Programación estructurada y orientada a objetos

UN ENFOQUE ALGORÍTMICO

3ª Edición

LEOBARDO LÓPEZ ROMÁN

libroWeb

Contenidos Interactivos Web

Alfaomega

Programación estructurada y orientada a objetos

UN ENFOQUE ALGORÍTMICO *3ª Edición*

Programación estructurada y orientada a objetos

UN ENFOQUE ALGORÍTMICO *3ª Edición*

LEOBARDO LÓPEZ ROMÁN

Alfaomega

Datos catalográficos:

López Román, Leobardo
Programación estructurada y orientada a objetos. Un enfoque algorítmico
Tercera edición
Alfaomega Grupo Editor S.A. de C.V., México

ISBN 978-607-707-211-9

Formato: 21 x 24 cm Páginas: 576

Programación estructurada y orientada a objetos. Un enfoque algorítmico
Leobardo López Román

Tercera Edición: Alfaomega Grupo Editor, México, agosto de 2011

Pitágoras 1139, Col. Del Valle, 03100, México D.F.
Miembro de la Cámara Nacional de la Industria Editorial Mexicana
Registro Nº 2317
Página Web: http://www.alfaomega.com.mx
E-mail: atencionalcliente@alfaomega.com.mx

ISBN: 978-607-707-211-9

Impreso en México. Printed in Mexico.

Empresas del grupo:

Argentina: Alfaomega Grupo Editor Argentino, S.A.
Paraguay 1307 P.B. "11", Buenos Aires, Argentina, C.P. 1057
Tel.: (54-11) 4811-7183 / 0887 - E-mail: ventas@alfaomegaeditor.com.ar

México: Alfaomega Grupo Editor, S.A. de C.V.
Pitágoras 1139, Col. Del Valle, México, D.F., México, C.P. 03100
Tel.: (52-55) 5575-5022 - Fax: (52-55) 5575-2420 / 2490. Sin costo: 01-800-020-4396
E-mail: atencionalcliente@alfaomega.com.mx

Colombia: Alfaomega Colombiana S.A.
Carrera 15 No. 64 A 29, Bogotá, Colombia - PBX (57-1) 2100122 - Fax: (57-1) 6068648
E-mail: scliente@alfaomega.com.mx

Chile: Alfaomega Grupo Editor, S.A.
Dr. La Sierra 1437, Providencia, Santiago, Chile
Tel.: (56-2) 235-4248 - Fax: (56-2) 235-5786 - E-mail: agechile@alfaomega.cl

A mis padres Socorro y Celedonio †
A mi esposa Maricela Villa Acosta
A mis hijos Leobardo y Eleazar

Leobardo López Román

Acerca del Autor

Leobardo López Román

Nació en Guamúchil, Sinaloa, México.

Es Licenciado en Informática en Sistemas de Información, egresado del Instituto Tecnológico de Culiacán, Sinaloa, México, en 1981.

En 1982 obtuvo el Diplomado en Ciencias de la Computación; y en 1990 la Maestría en Ciencias de la Computación en la Fundación Arturo Rosenblueth, A.C. En México, D.F.

Profesionalmente se ha desempeñado como programador, programador-analista y analista de sistemas en empresas públicas, privadas, de investigación y educativas.

Desde 1982 ha sido profesor de programación de computadoras en diversas instituciones de educación superior:

- Instituto Tecnológico de Ciudad Guzmán
- Instituto Tecnológico de Culiacán
- Instituto Tecnológico de Hermosillo
- Actualmente es Maestro de Tiempo Completo Titular con perfil PROMEP en la Universidad de Sonora.

Ha recibido el Premio Anual de Profesor Distinguido de la División de Ingeniería de la Universidad de Sonora en 1995, 1998, 2003, 2005 y 2006.

Asimismo ha desempeñado otras actividades académicas, como revisión curricular y diseño y creación de postgrados en Computación e Informática, presidente de academia de Computación y Tecnología de la Información, Coordinador de la Licenciatura en Informática, ha dirigido más de 40 trabajos profesionales para titulación, y ha dictado conferencias a nivel nacional e internacional.

Es autor de los libros:

- "Metodología de la Programación Orientada a Objetos", Alfaomega, México, 2006.
- "Programación Estructurada en Lenguaje C", Alfaomega, México, 2005.
- "Programación Estructurada un enfoque algorítmico 2da. Edición", Alfaomega, México, 2003.
- "Programación Estructurada en Turbo Pascal 7", Alfaomega, México, 1998.
- "Programación Estructurada un enfoque algorítmico", Alfaomega, México, 1994.

Mensaje del Editor

Los conocimientos son esenciales en el desempeño profesional. Sin ellos es imposible lograr las habilidades para competir laboralmente. La universidad o las instituciones de formación para el trabajo ofrecen la oportunidad de adquirir conocimientos que serán aprovechados más adelante en beneficio propio y de la sociedad. El avance de la ciencia y de la técnica hace necesario actualizar continuamente esos conocimientos. Cuando se toma la decisión de embarcarse en una vida profesional, se adquiere un compromiso de por vida: mantenerse al día en los conocimientos del área u oficio que se ha decidido desempeñar.

Alfaomega tiene por misión ofrecerles a estudiantes y profesionales conocimientos actualizados dentro de lineamientos pedagógicos que faciliten su utilización y permitan desarrollar las competencias requeridas por una profesión determinada. Alfaomega espera ser su compañera profesional en este viaje de por vida por el mundo del conocimiento.

Alfaomega hace uso de los medios impresos tradicionales en combinación con las tecnologías de la información y las comunicaciones (IT) para facilitar el aprendizaje. Libros como éste tienen su complemento en una página Web, en donde el alumno y su profesor encontrarán materiales adicionales, información actualizada, pruebas (test) de autoevaluación, diapositivas y vínculos con otros sitios Web relacionados.

Esta obra contiene numerosos gráficos, cuadros y otros recursos para despertar el interés del estudiante, y facilitarle la comprensión y apropiación del conocimiento.

Cada capítulo se desarrolla con argumentos presentados en forma sencilla y estructurada claramente hacia los objetivos y metas propuestas. Cada capítulo concluye con diversas actividades pedagógicas para asegurar la asimilación del conocimiento y su extensión y actualización futuras.

