valor del pago por seguro social, se observa que trato de expresarlo en sus propias palabras y de buena forma, por lo que la dificultad en este caso más parece estar en la sintaxis del lenguaje. Considerando esta posibilidad también tiene dificultades en el cálculo del sueldo neto, pues se lo considera como la diferencia entre el total que no ha esclarecido y el seguro. Para concluir se ve que realiza la visualización de una forma incorrecta.

Figura 4.



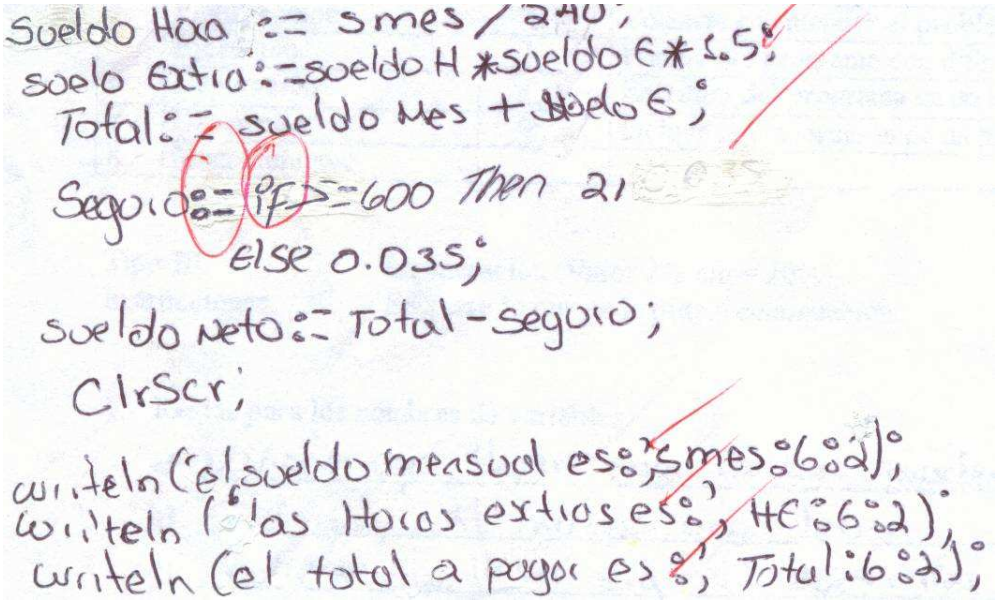

En este caso primeramente se puede ver que se calcula correctamente el sueldo por hora únicamente teniendo dificultad en la sintaxis al utilizar el símbolo ÷ en vez de / para establecer el cociente. En el caso de la condicional puede verse que se establece correctamente la primera parte, pero la segunda tiene dificultad al reafirmar el caso contrario escribiendo 3.5 <:= Total, es destacable la comprensión de la negación en el caso de los intervalos abiertos. En este caso es importante también señalar que hay ausencia de los resultados que se mostraran lo que equivaldría en otra rama de la matemática al realizar cálculos sin especificar el resultado solicitado o requerido. Por otra parte puede verse que hay una idea bastante clara acerca del uso de los lenguajes de programación en el caso especial de los formatos en las salidas de pantalla. También se puede observar que existe coherencia lógica en la estructura general y las dificultades están en pequeños detalles de la sintaxis.

Figura 5.



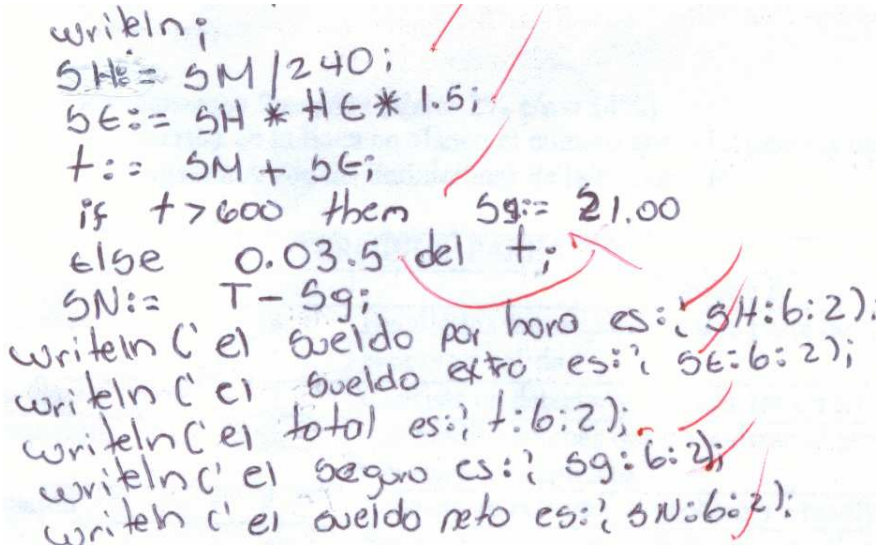

El estudiante en este caso trató de asignar en una variable el resultado de una condicional contradiciendo la sintaxis pero también se notan errores de lógica al establecer que total>=600, a esto se agrega un calculo incorrecto del valor del seguro social y para completar un calculo inadecuado de del sueldo neto. En el calculo del seguro se ve en la primera parte del condicional que no existe una variable donde se asigna el resultado de la diferencia igual que en la segunda parte considerando que ya lo hizo en la variable seguro.

Figura 6.




Aquí se puede resaltar que lo referente al algoritmo se encuentra bien desarrollado solo puede observarse que en el caso de la condicional se trató de asignar en una variable el resultado de cualquiera de los dos casos, por eso escribió: seguro:= if. Por otra parte se notan dificultades en el cálculo de porcentajes, cuando trata de calcular el 3.5% pero no establece de que cantidad. Y por último también se puede señalar que el cálculo del sueldo neto no se hizo de manera correcta pues el total no se calculó de forma correcta total.=sueldo mes + ¿?, y allí no se observa cual es el valor que quiere sumar.

Figura 7.



Este caso muestra una secuencia lógica muy adecuada pero nuevamente se ven dificultades en el momento de establecer la condición que permite hacer el cálculo correcto para el valor del seguro social, la expresión lógica que permite tomar la decisión de realizar el cálculo de una u otra forma se construyó bien pero no así el cálculo, los demás aspectos se ven bien establecidos.

Como puede verse en todos los casos anteriormente descritos, en la mayoría muestran dificultades en el momento de establecer los procedimientos en los que se involucran condicionales, otro de los aspectos que parecen ser muy típicos es la organización de las secuencias para que se puedan realizar los cálculos en forma correcta cuando un resultado depende de otro. En los primeros casos parece que los estudiantes no logran comprender que en un algoritmo los procesos tienen que expresarse como fórmulas en los que sus resultados deben asignarse en una viable de memoria o darle un nombre determinado. También en ninguno de los casos anteriormente descritos el estudiante detectó la necesidad de utilizar una estructura cíclica para resolver el problema en el aspecto de realizarlo con los 10 empleados de la empresa.

Un aspecto en común en todos los casos mostrados es que todos tienden a focalizar en palabras claves de la estructura del problema en coincidencia con Davies (1994), en este caso en palabras claves de la sintaxis del lenguaje de programación, para el caso en la figura 1, el estudiante trata de asignar el valor 600 para el seguro social sin considerar que puede tomar diferente valor de acuerdo al sueldo base ignorando totalmente la naturaleza del problema. En la figura 2 ocurre algo similar donde lo que trata el alumno es realizar cálculos independientemente de las relaciones que interesan en la resolución del problema.

Para culminar esta etapa donde se valoran los procedimientos utilizados por los estudiantes que al estructurar un programa fallan y los procedimientos que utilizan los estudiantes que al estructurar un programa lo hacen en forma correcta. Para esta parte se consideran un estudio de corte cualitativo pues no existe ninguna prueba que permita determinar las habilidades lógicas que posee un estudiante para poder desarrollar un algoritmo solución para un problema y de igual forma poder determinar que habilidades

lógicas son las que le dificultan el desarrollo de una solución algorítmica al mismo problema.

Cabe hacer notar que existen algunos exámenes de diagnóstico validados (anexo 14 y anexo 16) por algunos investigadores en el campo de la lógica desde el punto de vista filosófico, pero que no se han relacionado aún con la lógica y las dificultades en el pensamiento algorítmico. Sin embargo se han aplicado estos exámenes a un grupo de 30 estudiantes para poder determinar las habilidades lógicas que estos poseen y en que puntos muestran mayores dificultades. Se ha considerado este examen pues contiene reactivos que involucran las estructuras básicas necesarias para desarrollar correctamente un algoritmo solución.

En los resultados generales de la prueba (anexo 15) puede verse que en la pregunta 1, pregunta 2 y pregunta 3 que tienen que ver con la realización de asignaciones y sustituciones de variables tuvieron medianas dificultades. En los tres casos el número de incorrectas no superó el 50%, en la pregunta 1 las incorrectas fueron 43% en la pregunta 2 las incorrectas fueron un 50% y en la pregunta 3 un 40%.

Muchas dificultades se encuentran con la pregunta 4 que explora la comprensión acerca de la negación de una implicación y se observa que solo un estudiante la contesto correctamente, lo que representa sólo un 3%. Es importante destacar que en el desarrollo de algoritmos es de mucha utilidad principalmente cuando se trata de estructuras cíclicas que tienen que ver con ciclos controlados por tareas. La pregunta 6 reafirma la dificultad, solo un 40% la contestaron correctamente, y la pregunta numero 7 que también involucra la negación con un acercamiento mucho mejor a las estructuras condicionales de los algoritmos se ve que contestaron correctamente el 16.67% de los estudiantes, evidencia muchas dificultades en la comprensión de los intervalos abiertos. Y para confirmar, en la pregunta numero 8 un 13.33% de los estudiantes la contestó correctamente. Puede verse, que existen muchas dificultades en la comprensión de la negación de expresiones lógicas.

Los reactivos 9, 10 y 11 están orientados a detectar el reconocimiento de argumentos lógicamente correctos de los argumentos lógicamente incorrectos lo que daría una

dimensión mas formal en el conocimiento de la lógica, pero esto a nivel de estudiantes de secundaria es casi imposible pues en ninguno de los niveles inferiores se trata la formalidad de la lógica en ninguna de sus primeras manifestaciones. En el caso particular de la pregunta 10, puede verse que el 80% de los estudiantes contestaron correctamente, quizás guiados por el contexto, que en las preguntas 9 y 11 no aplicaron ninguno de los métodos lógicos necesarios y se guiaron mas por el azar. En el caso de la pregunta 11 puede verse que el 33.3% contestaron correctamente, pero hay que reconocer el nivel de dificultad de la situación.

La pregunta 12 fue contestada correctamente por un 23.3% de los estudiantes, en este caso había que considerar tanto la instanciación universal como la instanciación existencial, y eso representa un alto grado de dificultad. En el caso del reactivo numero 13, contestada correctamente por un 63.335 de los estudiantes, puede verse que el contexto favoreció notablemente la comprensión del argumento. Sorprendentemente la pregunta 14 solo fue respondida correctamente por un 33.3% de las personas, situación que modela un caso de implicación material, pues el hecho de que Romeo ame a una palabra de siete letras favoreció que los estudiantes consideraran el argumento incorrecto. Este efecto puede verse también en la pregunta número 15 en la que sólo el 20% de los estudiantes lo contestaron correctamente. En la pregunta numero 16 se percibe que los estudiantes definitivamente desconocen los métodos lógicos relacionados que permiten distinguir entre argumentos lógicamente válidos y argumentos que no son lógicamente inválidos, aquí se ve que únicamente el 3.33% de las respuestas fueron correctas.

En resumen los estudiantes contestaron el 33.96% de los reactivos en forma correcta y un 66.04% de manera incorrecta. Puede inferirse que existen muchas dificultades en la deducción de resultados lógicos a partir de premisas verdaderas o falsas.

En términos generales puede afirmarse que en este grupo de estudiantes desconocen en su mayoría los métodos lógicos que se utilizan para distinguir los argumentos lógicamente correctos de los incorrectos, y los valores de verdad para la negación de afirmaciones se realiza de igual forma incorrecta, situación que preocupa, pues en el desarrollo de algoritmos esta es una de las principales herramientas que se necesitan

considerando que esto involucra normalmente la toma de decisiones. Estas situaciones llevan a los estudiantes a comprender la lógica de los algoritmos por un proceso de maduración,  necesita darse cuenta de los procesos correctos de los incorrectos a través del ensayo y el error, pues en ningún otro momento o en anteriores se estudian estos métodos, por lo que la única oportunidad  aparecida hasta el momentos es cuando se estudian los algoritmos, en consecuencia se espera que tengan mas éxitos aquellos estudiantes que han experimentado un aprendizaje escolar basado en la práctica del ensayo y el error, muy diferente de aquellos que han sido educados bajo un modelo memorístico, cómo es tradicional en el país en los recientes años en la mayoría de centros educativos.

En conclusiones para este capitulo puede verse que los estudiantes normalmente tienen dificultades al momento de trabajar con las estructuras condicionales que de igual no pudieron responder correctamente en la prueba de diagnóstico aplicada al grupo de estudiantes que tomaron el curso de iniciación para el examen de admisión. Las dificultades en el campo de la lógica se manifiestan en la dificultad para detectar las falacias en argumentos que parecen ser válidos y que en muchos casos podrían ser detectados usando los ya conocidos métodos lógicos para determinar la validez o invalidez de un argumento

# CAPITULO III

# *6. Propuesta de una alternativa didáctica para la enseñanza de la lógica a los alumnos de primero de bachillerato en computación que conlleve al desarrollo del pensamiento algorítmico.*

Este capitulo es la sección donde se expone la solución dada al problema científico de investigación planteado en la introducción, incluye la valoración de un grupo de expertos que dan fe de la veracidad y validez de la propuesta.

En la primera parte se describe el modelo teórico que permite la elaboración de la alternativa didáctica que se presenta en la segunda parte, a continuación se muestran algunas situaciones que ejemplifican la propuesta didáctica y por último se realiza la valoración del criterio de expertos a partir de la aplicación del método Delphi.

Partiendo de la interrogante inicial: ¿Cómo enseñar lógica a los alumnos de primero de bachillerato de computación para facilitar el diseño de algoritmos que permitan resolver problemas computacionales?, siendo un problema científico tal como se ha considerado puede tener varias soluciones por lo que en esta parte se mostrará una solución, pudiendo existir muchas mas sin duda alguna. En esa dirección, cabe una alternativa didáctica para la enseñanza de la lógica de la programación, como se vio en los apuntes anteriores, es muy diferente de la programación lógica.

La alternativa didáctica se describirá en varias etapas, primeramente se describen las unidades que conforman el curso inicial de programación, y posteriormente se presenta el mismo curso pero con cuatro vías diferentes. En términos generales la alternativa contempla presentar del desarrollo de un curso de programación en función de los estilo de aprendizaje planteados por Honey y Munford, pero desde la perspectiva de la resolución de problemas combinando el modelo planteado por Polya y Schoenfeld (1985). En ese sentido se organizará el mismo curso describiendo las actividades docentes y dicentes de acuerdo a los cuatro estilos de aprendizaje.

La propuesta se realiza sobre la base del teorema de Jacopini (Joyanes Aguilar, Luís 1999), por lo que la atención es centrada en algoritmos que contengan las siguientes estructuras básicas: Secuenciales, condicionales y cíclicas. Considerando las estructuras por separado inicialmente, y luego combinándolas en el orden de dificultad. En los algoritmos que conducen a estructuras secuenciales incluir la idea del ciclo del procesamiento de datos que vislumbre la entrada de datos, el proceso y la salida.

## 6.1 UNA ALTERNATIVA DIDACTICA

Lo más probable es que los grupos de estudiantes que se formen en los cursos de computación sean heterogéneos en cuanto al estilo de aprendizaje. Por lo que buscar la estrategia de aprendizaje que mejor se adapte será un poco difícil. Es mucho más factible organizar subgrupos de trabajo en función de los estilos de aprendizaje y consecuentemente diversificar sus correspondientes actividades de aprendizaje.

Hay que tomar en cuenta que la tarea del proceso enseñanza-aprendizaje siempre será difícil, es muy importante diversificar las actividades que se realizan dentro del aula de clases, si el objetivo es que el proceso sea eficiente, entonces se sugiere planear actividades de aprendizaje para cada subgrupo de estilo de aprendizaje. Así, de acuerdo a lo explicado en el epígrafe 1.2.1 podemos tener cuatro tipos de estudiantes en los cursos de primer año, tomando en cuenta su estilo de aprendizaje con predominancia alta o muy alta: Pragmáticos, teóricos, reflexivos y activos. Sin embargo, pueden surgir estudiantes que no tengan muy definido su estilo de aprendizaje y en consecuencia la predominancia pueda indicar características de diversos estilos de aprendizaje en cuyo caso deben agregarse en el subgrupo con mayor afinidad, porque los estilos en los que un individuo tiene menos preferencia pueden ser aprendidos, desarrollados y mejorados, pero esto requiere un proceso de entrenamiento, largo y costoso (Paule R., Salan, A., Pérez, P., y Gonzáles, R., 2002) y no se tendrá el tiempo suficiente para realizar tal faena.

