UNIVERSIDAD DE PANAMÁ
FACULTAD DE CIENCIAS NATURALES Y EXACTAS
ESCUELA DE MATEMÁTICA
CENTRO REGIONAL UNIVERSITARIO DE VERAGUAS

ASOCIACIÓN NACIONAL DE ESTUDIANTES DE MATEMÁTICA (A.N.E.MAT.)
CAPÍTULO DE VERAGUAS

SEMANA DE LA MATEMÁTICA

CONFERENCIA: ***DISEÑO Y CREACIÓN DE PROGRAMAS USANDO COMPUTADORAS.***

EXPOSITOR: ***RAÚL ENRIQUE DUTARI DUTARI.***

FECHA: ***26 DE NOVIEMBRE DE 1992.***

HORA: ***10:25 A. M.***

LUGAR: ***AULA B-1 DEL CENTRO REGIONAL UNIVERSITARIO DE VERAGUAS.***

DIRIGIDA A: ***PROFESORES Y ESTUDIANTES UNIVERSITARIOS DE MATEMÁTICA QUE PARTICIPARON EN EL EVENTO.***

DURACIÓN: ***40 MINUTOS.***

OBJETIVOS GENERALES

1. Conocer los rasgos generales de la programación de computadoras.

2. Esbozar los pasos a seguir, al diseñar programas de computadora.

3. Visualizar, dentro de cada paso, los aspectos más importantes que se deben tomar en cuenta, al diseñar programas de computadora.

OBJETIVOS ESPECÍFICOS

1. Comprender la importancia que tiene la definición del problema, en el diseño e implementación de un programa de computadora.

2. Conocer los métodos fundamentales que se pueden seguir en la solución de problemas.

3. Comprender el proceso general que se aplica en el diseño de un algoritmo.

4. Comprender el proceso seguido para transformar un algoritmo a programa de computadora.

5. Identificar las principales fuentes de error que pueden existir dentro de un programa.

6. Conocer la importancia que tiene la documentación de los programas.

7. Comprender en qué consiste el mantenimiento de un programa de computadora.

TABLA DE CONTENIDOS

1. Observaciones Preliminares.

El diseño de un programa de computadora es un proceso delicado. En él, en principio, le señalamos a la máquina, exactamente, cuáles son los procesos que debe seguir para el procesamiento de la información.

Es decir, “NOSOTROS” le señalamos a “LA COMPUTADORA” lo que debe hacer. Quién usa la computadora, tiene la responsabilidad de señalarle “EXACTAMENTE”, lo que debe hacer. La máquina siempre realizará lo que programemos, y solamente eso. Todo lo que hagamos usando computadoras debe enmarcarse, necesariamente, dentro de este lineamiento fundamental.

La programación de computadoras es, para la mayoría de las personas, un tema desconocido. No se atreven a incursionar en él, tal vez porque temen a lo desconocido. Esta conferencia pretende, precisamente, darnos algunas luces acerca de esta materia.

2. La Creación De Programas Para Computadoras: Un Proceso Con Siete Pasos.

La creación de un programa de computadora significa más que el simple trabajo mecánico del hombre y la máquina. Constituye la culminación de todo un proceso complejo, que involucra a los dos entes, aportando cada uno su parte en la realización de la tarea.

Esta tarea ha sido sistematizada por los especialistas del área, buscando optimizar la eficiencia de los programas producidos. La sistematización comprende un proceso que consta de siete pasos. Estos son:

⇒ Definición del problema.

⇒ Selección del método de solución.

⇒ Creación del algoritmo.

⇒ Programación del algoritmo.

⇒ Depuración y documentación del programa.

⇒ Validación de la solución.

⇒ Instalación, producción y mantenimiento del programa.

La denominación exacta de cada uno de ellos, así como el número de pasos involucrados, generalmente varía de un autor a otro. Sin embargo, en esencia, son la “APLICACIÓN DEL MÉTODO CIENTÍFICO” a la solución de un problema práctico.