Los libros de Alfaomega están diseñados para ser utilizados dentro de los procesos de enseñanza-aprendizaje, y pueden ser usados como textos guía en diversos cursos o como apoyo para reforzar el desarrollo profesional.

Alfaomega espera contribuir así a la formación y el desarrollo de profesionales exitosos para beneficio de la sociedad.

Agradecimientos

A Dios, que es mi guía e inspiración, le doy las gracias por su bondad y generosidad al darme salud, fuerza de voluntad y capacidad para desarrollar mi obra editorial.

Mi gratitud para esa gran y noble institución que es la Universidad de Sonora, que en su seno he encontrado la oportunidad de desarrollar mi obra editorial; la cual, ya es parte de su esencia misma.

También quiero agradecer a mis compañeros maestros y alumnos del Departamento de Ingeniería Industrial y de Sistemas de la Universidad de Sonora; y del Instituto Tecnológico de Hermosillo quienes de una u otra forma han contribuido para que mi obra editorial sea una realidad.

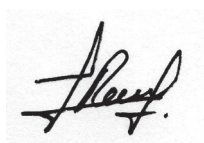
Asimismo agradezco a mis maestros y amigos Manuel Sáiz y Maria Luisa López López (Malichi) sus consejos y su guía.

Un agradecimiento muy especial para mi esposa Maricela Villa Acosta, y para mis hijos Leobardo y Eleazar, a quienes les he quitado mucho tiempo y paciencia para desarrollar esta obra.

Y especialmente a Usted, amigo lector, que me ha dado la oportunidad, quiero serle útil a través de mi obra editorial: En primera instancia a través de mis libros anteriores, ahora a través de este libro; y, en un futuro espero poner a su disposición un libro que implemente esta metodología en lenguaje C, C++, Java u otro.

Indudablemente habrán quedado errores u omisiones las cuales espero subsanar en una futura edición, así que, amable lector, si tiene alguna sugerencia para el mejoramiento del mismo, mucho le agradeceré me la haga llegar. Gracias.

"Escribir un libro, es el más alto honor de un maestro"



Leobardo López Román

Universidad de Sonora
Hermosillo, Sonora, México
llopez@industrial.uson.mx

Contenido

Convenciones utilizadas en el texto

	Conceptos para recordar: bajo este icono se encuentran definiciones importantes que refuerzan lo explicado en la página.
	Comentarios o información extra: este ícono ayuda a comprender mejor o ampliar el texto principal
	Contenidos interactivos: indica la presencia de contenidos extra en la Web.

Registro en la Web de apoyo

Para tener acceso al material de la página Web de apoyo del libro:

1. Ir a la página http://virtual.alfaomega.com.mx
2. Registrarse como usuario del sitio y propietario del libro.
3. Ingresar al apartado de inscripción de libros y registrar la siguiente clave de acceso

4. Para navegar en la plataforma virtual de recursos del libro, usar los nombres de Usuario y Contraseña definidos en el punto número dos. El acceso a estos recursos es limitado. Si quiere un número extra de accesos, escriba a webmaster@alfaomega.com.mx

Estimado profesor: Si desea acceder a los contenidos exclusivos para docentes, por favor contacte al representante de la editorial que lo suele visitar o escribanos a:

webmaster@alfaomega.com.mx

Prefacio

Usted tiene en sus manos un libro que contiene la metodología de la programación de computadoras conocida como programación estructurada; y su evolución al paradigma conocido como programación orientada a objetos. La metodología se presenta en dos fases.

En la primera fase, la metodología está basada en el uso de un pseudolenguaje llamado seudocódigo integrado con la técnica diseño descendente (Top Down Design). Dicha metodología permite diseñar programas bien estructurados, bien documentados eficaces, eficientes y fáciles de darles mantenimiento. El método, ha sido plasmado en forma que conduce el proceso enseñanza-aprendizaje de la programación de computadoras en forma didáctica, simple, completa, consistente y práctica con una gran cantidad y variedad de ejercicios cuidadosamente seleccionados y probados en clase por el autor; y, que va desde un nivel de principiante, pasando por intermedio y deja las bases para el nivel avanzado.

Lo relevante de este método es que enseña a programar computadoras utilizando un pseudolenguaje, es decir, sin utilizar la computadora. Esto permite desarrollar las capacidades mentales que una persona debe tener para programar computadoras y sienta las bases de disciplina y buena estructura. Este enfoque se le dificulta a mucha gente, sin embargo, hay que enfrentarlo, porque siendo la programación una actividad intelectual que requiere mucha creatividad, capacidad de abstracción, de análisis y de síntesis; éstas, no se pueden desarrollar operando un lenguaje en la computadora, sino ejercitando la mente en forma apropiada.

Se ha logrado una metodología de la programación que contiene en forma natural los conceptos, estructuras, filosofía y postulados de la programación estructurada, rescatando las bases de la programación en las que se sustenta, y, que han hecho posible los nuevos avances en la programación como lo es la programación orientada a objetos.

En la segunda fase, se presenta la evolución de la programación estructurada a la programación orientada a objetos. Esta metodología es el resultado de la integración y adaptación de varias técnicas, como son; los conceptos y estructuras de la programación orientada a objetos: objetos, clases, encapsulación; el diagrama de clases de UML (Unified Modeling Language, desarrollado por G. Booch, I. Jacobson y J. Rumbaugh); la arquitectura modelo-vista-controlador; algunos conceptos introducidos por el lenguaje Java; y los conceptos y bases lógicas de la programación estructurada en seudocódigo, que se tratan en la primera fase de este libro.

En la actualidad se ha observado que mucha gente desea aprender la programación orientada a objetos dejando de lado la programación estructurada, esto es un error, porque la programación estructurada es la base de la programación orientada a objetos; y, si deseamos aprender la programación orientada a objetos, primero debemos dominar la programación estructurada.