Considerando los aspectos anteriormente señalados en la metodología de la programación (Joyanes, 1999), los estilos de aprendizaje de Honey y Mumford (1992),

además, tomando en cuenta el modelo presentado por Gutiérrez y Vargas en el desarrollo de la lógica de la programación y las estrategias de resolución de problemas presentadas por Schoenfeld y Polya en los epígrafes anteriores y considerando que una tarea puede resultar muy sencilla para un individuo situado en la preferencia de un estilo de aprendizaje y, la misma tarea, puede resultar difícil para otro individuo con otro estilo de aprendizaje (Alonso Y Gallego, 2003). Se puede sugerir como alternativa didáctica el desarrollo de las siguientes actividades que permitirán al estudiante entender y desarrollar de una mejor forma la lógica de la programación bajo el paradigma imperativo:

| **ESTILOS DE APRENDIZAJE** | | | |
|---|---|---|---|
| **PRAGMATICO** | **TEÓRICO** | **REFLEXIVO** | **ACTIVO** |
| • Se deben desarrollar los algoritmos acompañados de continuas prácticas con la computadora, desestimando la carga teórica.<br>• Los ejercicios deben ser de situaciones reales con resultados prácticos.<br>• Proporcionar algoritmos de problemas que permita a los estudiantes encontrar soluciones alternas, así como realizar pruebas de escritorio sobre algoritmos preconstruidos.<br>• Los problemas deben estar relacionados con los temas que se desarrollan en forma sistemática.<br>• Los problemas | • Se deben desarrollar los algoritmos en papel, generalizando y teorizando en la mayoría de los casos. Tratando de acercarlos a la comprensión del funcionamiento de la computadora de a poco.<br>• El desarrollo de los contenidos debe ser en forma lógica y metódica.<br>• Se debe proponer a los teóricos que realicen los análisis de varios problemas para que puedan diseñar y generalizar algoritmos de solución.<br>• Proponer que inventen nuevas funciones para optimizar determinados programas<br>• Plantear problemas que impliquen pequeñas | • Se deben proponer algoritmos que presenten soluciones erróneas para puedan determinar las causas y soluciones.<br>• De acuerdo a la metodología de resolución de problemas deben realizarse exposiciones en los que se presenten diferentes alternativas de solución para un problema, de forma tal que se les pueda brindar la oportunidad de visualizar diferentes puntos de vista, preferentemente de personas con enfoques y opiniones distintas. Para este punto, se debe ofrecer el tiempo suficiente que les permita una preparación adecuada.<br>• Asignar muchos trabajos para desarrollar con tiempo suficiente, por lo que el trabajo en clase puede | • Se deben plantear problemas con soluciones inmediatas.<br>• Resolver problemas en grupo planteando competencia entre los grupos en la presentación de soluciones novedosas, eficientes y óptimas.<br>• Presentar problemas en orden de dificultad, planteando encontrar soluciones novedosas.<br>• Presentar algoritmos de solución mediante exposiciones sin mucha carga teórica. |

| | | | |
|---|---|---|---|
| deben tener instrucciones claras para lo que se le pide.<br>• Se deben resolver problemas ilustrativos, para que puedan seguirlos como modelos de emulación.<br>Proponer algoritmos desarrollados para que los puedan optimizar de forma tal que sean más rápidos y más eficientes. | demostraciones algorítmicas de fácil implementación, pueden incluir áreas como la teoría de números, álgebra y combinatoria. | resultar insuficiente. Pueden asignarse trabajos por semana o proyectos por mes según sea el caso.<br>Presentar una base teórica acerca del funcionamiento de la computadora y a continuación plantear problemas que puedan ser resueltos en grupo, para que puedan ser discutidos y analizados en forma exhaustiva. | |

Todas estas actividades pueden condensarse en un modelo de enseñanza de la lógica de programación que favorece la incorporación de aquellos estudiantes que muestran mayores dificultades en la maduración de la lógica de la programación por parte de aquellos estudiantes que tienen mayor disponibilidad y actitud ante las mismas dificultades. En el modelo se sugiere la conformación de grupos de trabajo por parte de los alumnos en los que se incluyan estudiantes de los diferentes estilos de aprendizaje, al menos uno de cada grupo. Esto como primera estrategia en el desarrollo de las clases. Esto se apoya en las afirmaciones de Mayer (Mayer, Dick y Vilberg, 1986) para quien enseñar a programar puede realizarse desde la perspectiva de un entrenamiento adecuado en la evolución y construcción de procedimientos para estudiantes novatos.

La evaluación de las diferentes perspectivas presentadas por los estudiantes debe conducir a una acumulación de estrategias que deben favorecer la implantación de un esquema subyacente que favorece el desarrollo de los algoritmos computacionales con el consecuente desarrollo de la lógica de la programación y la programación misma.

Se sugiere que los subgrupos ínter estilos puedan afrontar cada problema inicialmente de manera independiente por los integrantes del grupo, a continuación presentar las diferentes alternativas dentro del grupo y de esta forma evaluar diferentes puntos de

vista en la resolución del problema. Posteriormente a nivel de curso para homogenizar los resultados de cada uno de los grupos.

Estas actividades también pueden tener las variantes que dentro del subgrupo, se puede resolver un mismo problema con la aportación individual en cada una de las siete etapas del desarrollo de un programa tal como lo señala Joyanes Aguilar (1999).

Esta situación puede resumirse en el siguiente esquema que trata de representar en el círculo superior un grupo de estudiantes cualquiera en el que hay alumnos de todos los estilos de aprendizaje: activos, representados con una letra A; teóricos, con una letra T; pragmáticos, con una letra P y reflexivos con una R. En el círculo también se incluyen $G_1$, $G_2$,…, $G_n$ que representan los diversos grupos que deberían formarse para el trabajo en clase y cada grupo deben integrarse con estudiantes de los diferentes estilos de aprendizaje.

Las acciones didácticas inicialmente estarán orientadas para que cada estudiante de cada grupo resuelva el problema planteado lo que implica un trabajo independiente en la primera etapa.

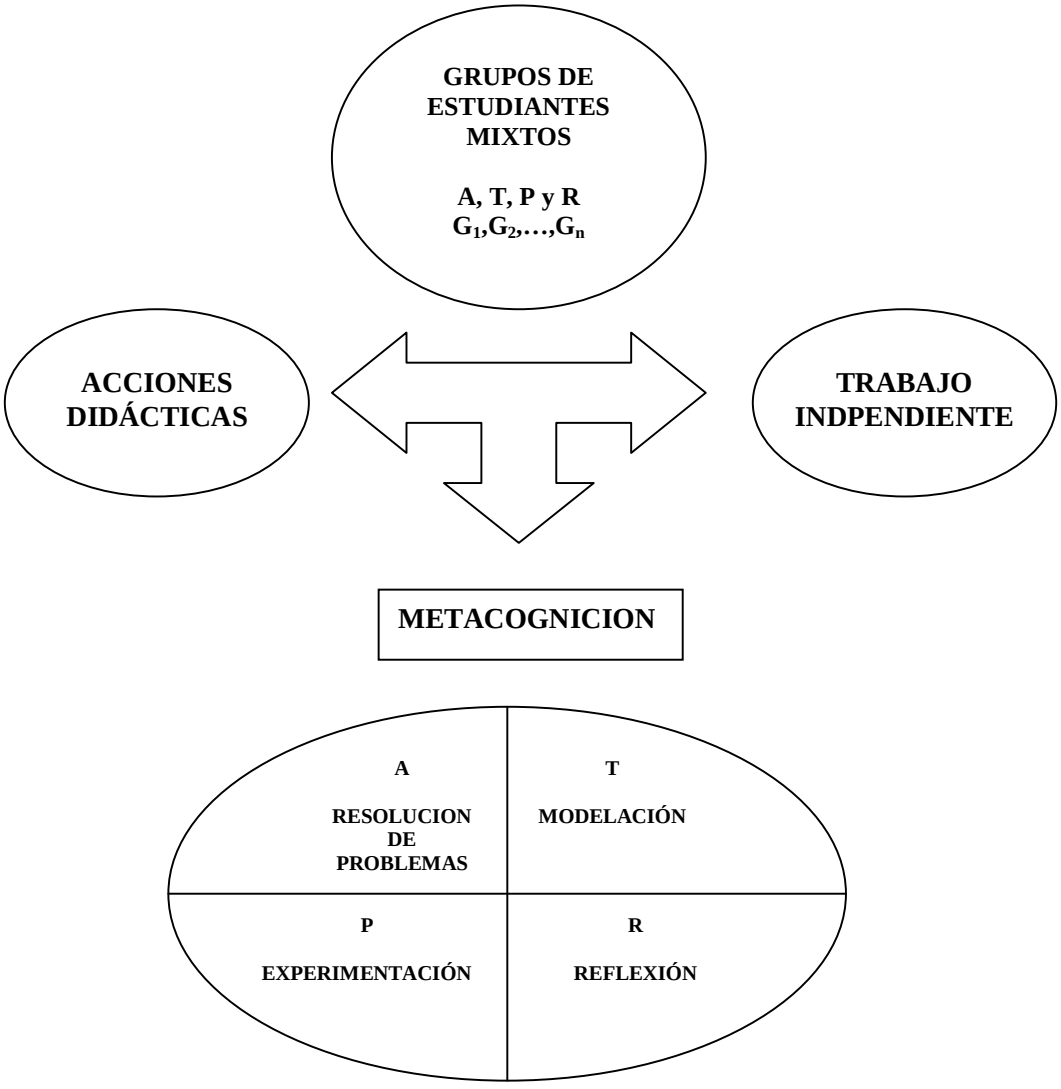
Cuando todos los integrantes del grupo hayan trabajado en resolver o intentar resolver el problema entonces pasaran a la segunda etapa que es la meta cognición donde al interior de cada grupo trataran de resolver el problema en forma conjunta, allí el aporte de los estudiantes con estilo de aprendizaje activo se verán retados por las competencias grupales y son a los que les atrae resolver problemas, ayudados por los teóricos trataran de crear un modelo por su capacidad para generalizar. En seguida participaran los estudiantes que tienen un estilo de aprendizaje pragmático quienes probablemente validaran las soluciones dadas al interior del grupo experimentando los resultados obtenidos, y por último los estudiantes con estilo de aprendizaje reflexivo examinaran todas las soluciones dadas para obtener diferentes puntos de visto y poder validar la mejor solución.

La tercera etapa comprende la interacción de todos los grupos de la clase, en los que se exponen los resultados obtenidos al interior de cada grupo, por lo que existirá una etapa

mas para la retroalimentación o la corrección. Si los problemas propuestos no lo resuelve ninguno de los grupos debe regresarse a los trabajos de grupo para replantear sus afirmaciones y procedimientos para validar los resultados.

## 6.2 DIAGRAMA DEL MODELO DE APRENDIZAJE PROPUESTO

**A**: ACTIVOS
**T**: TEÓRICO
**P**: PRAGMÁTICOS
**R**: REFLEXIVOS



En este y cualquier modelo de enseñaza aprendizaje de la programación es indispensable que los estudiantes puedan sentir la emoción de escribir un programa (Doouce, Chris, 2006), para lo que inicialmente deben sentirse parte de la solución de un problema.

Para el trabajo durante las clases se propone el siguiente esquema:

- Exposición del problema: El profesor da las instrucciones, distribuye eventualmente el material, se asegura mediante una discusión con los alumnos que las instrucciones tengan sentido para cada uno de ellos.

- Fase de búsqueda, de investigación: Los alumnos trabajan individualmente o en equipo. En esta fase, puede ser que las dificultades constituyan objeto de una discusión.

- Presentación de resultados: El profesor recoge los resultados y los hace comentar por la clase, o bien los equipos presentan su trabajo y lo someten a la crítica de los otros compañeros. Aquí los alumnos están obligados a convencer a sus compañeros de la validez de su respuesta o también, como es natural, dejarse convencer, rectificar sus errores o su mala interpretación. En cualquier caso, una argumentación sobre el problema debe desarrollarse. Ello puede dar lugar a nuevas

- Interrogantes, una extensión del problema o de los procedimientos utilizados.

- Fase de síntesis y de institucionalización: En la siguiente clase se resume la sesión anterior sobre el mismo problema. Los alumnos recuerdan el problema, las soluciones que ellos plantearon así como los métodos utilizados. Los alumnos comparan los métodos, sus ventajas y sus inconvenientes.

- Nivelación de la clase y evaluación:
  Se trata de:

- Ubicar a los alumnos en dificultades y aportarles información complementaria y explicaciones.
- Familiarizar al alumno con los nuevos conocimientos, que deben retener y dominar.
- Refuerzo y evolución de las concepciones: Es importante proponer a los alumnos problemas más complejos en los cuales las nuevas concepciones deberían funcionar. (Saenz, 2006)

## 6.3 Valoración de los criterios de los expertos sobre la propuesta.

En la presente sección se describen los resultados de la aplicación del método de expertos, utilizado para obtener criterios valorativos sobre la pertinencia de la alternativa didáctica expuesta en la sección anterior, como una solución al problema de investigación.

Para la aplicación del método se ha utilizado el siguiente procedimiento, ideado a partir del artículo de Campistrous y Rizo (Campistrous y Rizo, 2002) reseñada por García (García Pupo, Mauro, 2003):

1) Selección de los expertos.
2) Determinación de un conjunto de indicadores para medir la pertinencia de la alternativa propuesta como solución del problema de investigación.
3) Confección de una escala para medir los indicadores.
4) Confección de una encuesta para acopiar los criterios de los expertos.
5) Procesamiento estadístico de la información acopiada.
6) Análisis de los resultados.

Ahora se describe la aplicación del procedimiento anterior.

### *6.3.1 Selección de los expertos.*

Para seleccionar los expertos se tomó como población a un conjunto de docentes que imparten la clase de programación para alumnos novatos en el área de la programación de computadoras de las principales universidades de la ciudad de san Pedro Sula entre privadas y publicas. Todas estas universidades proveen de profesionales de la

computación para el sector empresarial, social y publico de San Pedro Sula y el resto del país. Los docentes escogidos tienen un mínimo de cinco años en la enseñanza de la programación a nivel superior e igual experiencia a nivel medio con diferentes paradigmas de la programación. En ese sentido se escogieron catedráticos de la Universidad Pedagógica Nacional Francisco Morazán, Universidad Tecnológica de Honduras, Universidad Católica de Honduras y de la Universidad Privada de San Pedro Sula.

Para seleccionar los miembros de la población que pudieran dar una mayor objetividad a la valoración de la propuesta (expertos), se utilizó un procedimiento que descansa en la autovaloración de éstos (Campistrous y Rizo, 2002), el cual se puede resumir en los siguientes pasos:

- Determinación del coeficiente de competencia de cada miembro de la población escogida ($k_c$).
- Determinación del coeficiente de argumentación de cada sujeto ($k_a$).
- Cálculo del coeficiente de cada sujeto (k).

  Así, $K=1/2 ( k_c+ k_a )$
- Valoración de los resultados.

### *6.3.2 Determinación del coeficiente de competencia de cada miembro de la población escogida.*

El coeficiente de competencia de los sujetos se determina por medio de su propia valoración. Para obtenerlo, se le pide a cada uno que valore su competencia sobre el tema en una escala de 0 a 10 en un instrumento que se le aplica (anexo 1).

### *6.3.3 Determinación del coeficiente de argumentación.*

Este coeficiente se calcula también a partir de la propia valoración de cada sujeto. Para su determinación se le pide (anexo 1) que indique el grado de influencia (alto, bajo, medio) que tiene en sus criterios cada uno de los elementos siguientes: análisis teóricos realizados por él mismo, su experiencia, los trabajos de autores nacionales, los trabajos

de autores extranjeros, su conocimiento del estado del problema en el extranjero y su intuición.

A las categorías alto, bajo y medio dadas por cada sujeto a los elementos anteriores, se les asignan números según se especifica en el Anexo 2, se suman estos números y se obtiene como resultado el coeficiente de argumentación del sujeto.

### *6.3.4 Cálculo del coeficiente de cada sujeto.*

El coeficiente de cada sujeto se calcula como la media aritmética de los coeficientes de competencia y de argumentación.

En el anexo aparece, de manera resumida, la información obtenida como resultado de aplicar el procedimiento explicado a los sujetos de la población seleccionada.

### *6.3.5 Valoración de los resultados de la selección de los expertos.*

Se puede observar que el menor valor del coeficiente k es 0,7 por lo que se consideraron a 11 expertos. Se descartaron 9 de las personas escogidas por no llenar el valor mínimo para k.

Los profesores escogidos han trabajo en el primer curso de programación, en el que los alumnos son novatos en el área de la programación. Todos han trabajado como mínimo tres años en el área de la lógica de la programación.

### *6.3.6 Determinación de un sistema de indicadores para medir la pertinencia del procedimiento propuesto para la solución del problema de investigación.*

Se considera que la variable independiente en esta investigación es la alternativa didáctica elaborada y la variable dependiente, la calidad del aprendizaje, entendido como la adquisición de habilidades en la resolución de problemas, de los conocimientos básicos del desarrollo de algoritmos para la solución de problemas.

El análisis bibliográfico realizado, seguido de consultas informales a distintos profesores y finalmente de la consulta a expertos, utilizando el método propuesto por Campistrous y Rizo (1998), ha permitido establecer dimensiones e indicadores muy útiles o imprescindibles para medir la variable dependiente según muestra la tabla del anexo 8.

Teniendo en cuenta estos indicadores, se formularon otros para medir la pertinencia del procedimiento didáctico elaborado con respecto a la solución del problema de investigación.

$I_1$: originalidad de la propuesta.

$I_2$: relevancia para la teoría.

$I_3$: calidad de la estructuración de los elementos que componen la propuesta.

$I_4$: utilidad para una distribución racional del tiempo a favor de las funciones discentes que más aportan a la comprensión.