A continuación, nos referiremos a cada uno de estos pasos con más detenimiento.

2.1. Definición Del Problema.

La definición o enfoque del problema es el primer paso en el diseño de un programa de computadora. Consiste, fundamentalmente, en definir cual es la “SITUACIÓN QUE INTENTAMOS RESOLVER”.

La definición del problema debe ser tan clara y precisa como sea posible, de modo que no pueda ser malinterpretada por otras personas. Dicho requisito es

muy difícil de lograr. Sin embargo, este paso fundamenta todo lo que se hará en los otros pasos. Luego, no se puede avanzar en el proceso sin contar con esa definición.

Simultáneamente se establece el problema a resolver, deben diseñarse los objetivos específicos que se espera lograr al resolverlo, así como el alcance que ellos tendrán. Éstos aspectos de la definición del problema se relacionan, íntimamente, entre sí.

2.2. Selección Del Método De Solución.

Cuando el problema está bien enfocado, se le tiene que encontrar una solución. El método empleado para obtenerla puede ser planteado de distintas maneras. Pero, casi todos, podemos agruparlos dentro de tres categorías básicas. Ellas son, en orden creciente de importancia:

⇒ Métodos geométricos.

⇒ Métodos analíticos.

⇒ Métodos iterativos.

A continuación, explicaremos con más detalle esta clasificación.

2.2.1. Métodos Geométricos.

Los métodos geométricos, generalmente, emplean gráficas para resolver los problemas. Ellas pueden ser de tipo sintético o analítico.

De los dos enfoques geométricos, el analítico tiene más vigencia en la programación de computadoras.

Un ejemplo del último enfoque, lo vemos en la solución de sistemas de ecuaciones simultáneas por el método gráfico.

2.2.2. Métodos Analíticos.

Los métodos analíticos se basan en planteamientos derivados de las ramas clásicas de las matemáticas abstractas (álgebra, geometría analítica y análisis matemático, entre otras).

Generalmente, consisten en la aplicación de una fórmula o método preestablecido, con base a los supuestos con que fue construido. Ellos originan, el último enfoque para la solución de problemas (los planteamientos iterativos).

La programación de estos métodos puede presentar grados muy diversos de dificultad (dependiendo de cada fórmula particular).

Como ejemplo de ellos, tenemos la fórmula para la solución de una ecuación cuadrática, en el caso general.

2.2.3. Métodos Iterativos.

Los métodos iterativos son el enfoque más importante para resolver problemas, usando computadoras. También se fundamentan en las ramas clásicas de la matemática abstracta.

Se caracterizan porque emplean procesos iterativos y recurrentes en la solución de los problemas (fundamentados en la teoría de sucesiones y series del análisis matemático).

Independientemente de los tipos de métodos establecidos, en la práctica, la solución de un problema puede involucrar la combinación de uno o más métodos conocidos, y hasta el diseño de uno completamente nuevo.

Cuando no es posible encontrar una vía para resolver al problema, se hace necesario su replanteamiento, para simplificarlo. En consecuencia, se facilita su solución. Por otro lado, hay casos en los que un procedimiento de solución resulta obvio.

Su programación también puede presentar grados muy diversos de dificultad.

Como ejemplo de estos métodos, tenemos el cálculo de raíces de una función, empleando el procedimiento de bisección de Bolzano.

2.3. Creación Del Algoritmo.

Cuando se ha escogido un método de solución, debemos establecerlo, por escrito, con tanto detalle como sea posible. Se debe establecer como una secuencia de pasos “BIEN DEFINIDA” y tan “COMPLETA” como sea posible, para conservar la claridad de la definición del problema y del método que se aplicará en su solución.

No debe dejarse nada al azar. En ese sentido, si se presenta un problema particular, que el método no puede resolver, él debe ser capaz, al menos, de

informar al usuario que no puede obtener una respuesta satisfactoria con ese enfoque.