Haciendo un poco de historia

En los años 50 surgió la programación de computadoras con lenguajes de alto nivel; las estructuras de control que manejaban estos primeros lenguajes (como por ejemplo FORTRAN) eran muy limitadas. Poco a poco se fué ampliando y diversificando el uso de la computadora gracias al desarrollo tecnológico y la aparición de otros lenguajes como COBOL , PL/1, etcétera; al aumentar en tamaño y complejidad las aplicaciones que se desarrollaban, empezaron a presentarse muchos problemas en el desarrollo y mantenimiento de los programas, provocados por su inadecuada estructura y documentación. Por otro lado, el proceso enseñanza-aprendizaje de la programación de computadoras era una actividad extremadamente difícil. En estos tiempos los lenguajes de programación sólo soportaban la estructura de repetición FOR (DO en FORTRAN). A raíz de esto, se gestó una revolución en la programación y se añadieron las estructuras de control DO-UNTIL y DOWHILE; además se demostró la importancia de que los programas fueran estructurados en pequeños módulos. Con la inclusión de estos nuevos conceptos

La programación de computadoras se transformó en PROGRAMACION ESTRUCTURADA. Entre los padres de la programación estructurada destacan Dijkstra, Mills, Hoare, Kernigan, Plauger, Yourdon, Warnier, Searson, Wirth, entre otros. Los postulados de la Programación Estructurada son:

- Toda persona debe aprender a programar utilizando un seudolenguaje estructurado.

- Se deben usar sólo las estructuras de control estructuradas: SEQUENCE, IF-THEN, IF-THEN-ELSE, CASE, DO-UNTIL, FOR, DOWHILE.
- Los programas deben ser organizados en pequeños módulos.
- Se debe utilizar el lenguaje Pascal como prototipo para el aprendizaje de estos conceptos, es decir, como primer lenguaje.
- Donde se aplique la programación se debe hacer énfasis en el diseño de los programas antes de codificar.
- Seguir un estilo que haga más entendible el algoritmo (y el programa), como usar nombres de variables apropiados, dejar sangría para denotar subordinación, entre otros.

Problemática de la enseñanza de la programación en los 80 y 90

En los 90, todo mundo quería hacer y hablar sobre programación orientada a objetos, sin embargo, se observó que en realidad la programación estructurada no penetró en las instituciones de educación. En la mayor parte de las escuelas, universidades e institutos donde se enseña la programación, se incurria en alguno de estos errores:

1. Enseñaban a programar directamente con algún lenguaje; aunque éste fuera el Pascal no es bueno enseñar la lógica directamente con un lenguaje por el doble problema que significa aprender la lógica de programación y la sintaxis del lenguaje al mismo tiempo.
2. Enseñaban la lógica con diagramas de flujo; los cuales resultan obsoletos porque no soportan todas las estructuras de control de la programación estructurada en forma natural.
3. Enseñaban la lógica en pseudocódigos que utilizan las estructuras lógicas de control en forma castellanizada, es decir, en lugar de IF-THEN-ELSE utilizan SI-ENTONCES-SINO, en lugar de FOR utilizan PARA, en lugar de DOWHILE utilizan MIENTRAS; he usado estos conceptos y he comprobado que no es buena idea. Las estructuras de la programación estructurada tienen sus palabras reservadas originales y no deben ser cambiadas, porque causan mucha confusión al estudiar los lenguajes de programación.
4. Se utilizaba C (o C++) como primer lenguaje; que permite la programación estructurada pero no es un lenguaje estructurado en forma natural, además de que no es un lenguaje orientado para principiantes.
5. En las materias en las que se aplica la programación (estructura de datos, base de datos, etcétera), los maestros hacen énfasis en el código del programa y no en el diseño del mismo; lo cual es esencial en la programación estructurada.

Otro gran problema con la enseñanza de la programación estructurada es el gran ruido que ha causado la programación orientada a objetos; todo mundo habla de ella y se ha vendido la idea de que ha venido a reemplazar a la programación estructurada; y que debe enseñarse la programación desde un inicio con lenguajes orientados a objetos como Java, C++, etcétera. Sin embargo, la programación orientada a objetos es una herramienta que se le ha añadido a la programación estructurada, es decir, no ha venido a reemplazarla, sino que se basa en la estructurada y va más allá, pero se nutre de ésta. Porque las estructuras de la programación estructurada: Tipos de datos; entero, real, cadena, carácter, arreglos, registros y archivos; Estructuras de control; secuenciación, if-then, if-then-else, case. do-while, for, while; módulos y funciones; también son los elementos básicos de la programación orientada a objetos. Así que los conceptos y filosofía de la programación estructurada están hoy más vigentes que nunca.

Ahora en que se nos dice que debemos correr a través del uso de la programación orientada a objetos, visual, etcétera, yo les digo ¡¡¡ALTO!!!, primero tenemos que reconocer que no hemos aprendido a caminar a través de la asimilación adecuada de la programación estructurada: ¡Debemos primero aprender a caminar para poder aprender a correr!

Este autor ha desarrollado una metodología que viene a coadyuvar en la solución de la problemática antes referida, la cual ha sido publicada en el libro intitulado Programación estructurada un enfoque algorítmico en su primera y segunda edición. Ahora se presenta la tercera edición.

Mejoras de esta tercera edición

Las mejoras realizadas en esta tercera edición se dividen en tres partes:

Primera:

En la metodología de la programación presentada en la segunda edición, las estructuras lógicas de control: Secuenciación, IF THEN, IF THEN ELSE, CASE, DO UNTIL, FOR y DOWHILE, se utilizan en el formato en que originalmente fueron inventadas. Sin embardo, en lenguaje C la estructura CASE, se implementa como switch; la estructura DO UNTIL se implementa como do…while; y la estructura DOWHILE se implementa como while. Considerando que las estructuras lógicas del lenguaje C son la base de C++, Java y de la mayoría de los lenguajes actuales, es que la forma en que las implementa el lenguaje C, se han convertido prácticamente en el estándar de las estructuras de control. Esto, en cierta medida ha debilitado y desactualizado a la metodología de la segunda edición.

En esta tercera edición, se le han hecho ajustes a la metodología para actualizarla y fortalecerla, a continuación se describen:

1. Los identificadores como nombres de variables, funciones, etcétera; ahora se manejan con un estilo más actualizado.
2. La estructura CASE del capítulo cuatro de la segunda edición, ahora se implementa como switch.
3. La estructura DO UNTIL del capítulo cinco de la segunda edición, ahora se implementa como do… while.
4. La estructura DOWHILE del capítulo siete de la segunda edición, ahora se implementa como while.