$I_5$: utilidad para potenciar el aprendizaje asociado a los procesos matemáticos de resolución de problemas, razonamiento, comunicación, representaciones y conexiones.

#### 6.3.6.1 Confección de una escala para medir los indicadores.

Para la medición de los indicadores se utilizó una escala ordinal de cinco categorías como se indica a continuación:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| Inadecuado (I) | Poco adecuado (P.A) | Adecuado (A) | Bastante Adecuado (B.A) | Muy adecuado (M.A) |

#### 6.3.6.2 Confección de una encuesta para acopiar los criterios de los expertos.

Para valorar cada uno de los indicadores, se utilizó el escalamiento tipo Likert (Hernández, 1991), que consiste en un conjunto de ítems presentados en forma de afirmaciones o juicios ante los cuales se pide la reacción de los sujetos a los que se les administra (anexo 22). Existe una correspondencia biyectiva entre el conjunto de indicadores y el conjunto de ítems.

#### 6.3.6.3 Procesamiento estadístico de la información acopiada.

Para el procesamiento estadístico de los datos se utilizó el modelo de Torgerson (Campistrous y Rizo, 1998, p.13), utilizando como medio la hoja Excel soportada en Windows.

La aplicación de este modelo se realizó según el procedimiento siguiente:

- Se construyó una tabla de doble entrada para registrar las respuestas de cada experto a cada ítem (tabla 1 anexo 5).
- Se construyó una tabla de frecuencias absolutas tomando a los indicadores como variables y a las categorías de la escala como sus valores (tabla 2 anexo 6).

- Se construyó una tabla de frecuencias acumuladas absolutas a partir de la tabla del paso anterior (tabla 3, anexo 6).
- Se construyó una tabla de frecuencias acumuladas relativas a partir de la tabla construida en el paso precedente (tabla 4, anexo 6).

Cada frecuencia acumulada relativa que aparece en una celda de esta tabla se toma como la probabilidad de que el indicador tome el valor de la categoría correspondiente a esa celda o de categorías inferiores y se considera que los indicadores son variables distribuidas normalmente con varianza 1 y media 0.

- Se diseñó una tabla (tabla 5, anexo 23) que contiene:

1) El valor de la distribución normal estándar inversa para cada una de las probabilidades de la tabla construida en el paso anterior, (sin tener en cuenta la columna correspondiente a la categoría 5).
2) Las sumas de los valores anteriores por filas y columnas.
3) La media aritmética de los valores por filas y columnas.

   Los promedios de las columnas representan los valores de los límites superiores de las categorías (excepto la última), llamados puntos de corte.
4) El promedio general (N), es decir, el promedio de los promedios de filas.
5) La diferencias entre el promedio general y el promedio de cada fila. Cada uno representa en valor de escala del indicador correspondiente.

# *7. Análisis de los resultados de la aplicación del modelo*

Para analizar los resultados de la aplicación del modelo se ejecutaron dos acciones:

- Se construyó un gráfico lineal con los puntos de corte.

| MA | BA | A | PA | I |
|---|---|---|---|---|
| 5 | 4 | 3 | 2 | 1 |
| -0.03739297 | 1.148480981 | 1.299879615 | 1.382994127 | |

- Se analizó la pertenencia de los valores de escala a cada intervalo de valores de categoría. El resultado de este análisis permitió extraer como conclusión que todos los indicadores están comprendidos en la categoría de bastante adecuado.

Lo anterior significa que los expertos valoran como bastante adecuada la originalidad de la solución que se le da al problema de investigación, así como la contribución del modelo elaborado al enriquecimiento de la teoría.

De igual manera los expertos valoran como bastante adecuada la correspondencia de la estructura de la alternativa didáctica construida, con el modelo teórico seguido y las exigencias prácticas de los docentes.

En cuanto a la contribución de la alternativa a una redistribución del tiempo a favor de dedicar una mayor parte de éste a las funciones discentes que más aportan a la comprensión, los expertos consultados la valoran como bastante adecuada. De igual manera valoran con la misma categoría a la utilidad de la solución que se propone al problema de investigación para potenciar los procesos matemáticos de resolución de problemas, razonamiento, comunicación, representaciones y conexiones.

Por tanto, la aplicación del método de la consulta a expertos confirma que los mismos consideran que el procedimiento propuesto es válido como solución del problema de investigación

# *8. CONCLUSIONES*

Al finalizar este trabajo se puede concluir que:

1. Las estructuras de programación que deben afianzarse en los estudiantes son las condicionales y las cíclicas, es donde muestran debilidades de acuerdo a la valoración cualitativa, y que son las pequeñas herramientas que le permitirán resolver problemas mucho más complejos.

2. Las estructuras lógicas que deben reforzarse directa o indirectamente son los silogismos, así como el modus ponens y el modus tollens especialmente. Estructuras que se encuentran directamente ligadas al modelo lógico de los algoritmos computacionales.

3. El desarrollo de habilidades para resolver problemas puede facilitar el aprendizaje de los estudiantes en la medida que se aprovechan los estilos de aprendizaje. Esto permitirá mejorar el aprendizaje de la lógica de la programación que naturalmente es un proceso dejado a la maduración de las estructuras de la programación, y que permitirá favorecer el pensamiento algorítmico.

4. El dominio de la programación se favorece siguiendo el proceso de trabajar en forma independiente que permite a cada estudiante expresar sus propias ideas en un proceso de metacognición matemático que en el trabajo en grupo permite fortalecer la resolución de problemas con las potencialidades de cada individuo en función de los estilos de aprendizaje. Sustentado por la valoración del criterio de expertos.

# *9. RECOMENDACIONES*

Que la presente alternativa sea tenida en cuenta en la asignatura programación para los estudiantes de los bachilleratos en computación que reciben mucha teoría y pocas oportunidades de desarrollar habilidades mentales en relación a su campo de trabajo considerando que los institutos de educación media, del sector oficial principalmente, carecen de la infraestructura necesaria para un desarrollo óptimo en cuanto al desarrollo de la asignatura de programación.

Incorporar como parte de la política de selección y como parte de del currículo de las carreras computacionales el modelo propuesto para aprovechar el máximo de las potencialidades de los estudiantes que en otras circunstancias tendrán opciones limitadas en el desarrollo de la lógica de la programación.

Incluir en la programación de la clase de matemáticas que se imparte en las carreras computacionales tópicos de matemática discreta. Esto ayudará a que los estudiantes puedan ampliar su visión respecto a las estructuras matemáticas y facilitar el aprendizaje de la lógica de la programación consecuentemente ampliar los métodos de resolución de problemas.

Que los esquemas para el tratamiento diferenciado de los estilos de aprendizaje sean tomados en cuenta en aquellas carreras universitarias donde se desarrolla esta disciplina.

# *10. BIBLIOGRAFIA*

ALAS SOLIS, MARIO; HERNANDEZ RODRIGUEZ, RUSSBEL; MONCADA GODOY, GERMAN EDGARDO (2004). Propuesta de reforma de la educación media. Documento Base. Republica de Honduras, Secretaría de Educación. Págs. 72-80

ALONSO, CATALINA M. (2002). Los estilos de aprendizaje: procedimientos de diagnóstico y mejora. Quinta edición. Bilbao, Ediciones mensajero. Pág. 44, 48.

ALONSO, CATALINA Y GALLEGO GIL, DOMINGO J. (2003). Los estilos de aprendizaje: Una propuesta pedagógica. Ante el primer congreso internacional de estilos de aprendizaje.
Consultado: 15 de diciembre de 2005
www.ciea.udec.cl/trabajos/Alonso_Gallego.pdf

AYALA MOLINA, MADYA INES. (2001). Tipos de razonamiento y su aplicación estratégica en el aula. Editorial Trillas, México, primera edición.

BELL, E. T. (1995). Historia de las matemáticas. Fondo de cultura económica. México. Segunda edición. Pág. 121.

BIGGE, MORRIS L. (2001). Teorías de aprendizaje para maestros. Editorial Trillas. México. Decimoséptima edición.

BLUE, LENORE. (2004). Notices of American Mathematical Society. Octubre. Vol. 51 Numero 9. Pags. 1024-1034.

BRODA, KRYSIA. EISENBACH, SUSAN. KHOSHNEVISAN HESSAM. VICKERS, STEVE. (2000). Reasoned Programming.

C.E. SHANNON. (1938). A symbolic analysis of relay and switching circuits. Transactions of the AIEE, 57, 713–723.

CAMPIRAN SALAZAR, ARIEL F. (2003). La razón comunicada II. Editorial Torres Asociados, primera edición.

CAMPISTROUS PÉREZ, LUIS Y CELIA RIZO CABRERA (2002): Didáctica y Solución de Problemas. Soporte OREALC-UNESCO. La Habana.

CAREAGA, ALFREDO ALEJANDRO. (2002). El teorema de Gödel. Hipercuadernos de divulgación científica, Universidad Nacional Autónoma de México, mayo. Pág. 9.

CAZAU, P. (2003). Estilos de aprendizaje: generalidades. Párrafo 1.
Consultado: 20 de Octubre de 2004
http://pcazau.galeon.com/guia_esti01.htm

COPI, IRVING M. (1987). Lógica Simbólica. Cia. Editorial Continental, México, sexta impresión. Pág. 16.

C. S. T. A. (2003). A model curriculum for K-12 Computer Science: Final report of the ACM K-12 task force curriculum commite. October, 2003. Pág. 7

DAVIES, S. (1994). Knowledge restrucTüring and the adquisition of programming expertise. International Journal of human computer studies. 40, 703-725.

DICCIONARIO ILUSTRADO OCEANO DE LA LENGUA ESPAÑOLA. (1998). Mm océano grupo editorial. Barcelona.

DIJSTRA, EDSGER W. (1968). “Go to statement considered harmful”. Communications of the ACM, Vol. 11, No. 3, March 1968, Pages 147-148.

DOUCE, CHRIS. (2006). “The emotional programmer”. Newsletter: June 2006.
Consultado: 20 de noviembre de 2005
www.ppig.org/newsletter

E.L. POST. (1921). Introduction to a general theory of elementary propositions. In Jean van Heijenoort, editor, From Frege to Gödel. A Source Book in Mathematical Logic, 1879–1931, pages 264–283. Harvard University Press, Cambridge MA.

FARAON LLORENS, SERGIO MIRA (2004). Adn: una herramienta para la enseñanza de la deducción natural. Departamento de ciencia de la computación e inteligencia artificial. Escuela politécnica superior de la Universidad de Alicante.

FELDER, R. M. y HENRIQUEZ, E. R. (2004). Learning and teaching styles in foreign and second language education.
Consultado: 15 de Noviembre de 2005
http://www.ncsu.edu/felder-public/Papers/FLAnnals.pdf.

G. PARKINSON. (1965). Logic and Reality in Leibniz's metaphysics. Clarendon Press.

GARCIA PUPO, MAURO (2003). Material sobre metodología de la programación. Universidad de Holguín. Págs. 81-97.

GARDNER, HOWARD. (1993). Marcos de la mente, teoría de las inteligencias múltiples. Ediciones Paidós, Barcelona, segunda edición. Págs. 50-70.

GARDNER, HOWARD. (1996). La nueva ciencia de la mente, Historia de la revolución cognitiva. Ediciones Paidós, Barcelona, segunda edición. Págs. 33, 26, 165.

GLASS, A. y HOLYOAK, K. J. (1986). Cognición. New York: Random House.

GOLEMAN, D. (1986). Inteligencia Emocional. Barcelona: Kairos.

GOMES, LEE. (2005). Lenguajes informáticos proliferan para preocupación de la industria. Diario La Prensa, Año XVI, No 18996, Pág. 75.

GONALES PIENDA, JULIO; NUÑEZ PEREZ, JOSE; ALVAREZ PEREZ, LUIS; SOLER VAZQUEZ, ENRIQUE. (2002). Estrategias de aprendizaje, Concepto, evaluación e intervención. Ediciones pirámide. Madrid.

GONZALES, PIENDA Y NUÑEZ PEREZ JOSE CARLOS. (2002).Estrategias de aprendizaje, concepto, evaluación e intervención. Ediciones Pirámide, Madrid.

GRAU, JORGE E. & OTROS. (2004). Posibles Aplicaciones de la informatización del CHAEA. Fundación para el desarrollo de los estudios cognitivos.
Recuperado: 8 de febrero de 2006
http://www.fundec.org.ar/Principal/investigacion/Documentos/Trabajo2_Posibles%20aplicaciones%20del%20CHAEA.pdf

GUTIERREZ TORNEZ, AGUSTIN Y VARGAS DE BASTERRA, RICARDO. (1999). "La lógica de la Programación como Habilidad para Diseñar Algoritmos".

HERNANDEZ, R., C. FERNANDEZ Y P. BAPTISTA (1991). Metodología de la Investigación. McGraw-Hill Interamericana de México, SA de C.V. México.

HITT ESPINOZA, FERNANDO (1996). Investigaciones en matemática educativa. XX aniversario Departamento de matemática educativa Cinvestav-IPN. Grupo editorial iberoamérica. Págs. 330-351.

HODGES, ANDREW. (2002). Alan Türing
Recuperado: 24 diciembre de 2005
http://plato.stanford.edu/archives/sum2002/entries/Türing/

HONEY, PETER. Y ALONSO, C. (2002). CHAEA, cuestionario Money alonso.
Recuperado: 20 de Octubre de 2005
http://www.aprenditransfer.com.ar/chaea.shtml

JONES, B. e IDOL, L. (1990). Dimensions of thinking and cognitive instruction. Hillsdale, NJ: Lawrence Erlbaum Associates.

JOYANES AGUILAR, LUIS. (1999). Metodología de la programación, Diagramas de flujo algoritmos y programación estructurada. Editorial McGraw Hill, México. Págs. 119-135

KNUTH, DONALD E. (1974). Computer Programming as an art. Communications of ACM. December 1974, volume 17, number 12. Pág. 673.

KOLB, A. y KOLB, D. A. (2001). Experiential learning Theory Bibliography 1971-2001, Boston, Ma.: McBer and Co,
Recuperado: 15 de noviembre de 2005
http://trgmcber.haygroup.co/Products/learning/bibliography.htm

KOLB, D. A. (1984). Experiential Learning: experience as the source of learning and development. Englewood Cliffs, NJ, Prentice-Hall.

KOLB D. A. Y FRY, R. (1975). Toward an applied theory of experiential learning. En C. Cooper (ed) Theories of groups process, London: John Wiley.

MARRERO DIAZ, MILAGROS (2004). Estilos de aprendizaje y su impacto en el proceso enseñanza-aprendizaje en el curso TEOC 2007 Aplicación de terapia ocupacional en disfunción.
Recuperado: 22 de noviembre de 2005
http://cuhwww.upr.clu.edu/~ideas/Paginas_htm

MARTIN GAVILANES, MARIA DE LOS ANGELES (2004). Software de autor y estilos de aprendizaje. Didáctica (Lengua y Literatura) vol. 16, Pág. 105-116

MAYER, R (1985). Learning in complex domains: a cognitive analysis for computer programming. The psychology of learning and motivation, vol. 19, Academia press.

MAYER, R. DYCK, J., VILBERG W. (1986). Learning to program and learning to think: what's the connection? Communications of ACM, July 1986, vol. 29 no 7

MORADO ESTRADA, RAYMUNDO. (1999). La razón comunicada. Editorial Torres Asociados. México. Primera edición.

NEWMAN, JAMES R. (1997). El Mundo de las Matemáticas. Grijalbo.

POLYA, GEORGE (2002). Como plantear y resolver problemas. México: Trillas 1965 (reimp. 2002) traducción de: How to solve it. Pág. 16-22.

PAULE R., MARIA DEL PUERTO; SALAN, AITOR DE LA PUENTE; PEREZ PEREZ, JUAN RAMON Y GONZALES RODRIGUEZ, MARTIN. (2002). Una manera de adaptación a los estilos de aprendizaje teóricos y activos con preferencia alta y muy alta con una herramienta informática. Departamento de Informática de la Universidad de Oviedo.
Recuperado: 20 de diciembre de 2005
www.tecnoneet.org/docs/2002/3-6002.pdf

OJEDA ACIEGO, MANUEL. (2000). Lógica, Matemática, Deducción Automática. Departamento de matemática aplicada, Universidad de Málaga.
Recuperado: 30 de agosto de 2005
www.sat.uma.es/aciego/tr/gacet.pdf. Párrafo 1

RESNICK, LAUREN B. y KLOPFER, LEOPOLD E. (1989). Currículum y cognición. Aique grupo editor. Argentina. Tercera edición. Pagina 148

SALVAT, BEGOÑA GROS (1999) Psicología cognitiva e informática educativa. Universidad de Barcelona.

SANCHEZ, M. (1985). Teaching thinking processes. En D. N. Perkins, J. Lockhead J. C. Bishop (Eds), thinking: The second international conference pp. 413-430. Hillsdale, NJ: Lawrence Erlbaum Associates

SANCHEZ, M. (1992). Programa de desarrollo de habilidades de pensamiento. Revista intercontinental de psicología y educación 5 (2), págs 207-236.

SANTOS TRIGO, LUZ MANUEL (1997). Principios y métodos de la resolución de problemas en el aprendizaje de las matemáticas. Centro de Investigación y Estudios Avanzados del IPN. Grupo editorial Ibero América. Segunda Edición. Págs. 91-94

SOLIS, JUAN FRAUSTO Y SANCHEZ ANTE, GILDARDO. (2000). Fundamentos de lógica computacional. Editorial Trillas, México, Primera edición. Págs. 4, 33, 34, 44.