Estas necesidades particulares se satisfacen a través del empleo de un algoritmo, que define “INEQUÍVOCAMENTE” la lógica usada en la solución del problema. En esencia, ellos no difieren significativamente de los usados dentro del álgebra cotidiana. En consecuencia, debemos construirlos, hasta donde sea posible, empleando el lenguaje natural.

Sin embargo, a nivel de la programación de computadoras, debemos ajustar el lenguaje natural a las tres clases fundamentales de órdenes estructuradas descendentes (la asignación, la decisión y la repetición). Adicionalmente, se deben implementar, en el algoritmo, estructuras adicionales de:

⇒ Entrada y salida de la información.

⇒ Cálculo numérico y manipulación de datos.

⇒ Comparación lógica y relacional.

⇒ Almacenamiento y recuperación de información.

La creación de un algoritmo es un proceso repetitivo, que se refina paulatinamente. Se parte de un esquema burdo de la lógica que soluciona el problema, y se llega (luego de mucho esfuerzo), al grado de detalle que caracteriza a los programas de computadora. Cuando logramos ese nivel en el planteamiento del algoritmo, su programación se realiza prácticamente sin esfuerzo (se reduce a la “TRADUCCIÓN” de las órdenes al formato que comprende la máquina).

2.4. Programación Del Algoritmo.

Dependiendo del grado de precisión logrado en la etapa anterior, el paso que mencionaremos en este apartado será, directamente, más sencillo de lograr.

En esencia, cada una de las pautas lógicas establecidas dentro del algoritmo, se debe traducir a una instrucción equivalente en el lenguaje de programación. Así, las características del algoritmo definen, directamente, las características de su programación. Técnicamente, hablamos de “CODIFICAR EL ALGORITMO”.

Posteriormente, el listado del programa es grabado en un medio de almacenamiento de la computadora. Luego, es traducido al lenguaje de la máquina empleando un conjunto de programas, denominados compiladores-enlazadores o intérpretes. Ellos, siguen procedimientos específicos de empleo, dependiendo de cada sistema particular que se use.

Muchos programadores novatos sienten la “TENTACIÓN” de escribir la solución del problema directamente en el lenguaje de programación, sin completar las etapas previas a este paso. Esta acción es una pérdida de tiempo, ya que generalmente, estos intentos no abarcan la totalidad del problema, o dejan algunos detalles al azar. Estas fallas, en su momento, se harán evidentes y habrá que retroceder a esta etapa.

2.5. Depuración Y Documentación Del Programa.

Es de humanos cometer errores, pero la verdad de esta afirmación es más evidente para quienes han tenido experiencias en la programación de computadoras. Prácticamente ningún analista logra que sus programas funcionen

correctamente en el primer intento. Debe someterlos a un proceso de corrección y depuración de errores de todo tipo.

Así, los compiladores-enlazadores e intérpretes requieren que el programa cumpla con toda una serie de reglas semánticas y sintácticas preestablecidas dentro del lenguaje, para realizar la traducción al lenguaje de máquina. Una coma mal ubicada, un paréntesis que falte o sobre, una orden mal escrita; todas son situaciones que provocan que un programa no se pueda compilar, o funcione incorrectamente.

Adicionalmente, en esta etapa podemos enfrentarnos a una serie de situaciones no previstas dentro del proceso. El caso más importante lo vemos cuando surgen resultados incorrectos; debido a que los algoritmos, que son exactos en teoría, sufren los problemas conocidos como error por redondeo y error por truncamiento, de las cifras involucradas en los cálculos. Sus efectos pueden ser tan catastróficos al aparecer, que llegan a invalidar completamente la efectividad del algoritmo (es decir, harán que no resuelva el problema originalmente planteado).

Estos errores deben ser eliminados del programa a través de un cuidadoso y exhaustivo análisis de los procesos previos a este paso. Es decir, revisando el método de solución de una manera más detenida, y, considerando los detalles de implementación en la computadora.