Los cambios anteriores, implicaron hacer ajustes en todos los capítulos, de tal manera que la metodología ha sido reestructurada para quedar acorde con la forma en que los lenguajes actuales implementan las estructuras de control. Logrando una metodología de la programación estructurada más actualizada, completa y consistente.

Segunda:

Se han agregado cuatro capítulos en los que se presenta la programación orientada a objetos, conjuntando lo que es pertinente de los primeros siete capítulos, con los conceptos de objetos, clases y encapsulación, planteando una evolución natural de la programación estructurada a la orientada a objetos.

Cabe aclarar que la programación orientada a objetos se trata de una forma introductoria y hasta un nivel medio, si desea mayor profundidad, puede hacerlo estudiando el libro Metodología de la programación orientada a objetos, de este mismo autor y publicado por esta misma editorial, que actualmente está en su primera edición, y en un futuro se publicará la segunda edición; en el cual se presenta la metodología de la programación orientada a objetos en forma más completa y con mayor profundidad.

Tercera:

Aunque el propósito de este libro no es estudiar ningún lenguaje de programación; en la dirección http://virtual.alfaomega.com.mx que es una web de ayuda que la editorial Alfaomega ha puesto a su disposición, podrá encontrar los programas correspondientes de los algoritmos del libro, codificados en lenguaje C los de los capítulos 3 al 9, que corresponde a la programación estructurada; y en Java, los de los capítulos del 3 al 7 y del 11 al 13. Cabe aclarar, que no obstante que Java es un lenguaje orientado a objetos, es posible programar en Java los algoritmos desarrollados mediante la programación estructurada. Los de los capítulos 8 y 9 no se presentan en Java porque va más allá del propósito y nivel de este libro. Además, en la web del libro, se encuentran ejercicios resueltos de los capítulos 3 al 8 y del 11 al 13.

Problemática actual y reivindicación de la programación estructurada

Hasta la aparición del lenguaje Java, la enseñanza de la programación se realizaba primero usando alguna metodología de la programación estructurada y luego el lenguaje C o C++ en la parte estructurada; después se enseñaba la programación orientada a objetos en C++. Cuando aparece Java, este empieza a enseñarse como un segundo o tercer lenguaje; ya sea despúes de C o después de C++. Sin embargo, en los últimos años, en muchas instituciones de educación adoptaron Java como primer lenguaje, eliminando la programación estructurada, el lenguaje C y C++.

Actualmente, cada día son más las instituciones que han comprobado que enseñar Java como primer lenguaje no ha sido una buena idea; y se han dado cuenta que es mejor enseñar primero la programación estructurada implementada en C o C++, y después de esas bases, enseñar la programación orientada a objetos en Java. Esto, está llevando a reivindicar al paradigma de la programación estructurada, como lo que es, la base en la que se sustenta la programación orientada a objetos, así que, la programación estructurada esta hoy más vigente que nunca. Es por ello que he decidido revitalizar esta metodología y además, añadirle su evolución natural hacia la programación orientada a objetos. Lo cual ha permitido desarrollar esta nueva edición, logrando una metodología más actualizada, completa y consistente.

Organización del libro

El material de la presente obra esta dividido en trece capítulos y cinco apéndices. En los capítulos del 1 al 9, es lo que se considera como la primera parte del libro, aquí se presenta la metodología de la programación estructurada.

En el capítulo uno se presenta una introducción a la programación. En el capítulo dos se presentan los elementos para solucionar problemas en seudocódigo, como son tipos de datos, identificadores, variables, operaciones aritméticas, lectura y escritura de datos, etcétera. En el capítulo tres se estudia la estructura de control secuenciación, su definición, estructura de un algoritmo y algunos ejemplos. El capítulo cuatro trata la selección simple, doble y múltiple, su definición, formato en seudocódigo y utilización con ejercicios resueltos y propuestos.

En el capítulo cinco se explican las estructuras de repetición do…while, for y while; su definición, formato, uso, ejercicios resueltos y propuestos.

En el capítulo seis se presentan los arreglos unidimensionales, bidimensionales, tridimensionales y tetradimensionales; su definición, formato, manipulación, uso, ejercicios resueltos y propuestos.

En el capítulo siete se trata la modularización de programas a través del diseño descendente; funciones que no regresan valor (módulos), funciones que regresan valor (funciones definidas por el usuario), el paso de parámetros entre funciones y algunas otras funciones estándar.

El capítulo ocho trata lo referente a registros y archivos, donde se presenta definiciones, organización de archivos, como manejarlos en seudocódigo, proceso de un archivo secuencial, proceso de un archivo directo, obtención de reportes, ejercicios resueltos y propuestos.

El capítulo nueve, como si fuera un apéndice, en el que se presentan una serie de temas sueltos: Clasificación (ordenación) de datos, definición de tipos de datos, apuntadores, recursividad, ejecución de otros programas ejecutables y graficación, donde cada tema puede estudiarse en el momento en que se considere necesario. Y, permiten que la metodología de la programación estructurada se utilice en temas más avanzados de programación.

A partir del capitulo diez se explica lo que se considera como la segunda parte del libro, es decir, se presenta la metodología integrando algunos conceptos y estructuras de la programación orientada a objetos: objetos, clases, encapsulación; el diagrama de clases de UML (Unified Modeling Language); la arquitectura modelo-vista-controlador; algunos conceptos introducidos por el lenguaje Java; y los conceptos y bases lógicas de la programación estructurada en seudocódigo estudiados en los capítulos del 1 al 7. Dicha metodología permite diseñar programas o algoritmos orientados a objetos, bien estructurados, bien documentados eficaces, eficientes y fáciles de darles mantenimiento.

En el capítulo diez, se estudian los conceptos de objetos, clases, métodos y encapsulación, y se explica la forma de cómo utilizarlos para diseñar el diagrama de clases. Asimismo se involucra el uso de la arquitectura modelo-vista-controlador.

En el capítulo once, se estudia cómo diseñar la lógica de cada una de las clases usando seudocódigo, incorporando en esta técnica los conceptos de clases, objetos y encapsulación. Se presenta el diseño de algoritmos orientados a objetos aplicando la estructura de secuenciación con ejemplos que se estudiaron en el capítulo tres.

En el capítulo doce se presenta el diseño de algoritmos orientados a objetos aplicando las estructuras de selección if then else, if then y switch, con ejemplos que se estudiaron en el capítulo cuatro.