STERNBERG, R. (1985). Beyond I. Q. A triarchic theory of human intelligence. Cambridge University Press

SUSTRICK, MARTIN (2006). "Linguistics and Programming Languages" Newsletter: June 2006.
Recuperado: 30 de junio de 2006
www.ppig.org/newslettter

TUCKER, ALLEN Y NOONAN, ROBERT. (2003). Lenguajes de programación principios y paradigmas. McGraw-Hill. Madrid España. Pág. 4, 5

ULRICH TOBIAS, CYNTHIA. (2000) Como aprenden los niños. Editorial Vida, Miami Florida. Págs. 15-17.

WILLOGHBY, THEODORE C. (1980). Computer programmer aptitude battery: validation study.

# *11. ANEXOS*

**Anexo 1**

Profesor (a): _____

Puesto que usted ha mostrado voluntad para colaborar con nuestra investigación y que tiene las condiciones profesionales exigidas para emitir un criterio sobre el trabajo realizado, se necesitan algunos datos para procesar las opiniones que nos dará como parte de la aplicación del método de expertos. Debe guiarse por las indicaciones que se dan.

1. En la tabla aparece una escala que le permitirá expresar el nivel que usted considera tiene, para realizar un análisis relacionado con la enseñanza de la lógica de la programación. Marque con una “X” en la casilla que considere.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |

2. Si usted tuviera que argumentar sus criterios sobre un trabajo que trate el tema de la enseñanza de la lógica de la programación, tendría que apelar a sus conocimientos, intuición, experiencia, etc. Señale con una X en la siguiente tabla, la influencia que tienen estos elementos en los criterios que usted puede dar sobre el tema.

| Fuentes de argumentación | Alta | Media | Baja |
|---|---|---|---|
| Análisis teóricos realizados por usted. | | | |
| Experiencia como profesor. | | | |
| Trabajos consultados de autores nacionales. | | | |
| Trabajos consultados de autores extranjeros. | | | |
| Su propio conocimiento sobre el estado actual del problema en el extranjero. | | | |
| Su intuición. | | | |

**Anexo 2.**

| Fuentes de argumentación | Alta | Media | Baja |
|---|---|---|---|
| Análisis teóricos realizados por el sujeto. | 0,3 | 0,2 | 0,1 |
| Experiencia. | 0,5 | 0,4 | 0,2 |
| Trabajo de autores nacionales consultados. | 0,05 | 0,04 | 0,02 |
| Trabajo de autores extranjeros consultados. | 0,05 | 0,04 | 0,02 |
| Conocimiento sobre el estado actual del problema en el extranjero. | 0,05 | 0,04 | 0,02 |
| Intuición | 0,05 | 0,04 | 0,02 |

**Anexo 3.**

Coeficiente de los profesores de la población

| Sujeto | $k_c$ | Anál. t. | Exper. | Aut. nac. | Aut. ext. | Prob. ext. | Intuic. | $k_a$ | k |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.9 | 0.2 | 0.5 | 0.02 | 0.04 | 0.02 | 0.05 | 0.83 | 0.87 |
| 2 | 0.9 | 0.2 | 0.5 | 0.02 | 0.04 | 0.02 | 0.05 | 0.83 | 0.87 |
| 3 | 0.9 | 0.2 | 0.5 | 0.02 | 0.04 | 0.04 | 0.05 | 0.85 | 0.88 |
| 4 | 1 | 0.3 | 0.5 | 0.02 | 0.02 | 0.04 | 0.05 | 0.93 | 0.97 |
| 5 | 0.9 | 0.3 | 0.5 | 0.04 | 0.04 | 0.04 | 0.04 | 0.96 | 0.93 |
| 6 | 0.8 | 0.2 | 0.5 | 0.03 | 0.02 | 0.03 | 0.04 | 0.82 | 0.81 |
| 7 | 0.8 | 0.2 | 0.5 | 0.02 | 0.04 | 0.02 | 0.05 | 0.83 | 0.82 |
| 8 | 0.7 | 0.2 | 0.5 | 0.02 | 0.04 | 0.02 | 0.05 | 0.83 | 0.77 |
| 9 | 0.9 | 0.3 | 0.5 | 0.02 | 0.04 | 0.03 | 0.05 | 0.94 | 0.92 |
| 10 | 0.8 | 0.2 | 0.5 | 0.02 | 0.03 | 0.02 | 0.05 | 0.82 | 0.81 |
| 11 | 0.9 | 0.3 | 0.5 | 0.03 | 0.04 | 0.04 | 0.05 | 0.96 | 0.93 |

**Anexo 4.**

Compañero profesor(a) debido a su preparación y voluntad de cooperar con la investigación titulada “Alternativa didáctica para la enseñanza de la lógica para estudiantes de primero de bachillerato en computación”, se necesita que conteste el siguiente cuestionario, después de haber leído las

**Instrucciones:**

- En cada pregunta aparece una escala del 1 al 5, que se interpreta de la siguiente manera:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| Inadecuado | Poco adecuado | Adecuado | Bastante adecuado | Muy adecuado |

- Ud. debe marcar con una **X** el número correspondiente a su respuesta de acuerdo a esta escala.

Cuestionario.

1. La solución que se propone al problema de investigación es original, en el sentido de no guardar analogías con los aportes de otros investigadores.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | |

2. La investigación aporta un modelo que enriquece la teoría sobre la enseñanza de la lógica de la programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | |

3. La alternativa elaborada posee una estructura adecuada en correspondencia con el modelo teórico seguido y las exigencias prácticas de la enseñanza-aprendizaje de la lógica de la programación independientemente del lenguaje de programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | |

4. La alternativa didáctica elaborada tiene claridad en el contenido de cada elemento del modelo.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | |

5. La alternativa didáctica propuesta es de utilidad para potenciar el aprendizaje y desarrollo del pensamiento algorítmico asociado a los procesos de resolución de problemas con ayuda de la computadora, razonamiento y desarrollo de habilidades lógicas.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | |

Escriba a continuación que momentos considera que deben ser incluidos o eliminados en esta propuesta:

| Momento que se propone ser incluido | Momento que se propone ser eliminado |
|---|---|
| | |
| | |
| | |

Señale a continuación, si considera que el nombre de alguno de los momentos debe ser cambiado.

| Momento que se propone ser incluido | Momento que se propone ser eliminado |
|---|---|
| | |
| | |
| | |

**Anexo 5.**

Tabla 1

| Expertos | Categorías | | | | |
|---|---|---|---|---|---|
| | I1 | I2 | I3 | I4 | I5 |
| E1 | PA | MA | MA | MA | MA |
| E2 | MA | MA | BA | MA | MA |
| E3 | MA | MA | MA | MA | MA |
| E4 | A | BA | BA | BA | BA |
| E5 | A | MA | MA | BA | BA |
| E6 | MA | MA | MA | BA | MA |
| E7 | BA | BA | BA | BA | BA |
| E8 | A | MA | MA | MA | BA |
| E9 | BA | MA | MA | BA | BA |
| E10 | MA | MA | BA | BA | BA |
| E11 | BA | MA | MA | BA | MA |

**Anexo 6.**

Tabla 2

| Frecuencias absolutas de las evaluaciones por indicador | | | | | | |
|---|---|---|---|---|---|---|
| Aspectos | MA | BA | A | PA | I | Total |
| I1 | 4 | 3 | 3 | 1 | 0 | **11** |
| I2 | 9 | 2 | 0 | 0 | 0 | **11** |
| I3 | 7 | 4 | 0 | 0 | 0 | **11** |
| I4 | 4 | 7 | 0 | 0 | 0 | **11** |
| I5 | 5 | 6 | 0 | 0 | 0 | **11** |
| **Total** | **29** | **22** | **3** | **1** | **0** | **55** |

Tabla 3

| Frecuencias acumuladas de las evaluaciones por indicador | | | | | |
|---|---|---|---|---|---|
| Aspectos | MA | BA | A | PA | I |
| I1 | 3 | 10 | 11 | 11 | 11 |
| I2 | 3 | 9 | 11 | 11 | 11 |
| I3 | 8 | 11 | 11 | 11 | 11 |
| I4 | 4 | 9 | 10 | 10 | 10 |
| I5 | 3 | 9 | 11 | 11 | 11 |

Tabla 4

| Frecuencias acumuladas relativas de las evaluaciones por indicador | | | | | |
|---|---|---|---|---|---|
| **Aspectos** | **MA** | **BA** | **A** | **PA** | **I** |
| **I1** | 0.333 | 0.583 | 0.833 | 0.917 | 0.917 |
| **I2** | 0.750 | 0.917 | 0.917 | 0.917 | 0.917 |
| **I3** | 0.583 | 0.917 | 0.917 | 0.917 | 0.917 |
| **I4** | 0.333 | 0.917 | 0.917 | 0.917 | 0.917 |
| **I5** | 0.417 | 0.917 | 0.917 | 0.917 | 0.917 |

**Anexo 7.**

Tabla 5

| Cálculo de puntos de corte y escala de los indicadores | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Aspectos** | **MA** | **BA** | **A** | **PA** | Suma | Promedio | N-Prom. | |
| **I1** | -0.43 | 0.21 | 0.97 | 1.38 | 2.130117 | 0.532529 | 0.416 | **BA** |
| **I2** | 0.67 | 1.38 | 1.38 | 1.38 | 4.823472 | 1.205868 | -0.257 | **MA** |
| **I3** | 0.21 | 1.38 | 1.38 | 1.38 | 4.359411 | 1.089853 | -0.141 | **MA** |
| **I4** | -0.43 | 1.38 | 1.38 | 1.38 | 3.718255 | 0.929564 | 0.019 | **MA** |
| **I5** | -0.21 | 1.38 | 1.38 | 1.38 | 3.938554 | 0.984638 | -0.036 | **MA** |
| Suma | -0.18696485 | 5.74 | 6.499398 | 6.914971 | 18.96981 | 4.742452 | | |
| Promedio Puntos de corte | **-0.03739** | **1.14848** | **1.29988** | **1.382994** | 0.948490438 | | | |

**Anexo 8.**

Guía de entrevista.
Objetivo.

Acopiar opiniones de profesores de experiencia acerca de las dificultades que manifiestan los alumnos y alumnas en el aprendizaje de la Lógica de la Programación.

Preguntas a realizar.

1. ¿Qué dificultades más comunes manifiestan los alumnos en el aprendizaje de la Lógica de la Programación?
2. ¿Cuáles son las estructuras que mas dificultan el desarrollo del pensamiento algorítmico?
3. ¿Cuáles de estas dificultades están relacionadas con la selección del lenguaje de programación?
4. ¿Qué cambios cree usted que son necesarios en la metodología docente para mejorar la calidad del aprendizaje de los conocimientos básicos relacionados con la lógica de la programación y consecuentemente con el desarrollo del pensamiento algorítmico?

Encuesta

Profesor (a), con el objetivo de llevar adelante una investigación relacionada con la enseñanza aprendizaje de la lógica de la programación, que se ejecuta como tesis de maestría en Didáctica de la Matemática, se necesita su colaboración en esta encuesta. Recabamos de usted la mayor objetividad posible, insistiéndole en que la encuesta tiene carácter anónimo y que la información suministrada sólo se utilizará con fines científicos.

Cuestionario.

1. A continuación se exponen algunas de las dificultades que se presentan en el aprendizaje de la lógica de la programación, marque con una X las que manifiestan los alumnos a los cuales usted imparte clases.

   __ No comprenden el problema.

   __ No distinguen los valores que servirán como entrada en el algoritmo

   __ El algoritmo encontrado no es solución al problema planteado

   __ Dificultad en la comprensión de la compuerta lógica NOT

   __ Dificultad en la comprensión de la compuerta lógica AND

   __ Dificultad en la comprensión de la compuerta lógica OR

   __ Dificultad en las estructuras de decisiones a la hora de formular la expresión lógica

   __ Dificultad en la secuenciación lógica del algoritmo

   __ Dificultad en formular algoritmos finitos

   __ Dificultad en utilizar un metalenguaje para estructurar el algoritmo solución

2. A continuación se exponen algunas de las estrategias utilizadas en la enseñanza de la lógica de la programación, marque con una X, las que para usted, muestran mayor efectividad.

___ Desarrollo de los trabajos de clase o tareas en forma individual.

___ Desarrollo de los trabajos de clase o tareas en parejas.

___ Desarrollo de los trabajos de clase o tareas en grupo.

___ Desarrollando clases magistrales

___ Siguiendo la metodología de la resolución de problemas

___ Desarrollando guías de trabajos con varios problemas a la vez.

___ Desarrollando guías de trabajo con un problema a la vez.

___ Planteando problemas no rutinarios

___ Planteando problemas clásicos en el desarrollo de algoritmos

___ Aumentando el número de contenidos programáticos para la enseñanza de la lógica

**Anexo 9.**

Encuesta

Objetivo.

Acopiar información de profesores los profesores que imparten la clase de programación a alumnos novatos en institutos que tienen el bachillerato en computación.

Nombre del instituto

Barrio o Colonia

Tipo de Institución en la que trabaja

a. Oficial b. Privado c. Semiprivado

Años de servicio en docencia

Titulo obtenido en educación media

Nivel máximo de estudio completado

a. Educación Media b. Universidad

Titulo obtenido en universidad

Universidad en la que se graduó

Usa libro de texto

a. Si b. No Titulo texto:

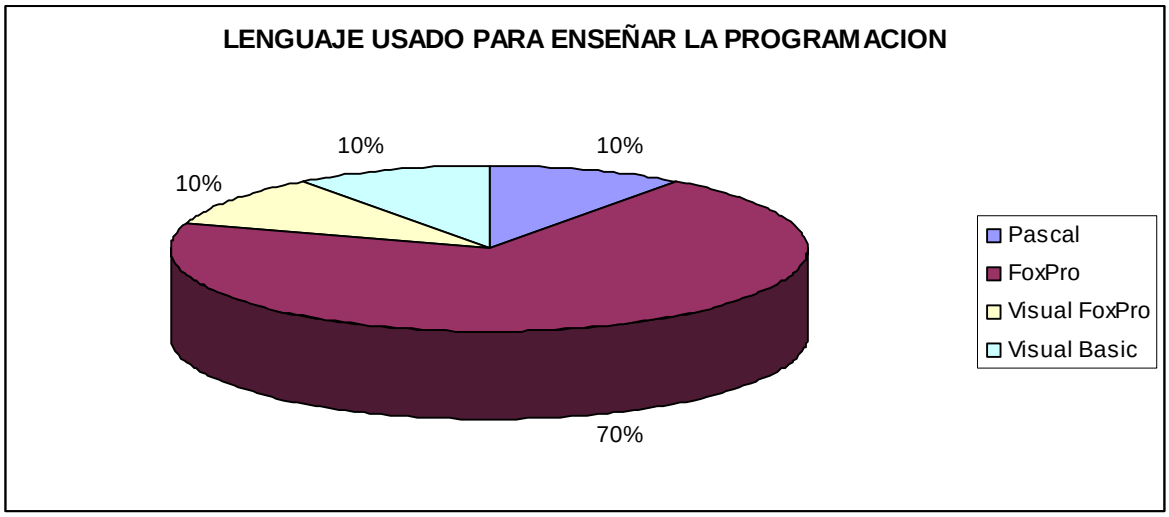
Porque utiliza el texto anterior:

Cuantos tiempo tiene de impartir clase de programación a primero de computación

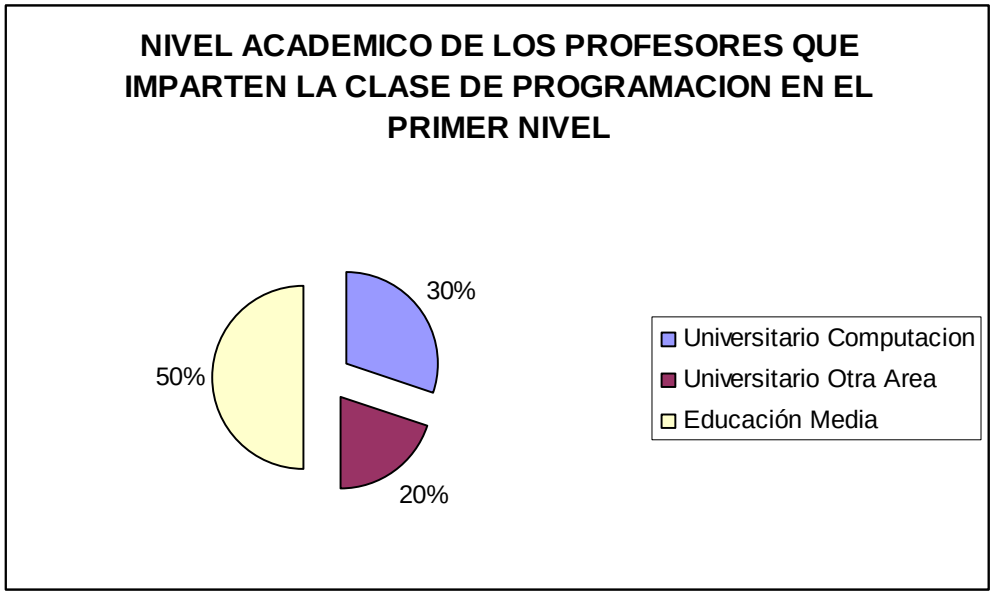
Cuales son las primeras tres unidades que desarrolla en la clase de programación