Por otro lado, se deben crear documentos escritos, complementarios al programa diseñado. Esta, información debe reunirse de manera histórica y sistemática, paralelamente a los cambios que sufra el programa en su vida útil. Estos escritos se conocen como “DOCUMENTACIÓN DEL SISTEMA” y deben reflejar, en lo posible, lo que pensaba el programador al diseñar la aplicación. Debe reunir, entre otras cosas, detalles acerca de:

⇒ La definición del problema que se intenta resolver.

⇒ La descripción del método de solución empleado.

⇒ “TODOS” los algoritmos y listados de los módulos que integran al programa principal y sus partes.

⇒ La enumeración detallada de “TODAS” las instrucciones que pueden realizar los operadores del programa, así como “TODOS” los pasos para realizarlas (manual del usuario).

⇒ La administración del teclado, por parte del programa.

⇒ Las convenciones que deben asumirse en el manejo de la información.

⇒ Los controles para verificar la integridad de la información.

⇒ Las limitaciones, libertades y potencialidades que tiene el programa.

Estos documentos deben reposar en el código fuente y en copias impresas separadas. Dicha información se orienta a los posibles usuarios del programa y a los programadores que, en el futuro, modifiquen su contenido.

La importancia de estos escritos se debe a que, el programador “GUARDA” la lógica del sistema, en su mente, cuando está recién confeccionado. Sin embargo, con el transcurrir del tiempo, esa información se olvida y, consecuentemente, el código fuente que una vez fue claro, lógico y sencillo; nos parecerá incomprensible, e imposible de modificar.

2.6. Validación De La Solución.

Si nuestro algoritmo no a sido objeto de un estudio cuidadoso y profundo en las etapas iniciales, al llegar a este punto, nos enfrentaremos a preguntas tales como:

- ⇒ ¿Las soluciones obtenidas tienen sentido al enfrentarlas a la teoría del problema planteado?

- ⇒ ¿Los casos triviales son contemplados adecuadamente dentro del enfoque de la solución planteada?

- ⇒ ¿Se obtienen las soluciones correctas al aplicar el algoritmo a una amplia gama de problemas específicos, con respuesta conocida?

- ⇒ ¿Las simplificaciones realizadas inicialmente (al definir el problema) afectan al algoritmo, al punto que sus resultados no tienen ninguna utilidad práctica?

Las preguntas planteadas, evidentemente, no las puede responder la computadora. Es la persona que realizó el análisis del problema quien debe enfrentarlas. Dependiendo del grado de eficiencia que se logró en las etapas iniciales del análisis del problema, la respuesta a estas preguntas será una trivialidad, o nos obligará a replantear todo el proyecto desde la definición del problema. Esta etapa se cumple, a veces, simultáneamente a la depuración del código fuente.

2.7. Instalación, Producción Y Mantenimiento Del Programa.

Finalmente, y luego de muchos esfuerzos, el programador logra que su diseño funcione correctamente. Nuestro programa es instalado permanentemente como sistema y entra a la fase de producción. En ella obtenemos los beneficios de la rapidez en los cálculos, al resolver los problemas planteados, con gran velocidad y confiabilidad. En este momento, podemos pensar que la faena del programador ha concluido. Desgraciadamente, eso es falso.

La implantación del programa no se puede realizar a la ligera. Es necesario que se establezca un período de “PRUEBA” para el sistema. En ese tiempo, se verificará que, en realidad, el programa ejecuta todos los procesos originalmente contemplados en el diseño. Esto se logra a través de la comparación de resultados logrados con la máquina, frente a los obtenidos por métodos manuales. Si es necesario, el programa es nuevamente revisado y corregido. Dichas correcciones deben ser reflejadas, lógicamente, en la documentación escrita que lo acompaña.

Por otro lado, al igual que cualquier creación del hombre, los programas de computadora no son eternos. Deben recibir un mantenimiento adecuado, con el objeto de que se mantengan a tono con las necesidades que van surgiendo, dinámicamente, con el transcurrir del tiempo.