En el capítulo trece se presenta el diseño de algoritmos orientados a objetos aplicando las estructuras de repetición do…while, for y while, con ejemplos que se estudiaron en el capítulo cinco.

En el apéndice A se presentan algunos algoritmos sin usar etiquetas. En el apéndice B se muestran algunos de los algoritmos en diagramas de flujo, de tal manera que al estudiar los algoritmos en seudocódigo, podrá auxiliarse de los diagramas de flujo correspondientes, y así facilitar su comprensión. En el apéndice C se explica otro método alternativo de diseño de programas usando los Diagramas Warnier. En el apéndice D se expone otro método para el diseño de programas usando los Diagramas Chapin (Nassi-Schneiderman). Y en el apéndice E se presentan algunos algoritmos usando una versión de seudocódigo castellanizado o español estructurado. A quien va dirigido

Este trabajo requiere que el lector tenga conocimientos previos sobre conceptos generales relacionados con las computadoras, en caso de no ser así, se recomienda la lectura de algún libro que sirva como introducción a las computadoras, a las tecnologías de la información o a la informática.

El libro puede ser utilizado como texto o consulta en materias de fundamentos de programación, metodología de la programación, programación de computadoras y similares, que se cursan en los primeros dos o tres semestres de carreras como Informática, Informática administrativa, Computación, Ciencias de la computación, Sistemas Computacionales, Sistemas de Información, Ingeniería Industrial y de Sistemas, Ingeniería Mecatrónica, Ingeniería Electrónica, Ingeniería Eléctrica, entre otras; también se puede usar en preparatorias, bachilleratos y carreras de nivel técnico.

Advertencia didáctica

Se ha tomado el ejemplo de calcular sueldo(s) de empleado(s) -pago de sueldo o nómina- como un problema pivote para facilitar la explicación de las estructuras que integran la metodología. Esto quiere decir, que al estudiar cada elemento o estructura, el primer problema que se plantea es el de pago de sueldo, primero de una manera muy simple, y luego, con una variante apropiada para la aplicación de lo que se está explicando en cada momento. Esto es con fines didácticos, ya que es una situación fácilmente entendible, que permite al estudiante (lector) familiarizarse con ella porque se trata desde el inicio y durante todo el libro; y así podemos dedicar todo nuestro esfuerzo mental al aspecto lógico de la metodología. Es decir, que los problemas no deben ser muy complicados, para dirigir todo nuestro esfuerzo mental a comprender la lógica de la metodología y no a entender problemas innecesariamente complejos.

En consecuencia, es probable que el lector, perciba que este es un libro con una orientación administrativa, sin embargo, después de explicarle cada estructura con el ejemplo antes referido; en los ejercicios resueltos, se presenta la aplicación de la estructura con otros tipos de problemas, dándole a la metodología un enfoque de aplicación general, es decir, tanto para el área administrativa, como para la ingeniería.

Resumiendo

Aprender a programar no es fácil ni rápido; es un proceso que debe iniciar con el desarrollo de la lógica usando un pseudolenguaje de diseño de programas o algoritmos. Con el estudio de la metodología y fundamentos de programación que le presento en este libro, el estudiante aprenderá la lógica de la programación estructurada y una introducción a la programación orientada a objetos sin estar "casado" con ningún lenguaje; y de aquí en adelante podrá aprender y comprender cualquier lenguaje estructurado u orientado a objetos como C, C++, Java, C#, UML, etcétera.

1

Introducción

Contenido

Objetivos del capítulo

- Repasar conceptos generales de computación.
- Comprender los fundamentos del funcionamiento de la computadora y sus componentes.
- Comprender el concepto de programa.
- Entender las características principales de un lenguaje de programación.
- Entender cuáles son las características de un buen programa.
- Conocer los diversos paradigmas de programación.
- Entender cuáles son los pasos que integran el proceso de programación.
- Entender qué es el algoritmo, cuáles son las características que tiene y aplicar los conceptos aprendidos en situaciones de la vida cotidiana.

Introducción

En este primer capítulo el lector efectuará un repaso de los principales conceptos acerca de las computadoras. Se presenta la computadora como una herramienta que se utiliza para resolver problemas en el accionar cotidiano de las empresas, organizaciones o instituciones. Esto se hace mediante el esquema del procesamiento electrónico de datos que es E-P-S (Entrada-Proceso-Salida), es decir: entran datos como materia prima, luego se procesan para transformarlos en información que se emite como salida.

Se estudian también los elementos funcionales básicos que componen una computadora, que son: la unidad central de proceso, la unidad de memoria, la unidad de entrada y la unidad de salida. Cabe aclarar que es deseable que estos conceptos ya los domine el estudiante, o bien, que los estudie en algún otro libro de introducción a la computación, a la informática o tecnologías de la información.

Se expone el concepto de programa, que es un conjunto de instrucciones que guían a la computadora para realizar alguna actividad o resolver algún problema. También se detalla que el programa se compone de estructuras de datos, operaciones primitivas elementales y estructuras de control.

Se explica que un lenguaje de programación es el medio a través del cual le comunicamos a la computadora la secuencia de instrucciones que debe ejecutar para llevar a cabo actividades, tareas o para solucionar problemas. Además se expone que todo lenguaje está compuesto por un alfabeto, un vocabulario y una gramática, luego se revisa el concepto de lo que es la programación y se enumeran las características de un buen programa.

Se presenta la evolución de los paradigmas de programación, que inicia con las primeras estructuras que se inventaron cuando aparece la programación tradicional, luego, sobre esas bases se gestó la programación estructurada, para dar lugar a la programación modular, enseguida se agrega la programación con abstracción de datos, para llegar al desarrollo de la programación orientada a objetos.

Se detallan los pasosque integran el proceso de programación:definición del problema, análisis del problema, diseño del programa, codificación del programa, implantación del programa y mantenimiento del programa.