Lenguaje de programación que utiliza en la clase

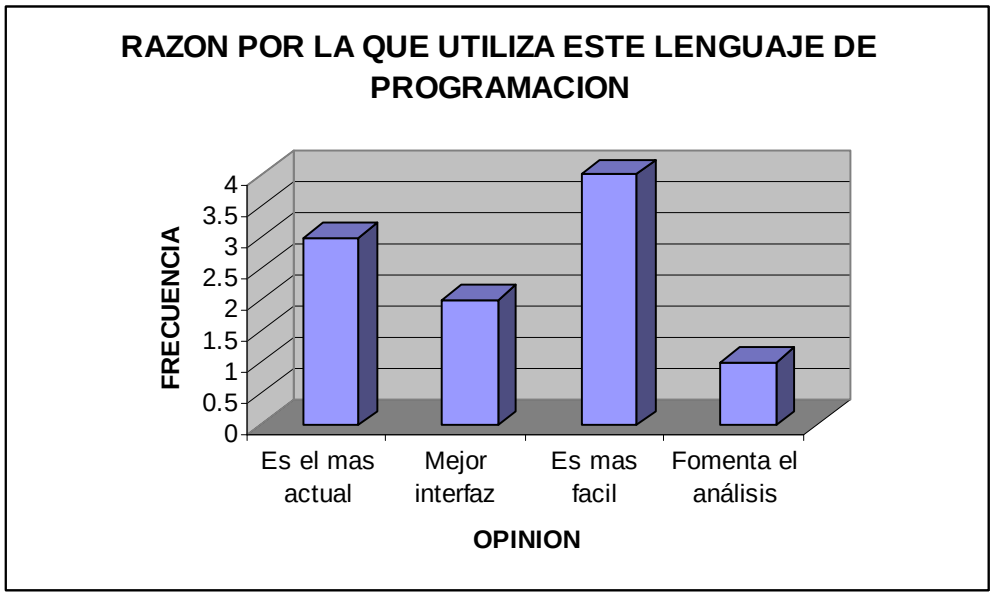
**Anexo 10.**



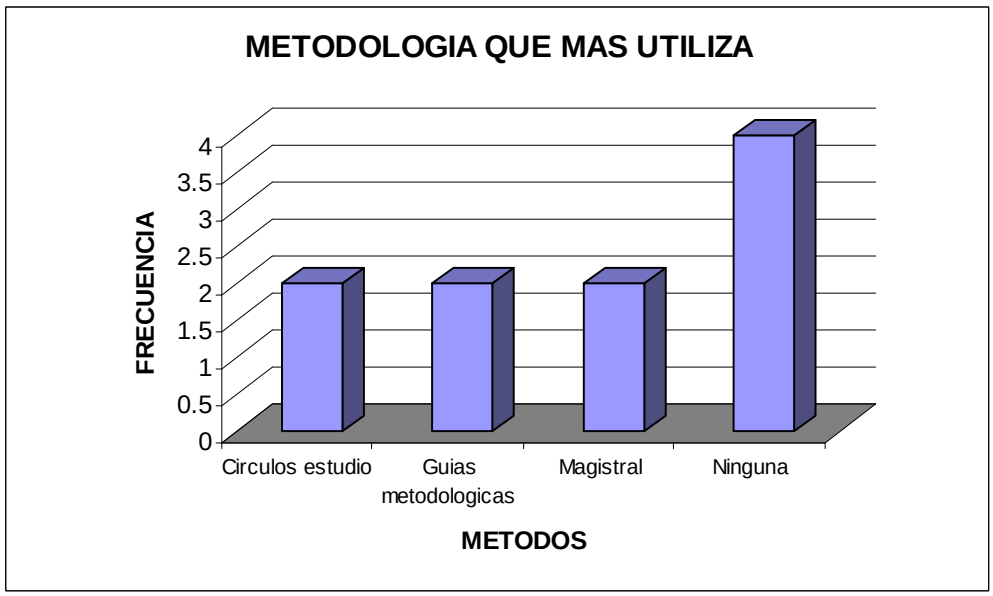

**Anexo 11.**



**Anexo 12.**




**Anexo 13.**



**Anexo 14.**

## EXAMEN DIAGNOSTICO

Instrucciones:

1. Escriba su nombre en la hoja de respuestas
2. responder las preguntas en la hoja de respuestas, escribiendo únicamente el número de la pregunta y la letra a) o b) o c) que corresponda a la respuesta que considere correcta.

**1.** Sustituir “m” por “p” en la palabra “mapa”. El resultado es:

a) papa

b) mama

c) pama

**2.** Sustituir “p” por “m” en la palabra “mapa”. El resultado es:

a) papa

b) mama

c) pama

**3.** Sustituir “p” en lugar de “m” en la palabra “mapa”. El resultado es:

a) mama

b) papa

c) pama

**4.** La negación lógica del enunciado “Si te portas bien entonces te llevo al cine” es:

a) Si no te portas bien entonces no te llevo al cine

b) Si te portas bien entonces no te llevo al cine

c) Te portas bien y no te llevo al cine

**5.** Sean A, B conjuntos y sea w un objeto tal que $w \notin A \cap B$, entonces:

a) $w \notin A$ y $w \notin B$

b) $w \notin A$ y ($w \in B$ ó $w \notin B$)

c) $w \notin A$ ó $w \notin B$

**6.** La negación lógica de “ser blanco” es:

a) ser negro

b) no ser blanco

c) ser de color distinto al blanco

**7.** La negación lógica de “3<x” es:

a) 3<x

b) $3 \geq x$

c) $3 \leq x$

**8.** La negación lógica de “Todos los perros ladran” es

a) Hay perros que no ladran

b) Todos los perros no ladran

c) Ningún perro ladra

**9.** Considere el siguiente argumento:

4 es número primo, si tiene exactamente dos divisores.

4 no tiene exactamente dos divisores ( tiene tres: 1, 2 y 4)

Por lo tanto, 4 no es primo.

a) El argumento es lógicamente correcto

b) El argumento es lógicamente incorrecto

c) El argumento no tiene sentido

**10.** Considere el siguiente argumento:

Juan vendrá, si hay un buen día.

No hay un buen día

Por lo tanto, Juan no vendrá.

a) El argumento es lógicamente correcto

b) El argumento es lógicamente incorrecto

c) El argumento no tiene sentido

**11.** Considere el siguiente argumento:

Cualquier barbero de San pedro sula, rasura a todos los hombres de San Pedro Sula que no se rasuran a si mismos y sólo a esos.

Por lo tanto, no hay barberos en San Pedro Sula.

a) El argumento es lógicamente correcto

b) El argumento es lógicamente incorrecto

c) El argumento no tiene sentido

**12.** Considere el siguiente argumento:

Todos los borogroves son kismis, si alguien tirila.

Nito tirila y Pac es un borogrove.

Por lo tanto, Pac es un kismi

a) El argumento es lógicamente correcto

b) El argumento es lógicamente incorrecto

c) El argumento no tiene sentido

**13.** Considere el siguiente argumento:

2 divide al numerador de 6/8

6/8 = 3/8

Por lo tanto, 2 divide al numerador de 3/4.

a) El argumento es lógicamente correcto

b) El argumento es lógicamente incorrecto

c) El argumento no tiene sentido

**14.** Considere el siguiente argumento:

Romeo ama a Julieta.

Julieta es una palabra de siete letras.

Por lo tanto, Romeo ama a una palabra de siete letras.

a) El argumento es lógicamente correcto

b) El argumento es lógicamente incorrecto

c) El argumento no tiene sentido

**15.** considere el siguiente argumento:

Todos le tienen miedo a Drácula

Drácula sólo le tiene miedo a William

Por lo tanto, William es Drácula

a) El argumento es lógicamente correcto

b) El argumento es lógicamente incorrecto

c) El argumento no tiene sentido

**16.** Considere el siguiente argumento:

Si hoy es martes, mañana será miércoles

Mañana será miércoles

Por lo tanto, hoy es martes.

a) El argumento es lógicamente correcto

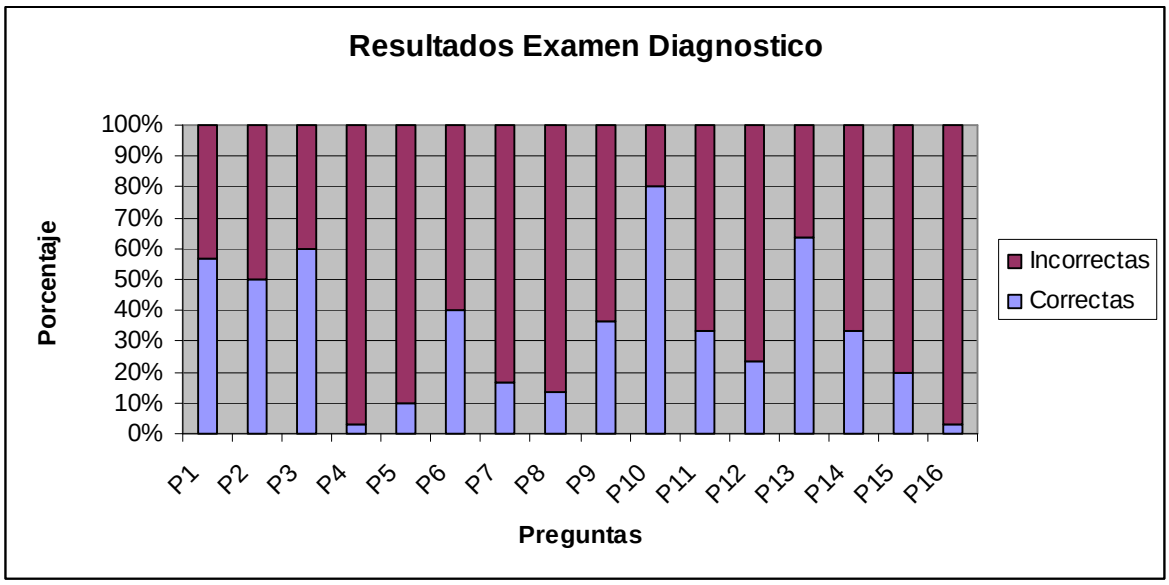
b) El argumento es lógicamente incorrecto

c) El argumento no tiene sentido

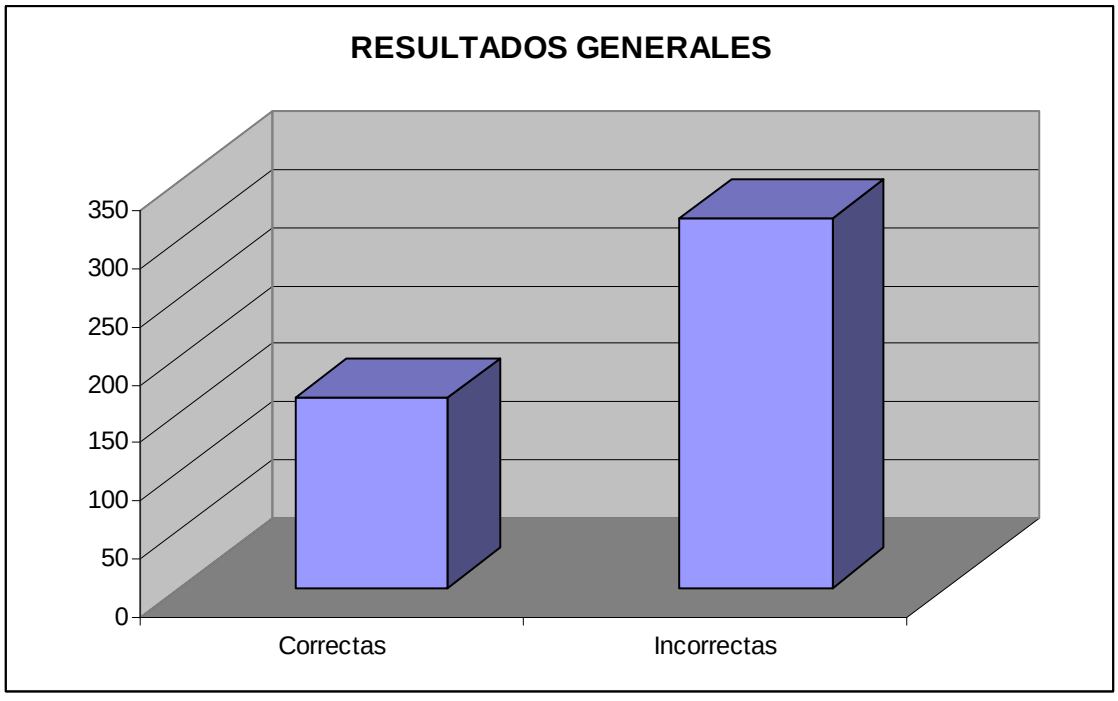
**Anexo 15.**

| No | EXAMEN DE DIAGNOSTICO | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | RESULTADOS POR PREGUNTA | | | | | | | | | | | | | | | |
| | A | B | B | C | C | B | B | A | A | A | B | A | B | A | A | B |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 1 | b | A | b | a | a | b | a | b | b | a | c | b | a | a | b | a |
| 2 | a | B | b | a | a | b | a | b | a | a | c | a | b | b | b | a |
| 3 | a | B | a | a | b | a | a | c | b | a | b | c | b | b | c | a |
| 4 | a | B | b | a | a | a | a | b | c | a | b | c | a | a | a | a |
| 5 | a | C | c | a | a | c | | a | a | c | b | c | a | a | c | a |
| 6 | b | A | b | a | a | b | a | c | c | a | b | c | b | a | b | a |
| 7 | a | | b | a | b | c | b | c | a | a | a | a | b | a | a | a |
| 8 | b | A | a | a | a | b | c | b | b | a | c | a | b | b | b | a |
| 9 | b | A | a | a | a | b | | b | b | a | c | c | | b | c | a |
| 10 | a | C | b | a | a | c | b | b | b | a | c | b | b | c | a | a |
| 11 | a | B | b | a | a | a | a | b | b | b | c | b | b | c | b | a |
| 12 | b | A | c | a | | b | a | b | b | a | b | c | b | b | b | b |
| 13 | a | B | a | a | a | a | a | c | c | a | b | c | b | a | b | a |
| 14 | b | A | b | b | a | c | a | b | a | a | a | c | b | c | c | a |
| 15 | a | B | a | a | | b | a | b | a | a | c | c | b | c | b | a |
| 16 | a | B | b | a | | c | b | a | a | a | c | a | | c | b | a |
| 17 | a | B | b | a | d | b | a | c | a | a | a | a | b | b | c | a |
| 18 | b | C | c | a | | c | b | b | c | a | b | c | a | b | b | a |
| 19 | a | B | c | a | a | c | a | c | b | a | c | c | b | a | a | a |
| 20 | b | A | b | a | c | a | a | c | b | a | c | a | b | b | a | a |
| 21 | a | B | b | a | | b | a | b | c | a | b | c | a | a | b | a |
| 22 | b | A | c | a | c | b | b | b | a | c | b | a | a | b | b | a |
| 23 | b | A | b | a | | a | a | b | b | a | c | c | b | b | c | a |
| 24 | b | A | b | a | b | c | a | a | c | b | b | c | a | b | b | a |
| 25 | a | B | a | a | | b | a | b | a | a | c | c | b | c | b | a |
| 26 | c | B | a | a | | a | a | | a | b | c | c | b | a | a | a |
| 27 | a | B | b | a | | a | a | a | b | a | c | c | b | b | b | c |
| 28 | a | B | b | a | a | b | a | b | a | a | c | c | a | a | b | a |
| 29 | a | B | b | a | a | a | a | b | b | b | c | b | b | c | b | a |
| 30 | b | A | b | c | c | a | a | b | b | a | c | c | a | b | b | a |

**Anexo 16.**




**Anexo 17.**



**Anexo 18.**

CUESTIONARIO HONEY-ALONSO DE ESTILOS DE APRENDIZAJE; CHAEA C.M.ALONSO,D.J.GALLEGO Y P.HONEY

1.- Tengo fama de decir lo que pienso claramente y sin rodeos.

2.- Estoy segura(o) de lo que es bueno y lo que es malo, lo que esta bien y lo que esta mal.

3.- Muchas veces actúo sin mirar las consecuencias.

4.- Normalmente trato de resolver los problemas metódicamente y paso a paso.

5.- Creo que los formalismos coartan y limitan la actuación libre de las personas.

6.- Me interesa saber cuales son los sistemas de valores de los demás y con que criterios actúan.

7.- Pienso que el actuar intuitivamente puede ser siempre tan valido como actuar reflexivamente.

8.- Creo que lo más importante es que las cosas funcionen.

9.- Procuro estar al tanto de lo que ocurre aquí y ahora.

10.- Disfruto cuando tengo tiempo para preparar mi trabajo y realizarlo a conciencia.

11.- Estoy a gusto siguiendo un orden, en las comidas, en el estudio, haciendo ejercicio regularmente.

12.- Cuando escucho una nueva idea enseguida comienzo a pensar como ponerla en práctica.

13.- Prefiero las ideas originales y novedosas aunque no sean prácticas.

14.- Admito y me ajusto a las normas solo si me sirven para lograr mis objetivos.

15.- Normalmente encajo bien con personas reflexivas, y me cuesta sintonizar con personas demasiado espontáneas, imprevisibles.

16.- Escucho con más frecuencia que hablo.

17.- Prefiero las cosas estructuradas a las desordenadas.

18.- Cuando poseo cualquier información, trato de interpretarla bien antes de manifestar alguna conclusión.

19.- Antes de hacer algo estudio con cuidado sus ventajas e inconvenientes.

20.- Crezco con el reto de hacer algo nuevo y diferente.

21.- Casi siempre procuro ser coherente con mis criterios y sistemas de valores. Tengo principios y los sigo.

22.- Cuando hay una discusión no me gusta ir con rodeos.