Por ejemplo, pueden surgir nuevas situaciones que originalmente no fueron previstas, y requieren la atención de quienes emplean el programa. En tal caso, el programa debe ser modificado para adaptarlo a esas circunstancias imprevistas inicialmente.

Usualmente, el proceso de mantenimiento se lleva a cabo en tanto el programa sea provechoso. Una vez deja de ser útil, sencillamente se abandona y la información que manejaba (si es necesario) es transferida a otros sistemas para que la aprovechen.

3. Comentarios Finales.

El proceso al que nos hemos referido, se ve como algo bastante difícil y tedioso. Sin embargo, la práctica de la programación de computadoras siguiendo este procedimiento, acarrea grandes beneficios, a largo plazo. Entre ellos, tenemos la creación de programas altamente confiables y eficientes. Para facilitar la comprensión de este proceso, hemos confeccionado la figura que, a continuación, mostramos:

Etapas a Cumplir en el Diseño de un Programa de Computadora

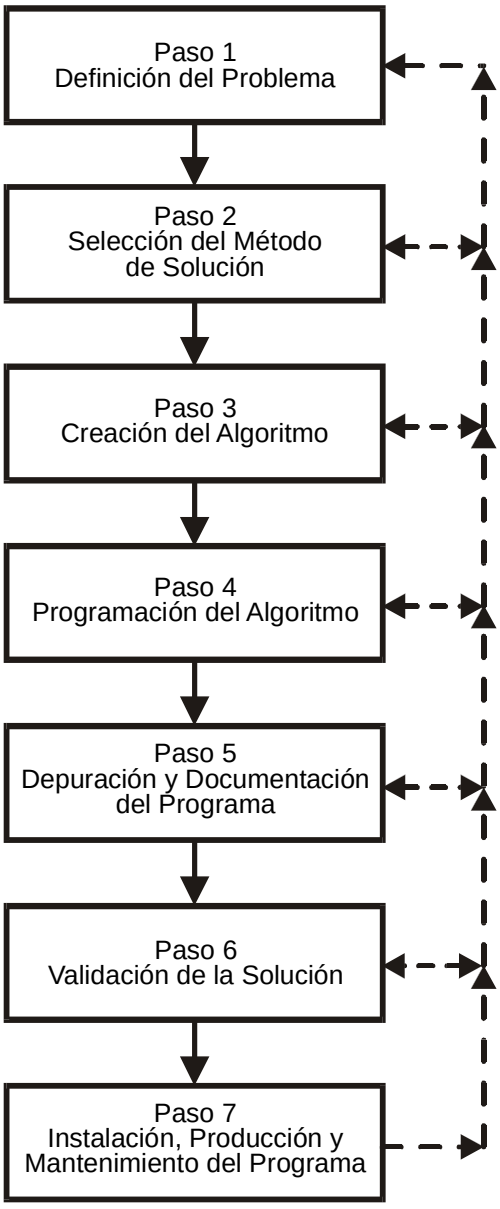

Figura 1: *Los siete pasos necesarios para producir programas de calidad. Las flechas con trazo continuo indican la secuencia a seguir, obligatoriamente. Las fechas punteadas señalan que se puede retroceder en el proceso cuando sea necesario, para mejorar el producto final.*

De ese proceso, podemos desprender toda una serie de señalamientos bien claros, a manera de conclusiones.

⇒ La computadora no es capaz de resolver problemas por sí sola. Simplemente puede realizar secuencias de pasos preestablecidas, a través de los programas que se le han instalado.

⇒ La computadora no releva al ser humano de la responsabilidad de estructurar su trabajo. Al contrario, lo obliga a realizar una planificación más detallada y cuidadosa del mismo.

⇒ La computadora no releva al hombre de interpretar los resultados obtenidos. Ella le exige un conocimiento más amplio de los problemas que enfrenta, al tener que conjugar una serie de situaciones no previstas cuando trabaja manualmente, y que pasan a formar parte adicional del problema que enfrenta.