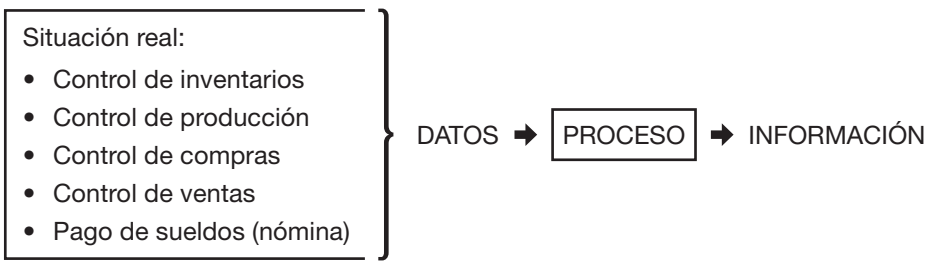
Se define que el algoritmo es una secuencia de pasos que llevan a la solución de un problema o a la ejecución de una tarea o actividad. Se plantea que los pasos del algoritmo deben tener las siguientes características: ser simples, claros, precisos, exactos; tener un orden lógico; tener un principio y un fin.

En el siguiente capítulo se estudian los elementos para solucionar problemas en seudocódigo.

1.1 Conceptos generales

La computadora

La computadora es una herramienta que se utiliza para representar cualquier situación de la realidad en forma de datos, los cuales se procesan después para generar información, esquemáticamente:



Esto quiere decir que toda situación que pueda ser abstraída y representada en forma de datos, puede manejarse mediante la computadora, porque el esquema del proceso de datos es E-P-S (Entrada-Proceso-Salida), es decir, datos entran como materia prima, se procesan para transformarlos en la información que se da como salida. Por ejemplo, en una situación de pago de sueldos (nómina), un trabajador puede representarse mediante los datos: Nombre del empleado, número de horas trabajadas y cuota por hora. El sueldo se obtiene multiplicando número de horas trabajadas por la cuota por hora. Y se da como salida el nombre y el sueldo. Tanto los datos como el procedimiento necesario para generar la información, se suministran a la computadora en forma de un programa constituido por instrucciones. La computadora interpreta y ejecuta las instrucciones del programa de acuerdo con ciertas reglas de sintaxis que conforman el lenguaje de programación, mediante el cual podemos comunicarle lo que debe hacer.

Los elementos básicos que componen una computadora son la unidad central de proceso, la unidad de memoria, la unidad de entrada y la unidad de salida.

La *unidad central de proceso* es el "cerebro" que controla el funcionamiento de los componentes y ejecuta las operaciones aritméticas y lógicas. Las operaciones del procesador central son muy simples, pero ejecutadas a una velocidad muy alta –del orden de millones por segundo– permiten la ejecución de tareas simples o complejas.

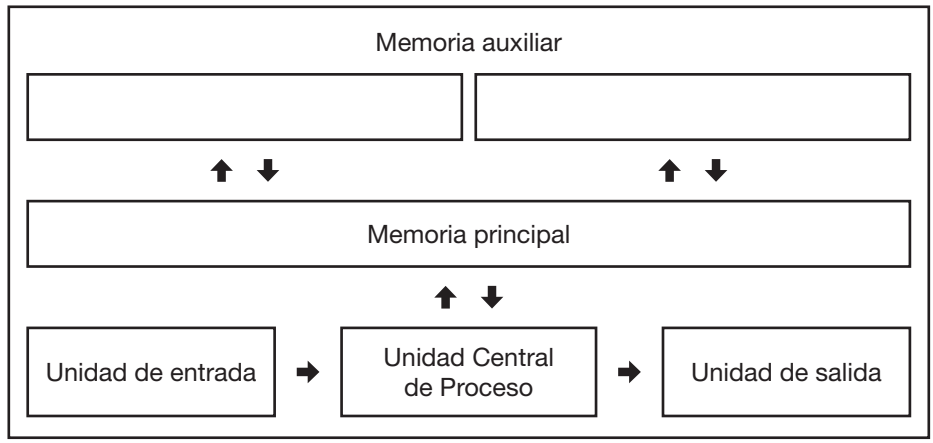


Fig. 1.1 Diagrama que describe la organización funcional de una computadora.

La *memoria* se utiliza para almacenar los datos, y a éstos se les aplican las operaciones del procesador. Existen dos tipos de memoria: la principal y la auxiliar. La memoria principal permite al procesador extraer y almacenar datos a una velocidad comparable con la propia. Cada operación propicia por lo menos un acceso a la memoria. Para que el procesador pueda avanzar de una operación a la siguiente sin retraso, el programa de instrucciones se almacena en esta memoria; en otras palabras, la memoria principal guarda tanto las instrucciones como los datos sobre los que actúa el procesador central. La memoria principal está limitada por su alto costo; debido a esto no es posible conservar en ella grandes cantidades de datos e instrucciones y, en consecuencia, sólo se usa para guardar lo que el procesador esté utilizando por el momento. Además, tiene la característica de que no permite almacenar datos permanentemente, pues si se apaga la computadora se pierde lo que haya en memoria. Por tales razones, las computadoras están equipadas con memorias auxiliares para almacenamiento masivo y permanente de datos, tales como discos magnéticos fijos, disquetes (discos flexibles) magnéticos removibles, discos compactos, cintas magnéticas, entre otros. Estos dispositivos tienen más capacidad que la memoria principal, pero son más lentos. Los datos pueden almacenarse en ellos de manera permanente, es decir, pueden guardarse para usos posteriores.

Fig. 1.2 Computadora personal.

La *unidad de entrada* se utiliza para introducir datos del exterior en la memoria de la computadora a través de dispositivos periféricos de entrada como teclados de terminales, ratón (mouse), discos, módem, lector de código de barras, etc. Esta unidad realiza automáticamente la traducción de símbolos inteligibles para la gente, en símbolos que la máquina pueda manejar.

Fig. 1.3 Pantalla o monitor, dispositivo periférico de salida estándar.

La *unidad de salida* permite transferir datos de la memoria al exterior, a través de dispositivos periféricos de salida como impresoras, pantallas de video, módem, etc. Esta unidad realiza automáticamente la traducción de símbolos que puede manejar la máquina, en símbolos inteligibles para la gente.

Fig. 1.4 Teclado, dispositivo periférico de entrada estándar.

Fig. 1.5 Impresora, dispositivo periférico de salida impresa.

Fig. 1.6 Mouse, dispositivo periférico de entrada.

Fig. 1.7 Unidad de DVD, dispositivo periférico de entrada/salida.