23.- Me disgusta implicarme afectivamente en mi ambiente de trabajo. Prefiero mantener relaciones distantes.

24.- Me gustan más las personas realistas y concretas que las teóricas.

25.- Me gusta ser creativa(o), romper estructuras.

26.- Me siento a gusto con personas espontáneas y divertidas.

27.-La mayoría de las veces expreso abiertamente cómo me siento.

28.- Me gusta analizar y dar vueltas a las cosas.

29.- Me molesta que la gente no se tome en serio las cosas.

30.-Me atrae experimentar y practicar las últimas técnicas y novedades.

31.-Soy cautelosa(o) a la hora de sacar conclusiones.

32.-Prefiero contar con el mayor número de fuentes de información. Cuantos más datos reúna para reflexionar, mejor.

33.-Tiendo a ser perfeccionista.

34.-Prefiero oír las opiniones de los demás antes de exponer la mía.

35.-Me gusta afrontar la vida espontáneamente y no tener que planificar todo previamente.

36.-En las discusiones me gusta observar cómo actúan los demás participantes.

37.-Me siento incómoda(o) con las personas calladas y demasiado analíticas.

38.-Juzgo con frecuencia las ideas de los demás por su valor práctico.

39.-Me agobio si me obligan a acelerar mucho el trabajo para cumplir un plazo.

40.-En las reuniones apoyo las ideas prácticas y realistas.

41.-Es mejor gozar del momento presente que deleitarse pensando en el pasado o en el futuro.

42.-Me molestan las personas que siempre desean apresurar las cosas.

43.-Aporto ideas nuevas y espontáneas en los grupos de discusión.

44.-Pienso que son más conscientes las decisiones fundamentadas en un minucioso análisis que las basadas en la intuición.

45.-Detecto frecuentemente la inconsistencia y puntos débiles en las argumentaciones de los demás.

46.-Creo que es preciso saltarse las normas muchas más veces que cumplirlas.

47.-A menudo caigo en cuenta de otras formas mejores y más prácticas de hacer las cosas.

48.-En conjunto hablo más que escucho.

49.-Prefiero distanciarme de los hechos y observarlos desde otras perspectivas.

50.-Estoy convencida(o) que deber imponerse la lógica y el razonamiento.

51.-Me gusta buscar nuevas experiencias.

52.-Me gusta experimentar y aplicar las cosas.

53.-Pienso que debemos llegar pronto al grano, al meollo de los temas.

54.-Siempre trato de conseguir conclusiones e ideas claras.

55.-Prefiero discutir cuestiones concretas y no perder el tiempo con charlas vacías.

56.-Me impaciento cuando me dan explicaciones irrelevantes e incoherentes.

57.-Compruebo antes si las cosas funcionan realmente.

58.-Hago varios borradores antes de la redacción definitiva de un trabajo.

59.-Soy consciente de que en las discusiones ayudo a mantener a los demás centrados en el tema, evitando divagaciones.

60.-Observo que, con frecuencia, soy una(o) de las(los) más objetivas(os) y desapasionados en las discusiones.

61.- Cuando algo va mal le quito importancia y trato de hacerlo mejor.

62.- Rechazo ideas originales y espontáneas si no las veo prácticas.

63.- Me gusta sopesar diversas alternativas antes de tomar una decisión.

64.- Con frecuencia miro hacia delante para prever el futuro.

65.- En los debates y discusiones prefiero desempeñar un papel secundario antes que ser el/la líder o el/la que más participa.

66.- Me molestan las personas que no actúan con lógica.

67.- Me resulta incomodo tener que planificar y prever las cosas.

68.- Creo que el fin justifica los medios en muchos casos.

69.- Suelo reflexionar sobre los asuntos y problemas.

70.- El trabajar a conciencia me llena de satisfacción y orgullo.

71.- Ante los acontecimientos trato de descubrir los principios y teorías en que se basan.

72.- Con tal de conseguir el objetivo que pretendo soy capaz de herir sentimientos ajenos.

73.- No me importa hacer todo lo necesario para que sea efectivo  mi trabajo.

74.- Con frecuencia soy una de las personas que más anima las fiestas.

75.- Me aburro enseguida con el trabajo metódico y minucioso.

76.- La gente con frecuencia cree que soy poco sensible a sus sentimientos.

77.- Suelo dejarme llevar por mis intuiciones.

78.- Si trabajo en grupo procuro que se siga un método y un orden.

79.- Con frecuencia me interesa averiguar lo que piensa la gente.

80.- Esquivo los temas subjetivos, ambiguos y poco claros.

## *PERFIL DE APRENDIZAJE*

1.- Rodee con una línea cada uno de los números que ha señalado con un signo más (+)

2.- Sume el número de círculos que hay en cada columna.

3.- Coloque estos totales en la gráfica. Así comprobará cual es su estilo o estilos de aprendizaje preferentes.

| I | II | III | IV |
|---|---|---|---|
| 3 | 10 | 2 | 1 |
| 5 | 16 | 4 | 8 |
| 7 | 18 | 6 | 12 |
| 9 | 19 | 11 | 14 |
| 13 | 28 | 15 | 22 |
| 20 | 31 | 17 | 24 |
| 26 | 32 | 21 | 30 |
| 27 | 34 | 23 | 38 |
| 35 | 36 | 25 | 40 |
| 37 | 39 | 29 | 47 |
| 41 | 42 | 33 | 52 |
| 43 | 44 | 45 | 53 |
| 46 | 49 | 50 | 56 |
| 48 | 55 | 54 | 57 |
| 51 | 58 | 60 | 59 |
| 61 | 63 | 64 | 62 |
| 67 | 65 | 66 | 68 |
| 74 | 69 | 71 | 72 |
| 75 | 70 | 78 | 73 |
| 77 | 79 | 80 | 76 |

Totales:

| Grupo | Activo | Reflexivo | Teórico | Pragmático |
|---|---|---|---|---|

**Anexo 19.**

| No | DIAGNOSTICO DE LOS ESTILOS DE APRENDIZAJE | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | ACTIVO | | REFLEXIVO | | TEORICO | | PRAGMATICO | |
| | MAS | MENOS | MAS | MENOS | MAS | MENOS | MAS | MENOS |
| 1 | 12 | 8 | 16 | 4 | 15 | 5 | 17 | 3 |
| 2 | 11 | 9 | 16 | 4 | 14 | 6 | 14 | 6 |
| 3 | 8 | 12 | 15 | 5 | 12 | 8 | 14 | 6 |
| 4 | 9 | 11 | 15 | 5 | 9 | 11 | 9 | 11 |
| 5 | 9 | 11 | 19 | 1 | 12 | 8 | 11 | 9 |
| 6 | 8 | 12 | 17 | 3 | 14 | 6 | 10 | 10 |
| 7 | 10 | 9 | 12 | 8 | 13 | 7 | 11 | 8 |
| 8 | 10 | 10 | 16 | 4 | 14 | 6 | 13 | 7 |
| 9 | 12 | 8 | 16 | 4 | 13 | 7 | 14 | 6 |
| 10 | 13 | 7 | 18 | 2 | 18 | 2 | 17 | 3 |
| 11 | 18 | 2 | 14 | 4 | 8 | 12 | 17 | 3 |
| 12 | 14 | 6 | 16 | 4 | 16 | 4 | 17 | 3 |
| 13 | 15 | 5 | 18 | 2 | 17 | 3 | 17 | 3 |
| 14 | 9 | 11 | 17 | 3 | 15 | 5 | 16 | 3 |
| 15 | 11 | 9 | 19 | 1 | 19 | 1 | 18 | 2 |
| 16 | 11 | 5 | 13 | 2 | 11 | 3 | 11 | 5 |
| 17 | 9 | 11 | 18 | 2 | 13 | 7 | 10 | 10 |
| 18 | 9 | 11 | 12 | 8 | 11 | 9 | 10 | 10 |
| 19 | 8 | 12 | 13 | 7 | 10 | 10 | 10 | 10 |
| 20 | 8 | 12 | 20 | 0 | 18 | 2 | 14 | 6 |
| 21 | 14 | 5 | 17 | 3 | 14 | 6 | 15 | 5 |
| 22 | 13 | 7 | 13 | 7 | 14 | 6 | 17 | 3 |
| 23 | 13 | 7 | 14 | 6 | 13 | 7 | 12 | 8 |
| 24 | 7 | 13 | 14 | 6 | 9 | 11 | 11 | 9 |
| 25 | 13 | 7 | 19 | 1 | 15 | 5 | 15 | 5 |
| 26 | 14 | 6 | 16 | 4 | 13 | 7 | 9 | 11 |
| 27 | 9 | 11 | 15 | 5 | 12 | 8 | 12 | 8 |
| 28 | 8 | 12 | 11 | 9 | 15 | 4 | 10 | 10 |

**Anexo 20.**

| FUNCIONES DOCENTES | FUNCIONES DICENTES |
|---|---|
| 1. Guía<br>2. Consultor | 1.Resolutor<br>2. Evaluador<br>3. Estimador |

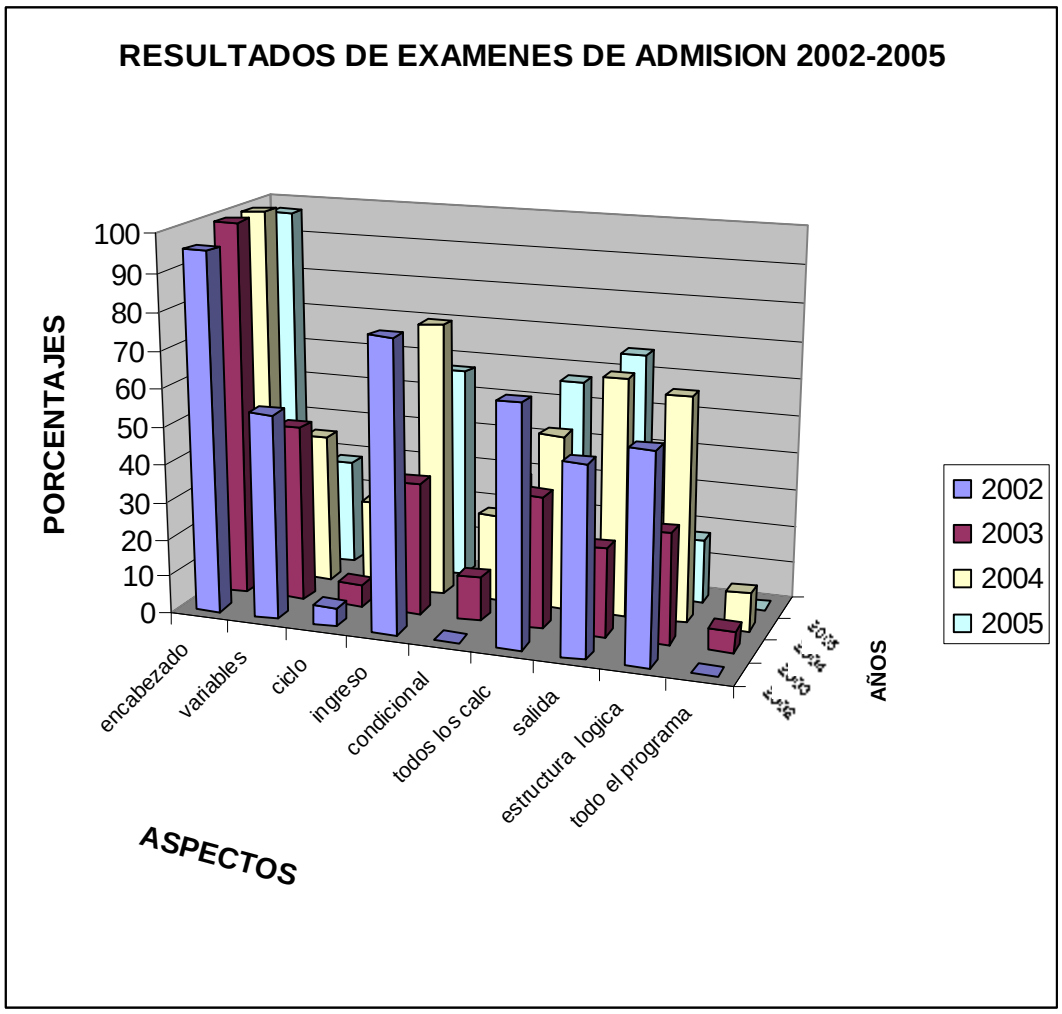
**Anexo 21.**

CUADRO RESUMEN DE EXAMENES APLICADOS A ESTUDIANTES DE PRIMER INGRESO

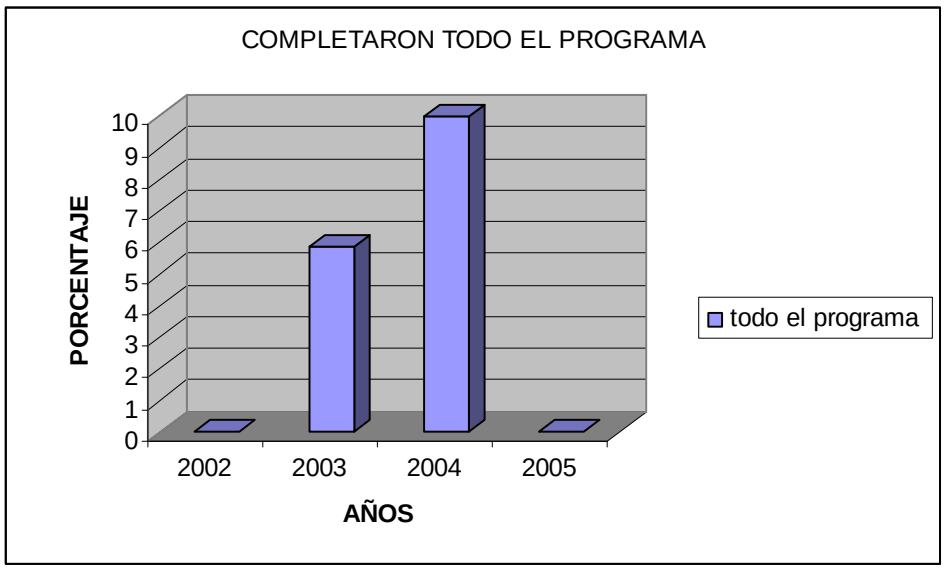
PERIODO 2002-2005

| AÑO | encabezado | variables | ciclo | ingreso | proceso | | salida | estructura lógica | todo el programa |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | condicional | Todos los Calc. | | | |
| **2002** | 95 | 55 | 5 | 77 | 0 | 64 | 50 | 55 | 0 |
| **2003** | 100 | 47 | 6 | 35 | 12 | 35 | 24 | 29 | 6 |
| **2004** | 100 | 40 | 23 | 73 | 23 | 47 | 63 | 60 | 10 |
| **2005** | 97 | 29 | 17 | 57 | 29 | 57 | 66 | 17 | 0 |

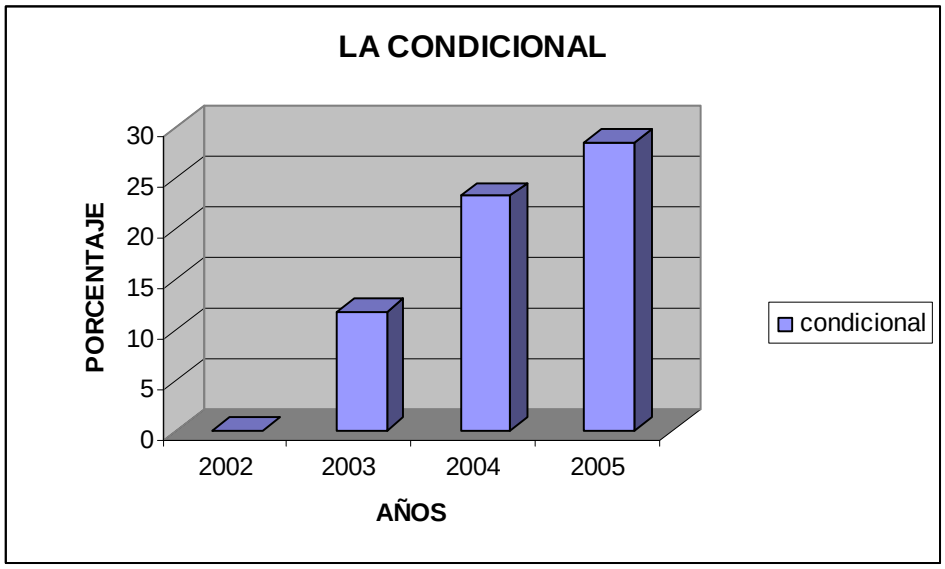
**Anexo 22.**




**Anexo 23**



**Anexo 24.**


LA CONDICIONAL
PORCENTAJE
30
25
20
15
10
5
0
2002
2003
2004
2005
AÑOS
condicional