BIBLIOGRAFÍA

1. ALLEN SMITH, W. Análisis Numérico. Traducido por Francisco Javier Sánchez Bernabe. Primera edición. México, D.F., México: Prentice-Hall, 1988. 608 páginas.

2. CHAPRA, Steven, y CANALE, Raymond P. Métodos numéricos para ingenieros con aplicaciones en computadoras personales. Traducido por Carlos Zapata S. Primera edición. México D.F., México: McGraw-Hill, 1990. 641 páginas.

3. GOTTFRIED, Byron S. Programación en Pascal. Traducido por Alfredo Bautista Paloma. Primera edición. México D.F., México: McGraw-Hill, 1988. 398 páginas.

4. JAMES, Merlin L, SMITH, Gerald M., y WOLFORD, James C. Métodos numéricos aplicados a la computación digital con FORTRAN. Traducido por José A. Nieto Ramírez. Primera edición. México, D.F., México: Representaciones y Servicios de Ingeniería, 1973. 575 páginas.

5. JOYANES AGUILAR, Luis. Turbo Basic. Manual de Programación. Primera edición. Madrid, España: McGraw-Hill, 1989. 525 páginas.

6. HENRICE, Peter. Elementos de análisis numérico. Traducido por Federico Velasco Coba. Primera edición. México, D.F., México: Trillas, 1972. 363 páginas.

7. KORFHAGE, Robert R. Lógica y algoritmos: Con aplicaciones a las ciencias de la computación e información. Traducido por Federico Velasco C. Primera edición. México, D.F., México: Limusa, 1970. 222 páginas.

8. LUTHE, Rodolfo, OLIVERA, Antonio, y SCHUTZ, Fernando. Métodos numéricos. Primera edición. México, D.F., México: Limusa, 1986. 443 páginas.

9. MCCRAKEN, Daniel D., y DORN, William S. Métodos numéricos y programación FORTRAN. Traducido por José A. Nieto Ramírez. Primera edición. México, D.F., México: Limusa, 1986. 476 páginas.

10. NELL, Dale, y LILLY, Susan C. Pascal y estructura de datos. Traducido por José María Troya Linero. Primera edición. México, D.F., México: McGraw-Hill, 1988. 491 páginas.

11. RALSTON, Anthony. Introducción al análisis numérico. Traducido por Carlos E. Cervántes de Gortari. Primera edición. México, D.F., México: Limusa, 1970. 629 páginas.

12. REINHARDT, Fritz y SOEDER, Heinrich. Atlas de matemáticas 1: Fundamentos, álgebra y geometría. Traducido por Juan Luis Vázquez Suárez y Mario Rodríguez Artalejo. Primera edición. Madrid, España: Alianza Editorial, 1984. 265 páginas.

13. SANDERS, Donald H. Informática: Presente y futuro. Traducido por Roberto Luis Escalona. Tercera Edición. México D.F., México: McGraw-Hill, 1991. 887 páginas.

14. SANTALÓ SORS, Marcelo, y CARBONELL CHAURE, Vicente. Geometría Analítica. Primera Edición. México D.F., México: Librería de Manuel Porrúa, 1959. 255 páginas.

15. SCHEID, Francis. Análisis numérico. Traducido por Hernando Alonso Castillo. Primera Edición. México D.F., México: McGraw-Hill, 1972. 422 páginas.

16. SCHEID, Francis. Introducción a la ciencia de las computadoras. Traducido por Alberto Jaime Sisa. Segunda Edición. México D.F., México: McGraw-Hill, 1985. 402 páginas.

17. SCHEID, Francis, y Di Constanzo, Rosa Elena. Métodos numéricos. Traducido por Gabriel Nagore Cázares. Segunda Edición. México D.F., México: McGraw-Hill, 1991. 709 páginas.

18. SQUIRE, Enid. Introducción al diseño de sistemas. Traducido por Jaime Luis Valls Cabrear. Primera Edición. México D.F., México: Fondo Educativo Interamericano, 1984. 345 páginas.