El programa

Un programa es un conjunto de instrucciones que guían a la computadora para realizar alguna actividad o resolver algún problema; en el programa se ejecutan diferentes acciones de acuerdo con los datos que se estén procesando. El programa debe incluir instrucciones para las acciones que deban ejecutarse sobre cada uno de los tipos de datos admitidos, además instrucciones que identifiquen los datos erróneos y recuperarse ante la aparición de éstos.

Cuando se ejecuta un programa con un tipo de datos específico, es probable que no se ejecuten todas las instrucciones, sino sólo las que sean pertinentes a los datos en cuestión. Un programa se compone de estructuras de datos, operaciones primitivas elementales y estructuras de control, como se muestra a continuación:

Programa = Estructuras de datos
+ Operaciones primitivas elementales
+ Estructuras de control

Estructuras de datos. Son las formas de representación interna de la computadora. Los hechos reales, representados en forma de datos, pueden estar organizados de diferentes maneras llamadas estructuras de datos. Por ejemplo el nombre, las horas trabajadas y el sueldo por hora son los datos mediante los cuales se representa un empleado en una situación de pago de sueldos (nómina).

Operaciones primitivas elementales. Son las acciones básicas que la computadora "sabe" hacer, y que se ejecutan sobre los datos para darles entrada, transformarlos y darles salida convertidos en información. Por ejemplo, el sueldo de un empleado se calcula multiplicando las horas trabajadas por la cuota horaria.

Estructuras de control. Son las formas lógicas de funcionamiento de la computadora mediante las que se dirige el orden en que deben ejecutarse las instrucciones del programa. Las estructuras de control son: La secuenciación, que es la capacidad de ejecutar instrucciones secuenciales una tras otra. La selección es la capacidad de escoger o seleccionar si algo se ejecuta o no, optar por una de dos o más alternativas, y la repetición, que es la capacidad de realizar

en más de una ocasión (es decir, varias veces) una acción o conjunto de acciones; por ejemplo calcular el sueldo a un empleado, pero repitiendo el cálculo *n* veces para *n* empleados.

El lenguaje de programación

Un lenguaje de programación es el medio a través del cual le comunicamos a la computadora la secuencia de instrucciones que debe ejecutar para llevar a cabo actividades, tareas o solución de problemas. Todo lenguaje permite el manejo de los tres elementos que componen un programa, a saber: estructuras de datos, operaciones primitivas elementales y estructuras de control.

Recordemos que mediante un programa podemos representar en forma de datos cualquier situación de nuestra realidad, a los datos se les da entrada a la computadora mediante dispositivos de entrada como teclado, lector óptico de caracteres, ratón, etc.; una vez que los datos están en la computadora, se procesan para convertirlos en información, la cual será emitida hacia el exterior de la computadora mediante dispositivos de salida como son la pantalla, impresora, etcétera.

• Características de los lenguajes de programación

Todo lenguaje está compuesto por un alfabeto, un vocabulario y una gramática. A continuación se describen estos componentes.

Alfabeto o conjunto de caracteres. Es el conjunto de elementos estructurales del lenguaje:

a) Caracteres alfabéticos (letras minúsculas y mayúsculas).

b) Caracteres numéricos (dígitos del 0 al 9).

c) Caracteres especiales (símbolos especiales tales como [.], [,], [:], [;], [$], [#], [/] y muchos otros).

Vocabulario o léxico. Es el conjunto de palabras válidas o reservadas en el lenguaje. Por ejemplo, las palabras program, begin, end, if, then, else, integer, real, string, repeat, for, while, char, procedure, function, byte, boolean tienen un significado predeterminado en el lenguaje Turbo Pascal, es decir, son las palabras reservadas del lenguaje Turbo Pascal. Así, cada lenguaje tiene sus propias palabras reservadas.

Gramática. Es el conjunto de lineamientos que se deben seguir para construir frases, oraciones o instrucciones. Mediante la gramática o sintaxis logramos transmitirle a la computadora lo que deseamos. Por ejemplo, para leer datos debemos seguir cierto lineamiento, también para imprimir, etcétera.

La programación

Generalmente se consideran sinónimos los conceptos *programación* y *codificación*, lo cual constituye un error. Debemos tener presente que la finalidad de un programa es realizar algún proceso sobre ciertos datos para obtener ciertos resultados. La preparación de un programa implica aspectos tales como: ¿para qué sirve el proceso que se desea representar?, ¿qué datos usará, qué resultados producirá y cómo se realizará el proceso sobre los datos para obtener los resul-

tados esperados? Una vez identificado lo anterior se procede a diseñar la manera de cómo la computadora deberá hacerlo, tomando en cuenta su estructura interna y su funcionamiento. Hasta ese momento se tiene representada la solución de una manera convencional (algoritmo), pero enseguida se procede a codificar el programa que solucionará el problema, utilizando un lenguaje de programación.

• Características de un buen programa

Un programa bien escrito debe tener ciertas características básicas que le permitan operar correctamente; las principales serían las siguientes:

Operatividad. Lo mínimo que debe hacer un programa es funcionar; es decir, producir los resultados esperados independientemente de cualquier otra característica.

Legibilidad. Un programa puede hacerse más legible dándole cierto formato al código, utilizando el sangrado (indentación) para reflejar las estructuras de control del programa, e insertando espacios o tabuladores. Es conveniente diseñar reglas propias para darle uniformidad a todos los programas.

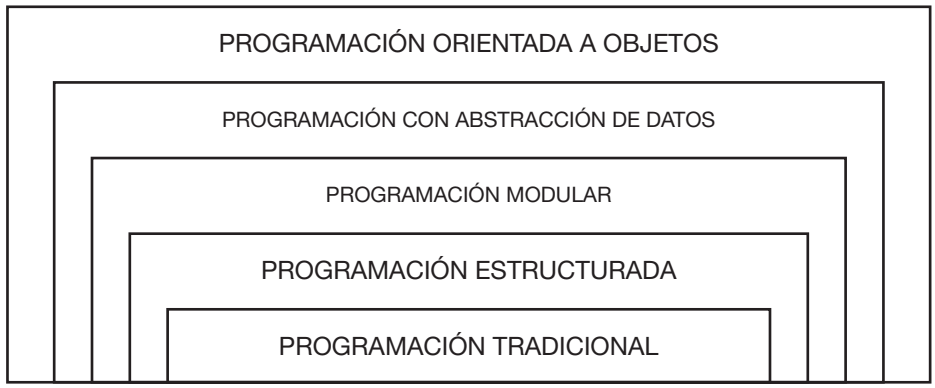
Transportabilidad. Un programa transportable es el que puede ejecutarse en otro entorno sin hacerle modificaciones importantes. Mientras menos modificaciones se hagan será más transportable, así que es conveniente no utilizar características especiales del hardware ni "facilidades" especiales del software.