## RESULTADOS EXAMEN 2002

| | Nombre alumno | encabezado | variables | ciclo | ingreso | proceso | | salida | estructura logica | todo el programa |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | condicional | todos los calc | | | |
| 1 | enil manases rodriguez | b | m | | m | m | b | b | b | m |
| 2 | karen barreda recarte | b | m | | b | m | m | b | m | m |
| 3 | luis castillo sarmiento | b | b | | b | m | b | b | b | m |
| 4 | karen paz hernandez | b | b | | b | m | m | m | m | m |
| 5 | Allan euceda garza | b | m | | b | m | b | b | m | m |
| 6 | Kelvin Chavez | b | m | | b | m | b | b | m | m |
| 7 | Wendy fuentes | b | b | | b | m | b | m | b | m |
| 8 | Alexis enamorado | b | m | | m | m | m | m | m | m |
| 9 | Yuliana quintanilla tejada | b | m | | m | m | m | m | m | m |
| 10 | Victor manuel vasquez | b | m | | b | m | m | m | b | m |
| 11 | Rebeca ayala meza | b | b | | b | m | b | m | b | m |
| 12 | Bessy cristina barahona | b | b | | b | m | b | b | b | m |
| 13 | Julio Quintanilla | b | b | | b | m | b | m | b | m |
| 14 | Roberto castillo | b | b | | b | m | b | b | m | m |
| 15 | ilza parrilla tobias | b | b | | b | m | b | m | b | m |
| 16 | Yohana Padilla | b | b | | b | m | b | b | b | m |
| 17 | Johana hernandez | m | m | | b | m | m | m | m | m |
| 18 | Jorge Urbina | b | m | | b | m | b | b | b | m |
| 19 | Luis rodriguez diaz | b | b | | b | m | m | b | m | m |
| 20 | Tania reyes aguilar | b | m | | m | m | m | m | m | m |
| 21 | Grisel torres trochez | b | b | | m | b | b | b | b | m |
| 22 | zoila euceda hernandez | b | b | | b | m | b | m | b | m |
| | Numero Buenos | 21 | 12 | 0 | 17 | 1 | 14 | 11 | 12 | 0 |
| | Porcentaje | 95 | 55 | 0 | 77 | 5 | 64 | 50 | 55 | 0 |

**Anexo 26.**

## RESULTADOS EXAMEN 2003

| | Nombre alumno | encabezado | variables | ciclo | ingreso | proceso condicional | proceso todos los calc | salida | estructura logica | todo el programa |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Saul javier vasquez | B | b | m | b | m | b | b | b | m |
| 2 | josian flores | B | b | m | b | m | b | b | b | m |
| 3 | Diana madrid hernandez | B | b | m | m | m | m | m | b | m |
| 4 | yury Gracivel madrid | B | b | m | m | m | m | b | m | m |
| 5 | Rony padilla alvarado | B | b | m | m | m | b | m | m | m |
| 6 | Bernadette Quintanilla | B | m | m | m | m | m | m | m | m |
| 7 | Nolvia arias Blanco | B | b | b | b | b | b | b | b | b |
| 8 | Irma arias blanco | B | b | m | b | b | b | m | b | m |
| 9 | Lesbia myleibis mendez | B | m | m | m | m | m | m | m | m |
| 10 | angel teruel perez | B | m | m | m | m | m | m | m | m |
| 11 | dennis geovanny moncada | B | m | m | m | m | m | m | m | m |
| 12 | darlin norely coto | B | m | m | m | m | m | m | m | m |
| 13 | Manuel alejandro bardales | B | m | m | m | m | m | m | m | m |
| 14 | gabriela elizabeth zaldivar | B | m | m | m | m | m | m | m | m |
| 15 | Wilson oswaldo calles | B | m | m | b | m | m | m | m | m |
| 16 | Ericka johana ramirez | B | m | m | m | m | b | m | m | m |
| 17 | Lesby faricio castro | B | b | m | b | m | m | m | m | m |
| | Numero de buenos | 17 | 8 | 1 | 6 | 2 | 6 | 4 | 5 | 1 |
| | PORCENTAJE BUENO | 100 | 47 | 6 | 35 | 12 | 35 | 24 | 29 | 6 |

## RESULTADOS EXAMEN 2004

| | Nombre alumno | encabezado | variables | ciclo | ingreso | proceso condicional | proceso todos los calc | salida | estructura logica | todo el programa |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Joasiel armijo | B | b | m | b | m | b | b | b | m |
| 2 | Carlos Roberto lopez escobar | B | m | m | m | m | m | m | m | m |
| 3 | Javier rapalo | B | m | b | b | m | b | b | b | m |
| 4 | Belkis Leiva | B | m | m | b | m | m | m | b | m |
| 5 | nataly bacila | B | b | b | b | b | b | b | b | b |
| 6 | Edgar Nuñez | B | m | b | b | b | b | b | b | m |
| 7 | Miguel angel cruz | B | b | b | b | b | b | b | b | b |
| 8 | Paola Melitina Alvarez | B | b | m | b | m | b | b | b | m |
| 9 | Mario Lainez Hernandez | B | m | m | m | m | b | b | b | m |
| 10 | Mario Rubio | B | b | b | b | b | b | b | b | b |
| 11 | Lorvin ramos perez | B | b | m | b | m | m | m | m | m |
| 12 | eva melissa castro | B | m | m | b | m | m | m | m | m |
| 13 | Orman Javier peña | B | m | m | b | m | m | m | b | m |
| 14 | Christopher zuniga | B | b | m | b | m | b | b | b | m |
| 15 | Ruth ramona barrera | B | m | m | b | b | b | b | b | m |
| 16 | Yerlyng Castellanos | B | m | m | b | b | m | b | b | m |
| 17 | Paola Gomez Borjas | B | b | m | b | b | b | b | b | m |
| 18 | Mirta Idalia gutierrez | B | m | m | b | m | m | m | m | m |
| 19 | Erick Josue Sanchez | B | m | m | m | m | m | m | m | m |
| 20 | Nancy nolasco Coello | B | m | m | m | m | m | b | b | m |
| 21 | jenny castro hernandez | B | m | b | b | m | m | b | b | m |
| 22 | dina marleni rege | B | m | m | m | m | m | m | m | m |
| 23 | christian ely garcia | B | m | m | m | m | m | m | m | m |
| 24 | marlon cantillano ferrera | B | b | m | m | m | m | b | m | m |
| 25 | fanny anabel Amaya | B | b | m | b | m | m | b | b | m |
| 26 | Manuel osorto | B | m | m | m | m | m | m | M | m |
| 27 | Fredy omar fuentes | B | b | m | b | m | m | b | M | m |
| 28 | Heidy Karina | B | b | b | b | m | b | b | B | m |
| 29 | Mirna baide Quintanilla | B | m | m | b | m | b | b | M | m |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 30 | Ofelia mejia enamorado | B | m | m | b | m | b | m | M | m |
| | Total buenos | 30 | 12 | 7 | 22 | 7 | 14 | 19 | 18 | 3 |
| | Prcentaje | 100 | 40 | 23 | 73 | 23 | 47 | 63 | 60 | 10 |

**Anexo 28.**

## RESULTADOS EXAMEN 2005

| | Nombre alumno | encabezado | variables | ciclo | ingreso | proceso condicional | proceso todos los calc | salida | estructura logica | todo el programa |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Mario Perdomo | b | m | m | m | m | m | m | m | M |
| 2 | Vanessa Hernandez castro | b | m | m | m | m | m | m | m | M |
| 3 | Estela lily coto | b | m | m | b | m | m | m | m | M |
| 4 | Dilcia caballero funez | b | m | m | m | m | m | m | m | M |
| 5 | allan caballero morales | b | m | m | m | m | m | m | m | M |
| 6 | Mario Roberto roque | b | m | b | m | m | b | b | m | M |
| 7 | elmer castro gomez | b | b | m | m | m | m | m | m | M |
| 8 | mario roberto izaguirre | b | b | m | m | m | m | b | m | M |
| 9 | Ernesto salazar flores | b | m | m | b | m | m | m | m | M |
| 10 | ingrid merlos antunez | b | m | m | m | m | m | m | m | M |
| 11 | christian teruel cruz | b | b | m | b | m | b | b | b | M |
| 12 | Claudio aldair argueta | b | b | m | m | b | b | b | m | M |
| 13 | carlos corea | b | m | m | b | m | b | b | m | M |
| 14 | Angela lucia martinez | b | b | m | b | m | b | b | b | M |
| 15 | Nohelia del rosario bustillo | b | b | m | b | b | b | b | m | M |
| 16 | Kory melissa salinas | b | m | b | b | m | b | m | m | M |
| 17 | griselda jackeline sanchez | b | m | m | b | m | m | m | m | M |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 18 | manuel alejandro interiano | b | m | m | b | m | m | m | m | M |
| 19 | Adriana ramirez gallegos | b | m | m | m | m | m | m | m | M |
| 20 | Sintia Yanarith Amaya | b | m | m | b | m | m | b | m | M |
| 21 | sergio castillo cabrera | b | m | b | m | b | b | b | m | M |
| 22 | Juan martinez | b | m | m | b | m | b | b | m | M |
| 23 | Francisco Bladimir espinal | b | b | m | b | b | b | b | m | M |
| 24 | Richel dayana martinez | b | m | b | m | m | b | b | m | M |
| 25 | Vicente flores Villena | b | m | m | b | m | b | b | m | M |
| 26 | Edwin jafet lopez | m | m | m | b | b | b | b | m | M |
| 27 | Jose wilmer flores | b | b | m | b | b | m | b | m | M |
| 28 | Lenny caballero | b | m | m | m | b | b | b | m | M |
| 29 | Viany julissa rivera | b | m | m | b | b | b | b | b | M |
| 30 | Merlin pineda argueto | b | m | b | b | b | b | b | b | M |
| 31 | Douglas Roque cueva | b | m | m | b | b | b | b | m | M |
| 32 | oscar rene lopez | b | m | m | m | m | m | b | m | M |
| 33 | kenia isabel Amaya | b | m | m | b | m | b | b | b | M |
| 34 | oscar fuentes calidonio | b | b | m | m | m | b | b | m | M |
| 35 | esther aribel paz | b | b | b | b | m | b | b | b | M |
| | | | | | | | | | | |
| | Numero buenos | 34 | 10 | 6 | 20 | 10 | 20 | 23 | 6 | 0 |
| | Porcentaje buenos | 97 | 29 | 17 | 57 | 29 | 57 | 66 | 17 | 0 |

**Anexo 29.**

**DIAGNOSTICO PARA DOCENTES DEL AREA COMPUTACIONAL DE INSTITUTOS DE SAN PEDROS SULA**

1 Oficial
2 Semi Oficial
3 Privado

1 Media
2 Universitario
3 Maestria

| No | INSTITUTO | TIPO | TITULO EDUC. MEDIA | NIVEL ACADEMICO | AÑOS | TIEMPO IMPARTIR | LENGUAJE USADO | RAZON POR LA QUE LO UTILIZA | METODOLOGIA |
|---|---|---|---|---|---|---|---|---|---|
| **1** | Jose Trinidad Reyes | 1 | Bach. CCLL | 2 | 1.5 | 1.5 | Pascal | Diseñado en plan basico de calses | ninguna |
| **2** | Valle Verde | 3 | Bach. Computo | 1 | 3 | 3 | FoxPro | El mas comun en las empresas | Teoria Practica |
| **3** | Tecnico Sta Martha | 3 | Bach. Computo | 1 | 1 | 0.5 | FoxPro | es facil | diagramas |
| **4** | Tecnologico del Norte | 3 | Psicologia educativa | 2 | 8 | 8 | Visual Estudio FoxPro | Rapido, sencillo Facil | Catedra libre |
| **5** | Saint Mary Land | 3 | Bach. Computo | 1 | 5 | 5 | FoxPro | fomenta el analisis al desarrollar programas | circulos de estudio |
| **6** | Privado Rio Blanco | 3 | Bach. Computo | 1 | 2 | 2 | FoxPro | es de mas facil comprension | ninguna |
| **7** | Tecnologico Hondureño | 3 | Bach. Computo | 1 | 8 | 8 | V. Basic, Fox | Es actualmente moderno | Practica, Expositiva |
| **8** | Tecnico Morazan | 3 | Informatica educativa | 2 | 4 | 4 | FoxPro | Para formatear es lo que esta en el mercado | Guias metodologicas |
| **9** | Renovacion | 3 | Pasante Informatica Educativa | 2 | 12 | 12 | FoxPro | Mas interactivo, mejor interfaz, lo que esta en el mercado | Guias metodologicas |
| **10** | Ramon Amaya Amador | 3 | Informatica y Programación | 2 | 15 | 15 | FoxPro | es el mejor para explicar y para que entiendan | Circulos de aprendizaje |

| PRIMERAS UNIDADES QUE DESARROLLA | | | METODOS LOGICOS QUE CONOCE |
|---|---|---|---|
| Principios logicos programacion | Estructuras turbo pascal | ciclos en turbo pascal | Algoritmos |
| Base de datos | Comandos Foxpro | Desarrollo Problemas | Desarrollo y entendimiento programas |
| conceptos basicos | Pasos para crear un programa | comados en foxpro | operadores |
| Introduccion | Base de datos | | No sabe |
| Conocer los comandos | creacion de bases de datos | desarrollo de 4 programas basicos | java log |
| lenguaje de promocion basico | comando y como funcionan | | No sabe |
| Introduccion | Sentencias | Proyectos | ciclos, condiciones |
| Diagramas | Visual Foxpro | ciclos | Hojas de diseño, exposiciones proyectos |
| Concepto | Base de datos | Diagramacion | Diagramas, pseudocodigo, Prueba escritorio |
| Introduccion | Objetivos | Proyectos | Ciclos, Visual Foxpro |

| METODOS LOGICOS EN LA PROGRAMACION | ESTRATEGIA QUE MEJOR LE FUNCIONA | QUE ES MAS IMPORTANTE | TEXTO QUE UTILIZA | OTRO | MEJOR |
|---|---|---|---|---|---|
| Algoritmo | Desarrollar la logica personal de cada alumno | 2 | Turbo pascal | 2 | 1 |
| que aprendan a traducir cada linea del programa | que el alumno aprenda a entender cada linea | 1 | Visual FoxPro | 1 | 1 |
| Operadores | la practica | 2 | Visual FoxPro | 2 | 1 |
| No sabe | Trabajando con una maquina y analizando un esuema | 2 | Visual FoxPro | 1 | 1 |
| flujogramas, hojas de diseño, desarrollo de formulas y condiciones | circulos de estudio, educ autogestionaria | 2 | ninguno | 1 | 2 |
| ejemplos, que ellos solos trabajen | que trabajen solos | ambos | Visual FoxPro | 2 | 1 |
| | la practica | 1 | Ninguno | 1 | 1 |
| Ejercicios practicos, exposicion de proyectos | Practica | 2 | No especifico | 1 | 2 |
| los anteriores | Practica, interes por el alumno | 2 | No especifico | 1 | 1 |
| los anteriores | la practica y explicar bien | 2 | FoxPro Avanzado | 1 | 1 |

**Anexo 30.**

| INSTITUTOS DE SAN PEDRO SULA QUE POSEEN LA CARRERA DE BACHILLERATO EN COMPUTACION | | |
|---|---|---|
| 1 | ACASULA CARMEN CASTRO | Bo. Guamilito |
| 2 | CENTRO CULTURAL SANPEDRANO | Bo. Guamilito |
| 3 | CENTRO EDUCATIVO EN COMPUTACION | Bo. Guamilito |
| 4 | DON BOSCO | Col. Villa Florencia |
| 5 | ENSEÑANZA COMPUTARIZADA | Bo. Guamilito |
| 6 | INSTITUTO RENOVACION | Bo. Guamilito |
| 7 | INSTITUTO TECNICO EN COMPUTACION | Bo. Guamilito |
| 8 | INTERNACIONAL TECNOLOGICO | Bo. Guamilito |
| 9 | JOSE TRINIDAD REYES | Col. San Jose |
| 10 | LICEO MORAZANICO | Bo. Guamilito |
| 11 | LICEO SANPEDRANO | Bo. Guamilito |
| 12 | LICEO VICTORIA | Centro |
| 13 | MARIA AUXILIADORA | Bo. Guamilito |
| 14 | POLITECNICO DEL NORTE | Las Brisas |
| 15 | TECNOLOGICO EN CIENCIAS DE LA COMPUTACION | m m |
| 16 | TECNOLOGICO SANTO TOMAS | Palenque |
| 17 | MANUEL PAGAN LOZANO | Col. Lopez |
| 18 | PRIMERO DE MAYO DE 1954 | Col. Fesitranh |
| 19 | VALLE VERDE | Col. Santa Marta |

Profesor (a): MARCO ANTONIO DÍAZ

Email: marco.diaz@usps.edu

Puesto que usted ha mostrado voluntad para colaborar con nuestra investigación y que tiene las condiciones profesionales exigidas para emitir un criterio sobre el trabajo realizado, se necesitan algunos datos para procesar las opiniones que nos dará como parte de la aplicación del método de expertos. Debe guiarse por las indicaciones que se dan.

1. En la tabla aparece una escala que le permitirá expresar el nivel que usted considera tiene, para realizar un análisis relacionado con la enseñanza de la lógica de la programación. Marque con una “X” en la casilla que considere.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | X | |

2. Si usted tuviera que argumentar sus criterios sobre un trabajo que trate el tema de la enseñanza de la lógica de la programación, tendría que apelar a sus conocimientos, intuición, experiencia, etc. Señale con una X en la siguiente tabla, la influencia que tienen estos elementos en los criterios que usted puede dar sobre el tema.

| Fuentes de argumentación | Alta | Media | Baja |
|---|---|---|---|
| Análisis teóricos realizados por usted. | | X | |
| Experiencia como profesor. | X | | |
| Trabajos consultados de autores nacionales. | | | X |
| Trabajos consultados de autores extranjeros. | | X | |
| Su propio conocimiento sobre el estado actual del problema en el extranjero. | | X | |
| Su intuición. | X | | |

Profesor (a): ADOLFO DI MARE

Email: adolfo@di-mare.com

Puesto que usted ha mostrado voluntad para colaborar con nuestra investigación y que tiene las condiciones profesionales exigidas para emitir un criterio sobre el trabajo realizado, se necesitan algunos datos para procesar las opiniones que nos dará como parte de la aplicación del método de expertos. Debe guiarse por las indicaciones que se dan.

1. En la tabla aparece una escala que le permitirá expresar el nivel que usted considera tiene, para realizar un análisis relacionado con la enseñanza de la lógica de la programación. Marque con una “X” en la casilla que considere.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | X |

2. Si usted tuviera que argumentar sus criterios sobre un trabajo que trate el tema de la enseñanza de la lógica de la programación, tendría que apelar a sus conocimientos, intuición, experiencia, etc. Señale con una X en la siguiente tabla, la influencia que tienen estos elementos en los criterios que usted puede dar sobre el tema.

| Fuentes de argumentación | Alta | Media | Baja |
|---|---|---|---|
| Análisis teóricos realizados por usted. | | X | |
| Experiencia como profesor. | X | | |
| Trabajos consultados de autores nacionales. | | X | |
| Trabajos consultados de autores extranjeros. | | X | |
| Su propio conocimiento sobre el estado actual del problema en el extranjero. | | X | |
| Su intuición. | X | | |

Profesor (a): PABLO RODRIGUEZ

Email: xpamilo@hotmail.com

Puesto que usted ha mostrado voluntad para colaborar con nuestra investigación y que tiene las condiciones profesionales exigidas para emitir un criterio sobre el trabajo realizado, se necesitan algunos datos para procesar las opiniones que nos dará como parte de la aplicación del método de expertos. Debe guiarse por las indicaciones que se dan.

1. En la tabla aparece una escala que le permitirá expresar el nivel que usted considera tiene, para realizar un análisis relacionado con la enseñanza de la lógica de la programación. Marque con una “X” en la casilla que considere.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | X | | | |

2. Si usted tuviera que argumentar sus criterios sobre un trabajo que trate el tema de la enseñanza de la lógica de la programación, tendría que apelar a sus conocimientos, intuición, experiencia, etc. Señale con una X en la siguiente tabla, la influencia que tienen estos elementos en los criterios que usted puede dar sobre el tema.

| Fuentes de argumentación | Alta | Media | Baja |
|---|---|---|---|
| Análisis teóricos realizados por usted. | x | | |
| Experiencia como profesor. | | x | |
| Trabajos consultados de autores nacionales. | x | | |
| Trabajos consultados de autores extranjeros. | x | | |
| Su propio conocimiento sobre el estado actual del problema en el extranjero. | | | x |
| Su intuición. | | x | |

Profesor (a): CARLOS ESTRADA

Email: carlosh_estrada@hotmail.com

Puesto que usted ha mostrado voluntad para colaborar con nuestra investigación y que tiene las condiciones profesionales exigidas para emitir un criterio sobre el trabajo realizado, se necesitan algunos datos para procesar las opiniones que nos dará como parte de la aplicación del método de expertos. Debe guiarse por las indicaciones que se dan.

1. En la tabla aparece una escala que le permitirá expresar el nivel que usted considera tiene, para realizar un análisis relacionado con la enseñanza de la lógica de la programación. Marque con una “X” en la casilla que considere.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | X | |

2. Si usted tuviera que argumentar sus criterios sobre un trabajo que trate el tema de la enseñanza de la lógica de la programación, tendría que apelar a sus conocimientos, intuición, experiencia, etc. Señale con una X en la siguiente tabla, la influencia que tienen estos elementos en los criterios que usted puede dar sobre el tema.

| Fuentes de argumentación | Alta | Media | Baja |
|---|---|---|---|
| Análisis teóricos realizados por usted. | x | | |
| Experiencia como profesor. | x | | |
| Trabajos consultados de autores nacionales. | | x | |
| Trabajos consultados de autores extranjeros. | | x | |
| Su propio conocimiento sobre el estado actual del problema en el extranjero. | | x | |
| Su intuición. | | x | |

Profesor (a): Agustín Rivera

Email: ardubon@unicah.edu

Puesto que usted ha mostrado voluntad para colaborar con nuestra investigación y que tiene las condiciones profesionales exigidas para emitir un criterio sobre el trabajo realizado, se necesitan algunos datos para procesar las opiniones que nos dará como parte de la aplicación del método de expertos. Debe guiarse por las indicaciones que se dan.

1. En la tabla aparece una escala que le permitirá expresar el nivel que usted considera tiene, para realizar un análisis relacionado con la enseñanza de la lógica de la programación. Marque con una "X" en la casilla que considere.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | X | | | |

2. Si usted tuviera que argumentar sus criterios sobre un trabajo que trate el tema de la enseñanza de la lógica de la programación, tendría que apelar a sus conocimientos, intuición, experiencia, etc. Señale con una X en la siguiente tabla, la influencia que tienen estos elementos en los criterios que usted puede dar sobre el tema.

| Fuentes de argumentación | Alta | Media | Baja |
|---|---|---|---|
| Análisis teóricos realizados por usted. | X | | |
| Experiencia como profesor. | | X | |
| Trabajos consultados de autores nacionales. | | | X |
| Trabajos consultados de autores extranjeros. | | X | |
| Su propio conocimiento sobre el estado actual del problema ~~en el extranjero~~. | X | | |
| Su intuición. | X | | |

Profesor (a): DAVID TORMO

Email: dtormo69@hotmail.com

Puesto que usted ha mostrado voluntad para colaborar con nuestra investigación y que tiene las condiciones profesionales exigidas para emitir un criterio sobre el trabajo realizado, se necesitan algunos datos para procesar las opiniones que nos dará como parte de la aplicación del método de expertos. Debe guiarse por las indicaciones que se dan.

1. En la tabla aparece una escala que le permitirá expresar el nivel que usted considera tiene, para realizar un análisis relacionado con la enseñanza de la lógica de la programación. Marque con una “X” en la casilla que considere.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | X |

2. Si usted tuviera que argumentar sus criterios sobre un trabajo que trate el tema de la enseñanza de la lógica de la programación, tendría que apelar a sus conocimientos, intuición, experiencia, etc. Señale con una X en la siguiente tabla, la influencia que tienen estos elementos en los criterios que usted puede dar sobre el tema.

| Fuentes de argumentación | Alta | Media | Baja |
|---|---|---|---|
| Análisis teóricos realizados por usted. | x | | |
| Experiencia como profesor. | x | | |
| Trabajos consultados de autores nacionales. | | x | |
| Trabajos consultados de autores extranjeros. | | x | |
| Su propio conocimiento sobre el estado actual del problema en el extranjero. | | x | |
| Su intuición. | | x | |

Ingeniero Marco Antonio Diaz

Compañero profesor(a) debido a su preparación y voluntad de cooperar con la investigación titulada “Alternativa didáctica para la enseñanza de la lógica de la programación”, se necesita que conteste el siguiente cuestionario, después de haber leído las

**Instrucciones:**

- En cada pregunta aparece una escala del 1 al 5, que se interpreta de la siguiente manera:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| Inadecuado | Poco adecuado | Adecuado | Bastante adecuado | Muy adecuado |

- Ud. debe marcar con una **X** el número correspondiente a su respuesta de acuerdo a esta escala.

Cuestionario.

1. La solución que se propone al problema de investigación es original, en el sentido de no guardar analogías con los aportes de otros investigadores.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | X |

2. La investigación aporta un modelo que enriquece la teoría sobre la enseñanza de la lógica de la programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | X |

3. La alternativa elaborada posee una estructura adecuada en correspondencia con el modelo teórico seguido y las exigencias prácticas de la enseñanza-aprendizaje de la lógica de la programación independientemente del lenguaje de programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

4. La alternativa didáctica elaborada tiene claridad en el contenido de cada elemento del modelo.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | X |

5. La alternativa didáctica propuesta es de utilidad para potenciar el aprendizaje y desarrollo del pensamiento algorítmico asociado a los procesos de resolución de problemas con ayuda de la computadora, razonamiento y desarrollo de habilidades lógicas.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | X |

Lic Adolfo Di-Mare

Compañero profesor(a) debido a su preparación y voluntad de cooperar con la investigación titulada “Alternativa didáctica para la enseñanza de la lógica de la programación”, se necesita que conteste el siguiente cuestionario, después de haber leído las

**Instrucciones:**

- En cada pregunta aparece una escala del 1 al 5, que se interpreta de la siguiente manera:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| Inadecuado | Poco adecuado | Adecuado | Bastante adecuado | Muy adecuado |

- Ud. debe marcar con una **X** el número correspondiente a su respuesta de acuerdo a esta escala.

Cuestionario.

1. La solución que se propone al problema de investigación es original, en el sentido de no guardar analogías con los aportes de otros investigadores.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

2. La investigación aporta un modelo que enriquece la teoría sobre la enseñanza de la lógica de la programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

3. La alternativa elaborada posee una estructura adecuada en correspondencia con el modelo teórico seguido y las exigencias prácticas de la enseñanza-aprendizaje de la lógica de la programación independientemente del lenguaje de programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | X | | |

4. La alternativa didáctica elaborada tiene claridad en el contenido de cada elemento del modelo.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

5. La alternativa didáctica propuesta es de utilidad para potenciar el aprendizaje y desarrollo del pensamiento algorítmico asociado a los procesos de resolución de problemas con ayuda de la computadora, razonamiento y desarrollo de habilidades lógicas.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

Ingeniero Pablo Rodriguez

Compañero profesor(a) debido a su preparación y voluntad de cooperar con la investigación titulada “Alternativa didáctica para la enseñanza de la lógica de la programación”, se necesita que conteste el siguiente cuestionario, después de haber leído las

**Instrucciones:**

- En cada pregunta aparece una escala del 1 al 5, que se interpreta de la siguiente manera:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| Inadecuado | Poco adecuado | Adecuado | Bastante adecuado | Muy adecuado |

- Ud. debe marcar con una **X** el número correspondiente a su respuesta de acuerdo a esta escala.

Cuestionario.

1. La solución que se propone al problema de investigación es original, en el sentido de no guardar analogías con los aportes de otros investigadores.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | X |

2. La investigación aporta un modelo que enriquece la teoría sobre la enseñanza de la lógica de la programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | X |

3. La alternativa elaborada posee una estructura adecuada en correspondencia con el modelo teórico seguido y las exigencias prácticas de la enseñanza-aprendizaje de la lógica de la programación independientemente del lenguaje de programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

4. La alternativa didáctica elaborada tiene claridad en el contenido de cada elemento del modelo.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

5. La alternativa didáctica propuesta es de utilidad para potenciar el aprendizaje y desarrollo del pensamiento algorítmico asociado a los procesos de resolución de problemas con ayuda de la computadora, razonamiento y desarrollo de habilidades lógicas.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

Ingeniero Carlos Estrada

Compañero profesor(a) debido a su preparación y voluntad de cooperar con la investigación titulada “Alternativa didáctica para la enseñanza de la lógica de la programación”, se necesita que conteste el siguiente cuestionario, después de haber leído las

**Instrucciones:**

- En cada pregunta aparece una escala del 1 al 5, que se interpreta de la siguiente manera:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| Inadecuado | Poco adecuado | Adecuado | Bastante adecuado | Muy adecuado |

- Ud. debe marcar con una **X** el número correspondiente a su respuesta de acuerdo a esta escala.

Cuestionario.

1. La solución que se propone al problema de investigación es original, en el sentido de no guardar analogías con los aportes de otros investigadores.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | X | | | |

2. La investigación aporta un modelo que enriquece la teoría sobre la enseñanza de la lógica de la programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

3. La alternativa elaborada posee una estructura adecuada en correspondencia con el modelo teórico seguido y las exigencias prácticas de la enseñanza-aprendizaje de la lógica de la programación independientemente del lenguaje de programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

4. La alternativa didáctica elaborada tiene claridad en el contenido de cada elemento del modelo.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | X |

5. La alternativa didáctica propuesta es de utilidad para potenciar el aprendizaje y desarrollo del pensamiento algorítmico asociado a los procesos de resolución de problemas con ayuda de la computadora, razonamiento y desarrollo de habilidades lógicas.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | |

Ingeniero Agustin Rivera

Compañero profesor(a) debido a su preparación y voluntad de cooperar con la investigación titulada “Alternativa didáctica para la enseñanza de la lógica de la programación”, se necesita que conteste el siguiente cuestionario, después de haber leído las

**Instrucciones:**

- En cada pregunta aparece una escala del 1 al 5, que se interpreta de la siguiente manera:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| Inadecuado | Poco adecuado | Adecuado | Bastante adecuado | Muy adecuado |

- Ud. debe marcar con una **X** el número correspondiente a su respuesta de acuerdo a esta escala.

Cuestionario.

1. La solución que se propone al problema de investigación es original, en el sentido de no guardar analogías con los aportes de otros investigadores.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

2. La investigación aporta un modelo que enriquece la teoría sobre la enseñanza de la lógica de la programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

3. La alternativa elaborada posee una estructura adecuada en correspondencia con el modelo teórico seguido y las exigencias prácticas de la enseñanza-aprendizaje de la lógica de la programación independientemente del lenguaje de programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

4. La alternativa didáctica elaborada tiene claridad en el contenido de cada elemento del modelo.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | X |

5. La alternativa didáctica propuesta es de utilidad para potenciar el aprendizaje y desarrollo del pensamiento algorítmico asociado a los procesos de resolución de problemas con ayuda de la computadora, razonamiento y desarrollo de habilidades lógicas.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | |

Ingeniero David Tormo

Compañero profesor(a) debido a su preparación y voluntad de cooperar con la investigación titulada “Alternativa didáctica para la enseñanza de la lógica de la programación”, se necesita que conteste el siguiente cuestionario, después de haber leído las

**Instrucciones:**

- En cada pregunta aparece una escala del 1 al 5, que se interpreta de la siguiente manera:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| Inadecuado | Poco adecuado | Adecuado | Bastante adecuado | Muy adecuado |

- Ud. debe marcar con una **X** el número correspondiente a su respuesta de acuerdo a esta escala.

Cuestionario.

1. La solución que se propone al problema de investigación es original, en el sentido de no guardar analogías con los aportes de otros investigadores.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | X |

2. La investigación aporta un modelo que enriquece la teoría sobre la enseñanza de la lógica de la programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | X | |

3. La alternativa elaborada posee una estructura adecuada en correspondencia con el modelo teórico seguido y las exigencias prácticas de la enseñanza-aprendizaje de la lógica de la programación independientemente del lenguaje de programación.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | X |

4. La alternativa didáctica elaborada tiene claridad en el contenido de cada elemento del modelo.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | X |

5. La alternativa didáctica propuesta es de utilidad para potenciar el aprendizaje y desarrollo del pensamiento algorítmico asociado a los procesos de resolución de problemas con ayuda de la computadora, razonamiento y desarrollo de habilidades lógicas.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | | | | |

New Reply Reply all Forward Delete Junk Move to

## Re: Consulta tesis maestria logica de la programacion

From: **Adolfo Di Mare** (adolfo@di-mare.com)
Sent: Friday, February 10, 2006 1:17:49 PM
To: Edgar Vasquez (edvasal@hotmail.com)

```
> Gracias por contestarme. Mira, con la lógica de la programación me
> refiero a la enseñanza de la lógica de los algoritmos computacionales.
Ok

> Y mi problema es como enseñar la lógica de los algoritmos a alumnos
> novatos, o sea a aquellos alumnos que nunca han programado. Bueno, te
> agradecere me ayudes con las rondas siguientes

La dificultad que hay que vences es lograr que las personas
comprendan que programar es encontrar una "secuencia" de
"decisiones" que llegan a solucionar una tarea. En otras
palabras, realmente lo que hay que hacer es lograr que los
estudiantes aprendan a "planificar" usando un conjunto reducido
de palabras.

Como todos hemos aprendido a programar rápido, podemos concluir
que una persona "normal" adquiere la destreza en 2 o 3 meses de
entrenamiento en las aulas. Reducir ese tiempo no ayuda mucho,
porque un buen programador requiere por lo menos 2 años de
entrenamiento para formarse, pues debe conocer las herramientas
disponibles (lenguaje, bibliotecas, plataformas) pero también
debe adquirir la capacidad de "diseñar" de manera que sus
programas puedan ser "baratos", en donde defino barato como
rápido de modificar o de arreglar, lo que implica que se use
poca plata para afinarlo.
Adolfo
///
```

## RV: Consulta sobre tesis/COMENTARIOS

From: **Marco Diaz** (marco.diaz@usps.edu)
Sent: Tuesday, March 14, 2006 4:07:15 PM
To: edgar vasquez (edvasal@hotmail.com)
Modelo p...doc (60.6 KB), Instrumen...doc (45.7 KB)



El modelo está super interesante sin embargo es crítico llevar a cabo la evaluación al inicio del curso para identificar los diferentes estilos de aprendizaje y poder organizar los grupo

Es importante antes de iniciar el curso y aplicar la evaluación, aplicar el modelo en el diseño de las actividades de aprendizaje dentro y fuera del salón de clases, dichas actividades deberán responder al estilo de aprendizaje identificado en los estudiantes, otro momento donde se deberá aplicar el modelo es en el diseño de los sistemas que evaluarán el aprendizaje

Estoy de acuerdo contigo en que el aprendizaje de las estructuras lógicas no resulta agradable para los estudiantes, ni es fácil de captar debido a que las actividades de aprendizaje generalmente se diseñan para un tipo de estudiantes que posee el estilo de aprendizaje teórico que no es precisamente el dominate en los grupos, esto da como resultado que la mayoría de los estudiantes de esta materia tiene dificultades para apreciar y aprender la