Claridad. Esta característica se refiere a la facilidad con la que el texto del programa comunica las ideas subyacentes. El programa debe indicar claramente lo que el programador desea. Una buena programación es similar a la elaboración de un documento legal; por ejemplo, conviene utilizar nombres adecuados para los identificadores, hacer comentarios correctos, claros y concisos, etcétera.

Modularidad. Dividir el programa en un número de módulos pequeños y fáciles de comprender puede ser la contribución más importante a la calidad del mismo. Cada módulo debe realizar sólo una tarea específica, y no más. Los módulos tienen la virtud de minimizar la cantidad de código que el programador debe comprender a la vez, además de que permiten la reutilización de código.

1.2 Evolución de los paradigmas de programación

Desde que la programación de computadoras apareció como tal, la forma, el paradigma o modelo que se usa ha evolucionado constantemente. Sin embargo, las bases de la programación no han cambiado, simplemente se han ido añadiendo nuevos conceptos y nuevas estructuras. Todo inicia con las primeras estructuras que se inventaron cuando aparece la programación tradicional, luego, sobre esas bases se gestó la programación estructurada, para dar lugar a la programación modular, enseguida se agrega la programación con abstracción de datos, para llegar al desarrollo de la programación orientada a objetos. En la siguiente figura se esquematiza la evolución de los paradigmas de programación.



Características de los paradigmas de programación

La evolución de los paradigmas de programación ha tenido tres grandes pasos, el primer gran paso es cuando la programación aparece como tal, es lo que se está esquematizando como la programación tradicional. Luego se dio un segundo gran paso y surge la programación estructurada. Después se experimentó un pequeño paso que dio lugar a la programación modular. Enseguida vino otro pequeño paso que permitió el surgimiento de la programación con abstracción de datos. Luego vino el tercer gran paso que es la aparición de la programación orientada a objetos.

Tradicional

La programación tradicional tuvo sus inicios a principios de la década de 1950. Los lenguajes de programación que se utilizaban eran los predecesores de FORTRAN, COBOL y BASIC. Las estructuras lógicas de control que se utilizaban eran: la secuenciación, IF-THEN, IF-THEN-ELSE y DO (en la actualidad conocido como FOR). La técnica de diseño de programas utilizada eran los diagramas de flujo.

La arquitectura de un programa consistía de un solo módulo, como se muestra a continuación:

Este módulo estaba formado por una secuencia ordenada de instrucciones:

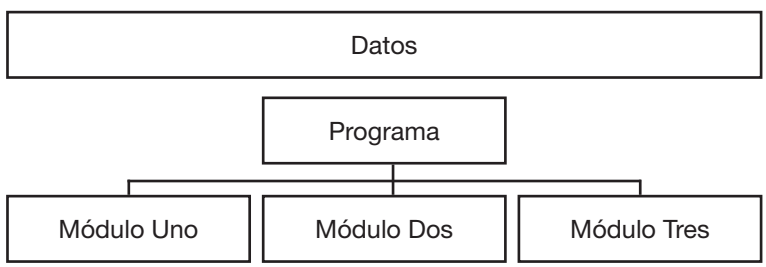
Instrucción 1
Instrucción 2
Instrucción 3
Instrucción 4
Instrucción 5
Instrucción 6

Instrucción 7
Instrucción 8
- - -
- - -
- - -
Instrucción N

Estructurada

La programación estructurada tuvo sus inicios a mediados de la década de 1960. Los lenguajes de programación que se utilizaban eran PASCAL, COBOL estructurado, BASIC estructurado, FORTRAN con estilo estructurado, FORTRAN 90 y Lenguaje C. Las estructuras de control utilizadas eran la secuenciación, IF-THEN, IF-THEN-ELSE, CASE, FOR, DO-UNTIL y DOWHILE. Otras características son: Dividir un programa en módulos y funciones y estilo de programación. Las técnicas de diseño de programas que se utilizaban eran diagramas Warnier, diagramas estructurados, diagramas Chapin, seudocódigo y Top Down Design, entre otras.

La arquitectura de un programa consistía en datos y en un conjunto de módulos jerarquizados, como se muestra a continuación:



Cada módulo estaba formado por un conjunto ordenado de instrucciones:

Módulo Uno	**Módulo Dos**	**Módulo Tres**
Instrucción 1	Instrucción 1	Instrucción 1
Instrucción 2	Instrucción 2	Instrucción 2
Instrucción 3	Instrucción 3	Instrucción 3
- - -	- - -	- - -
- - -	- - -	- - -
- - -	- - -	- - -
Instrucción N	Instrucción N	Instrucción N

Al diseñar la solución en módulos, la programación estructurada permite solucionar problemas más grandes y complejos, de una mejor forma que como se hacía anteriormente.

Modular

La programación modular tuvo sus inicios a fines de la década de 1970 y principios de la de 1980. El lenguaje de programación que se utilizó fue MODULA 2. Emerge el concepto de encapsulación, que en un módulo o paquete se encapsulan los datos y las funciones que los manipulan.

Con abstracción de datos

La programación con abstracción de datos se generó en la década de 1980. El lenguaje de programación que se utilizó fue ADA. Con éste emerge el concepto de Tipos Abstractos de Datos (TAD).

Orientada a objetos

La programación orientada a objetos, aunque el concepto se generó muchos años antes, es a finales de la década de 1980 y principios de la de 1990 que se pone en boga, la caracterizan los conceptos objetos, clases, encapsulación, herencia y polimorfismo. Los principales lenguajes de programación que se utilizan son C++, Java y C#. Las técnicas de diseño que se utilizan son Booch, Rumbaugh, Jacobson, Yourdon y UML (Unified Modeling Language), entre otras.

La arquitectura de un programa consiste en un conjunto de objetos, y cada objeto se compone por datos y un conjunto de métodos, donde cada método está formado por un conjunto de instrucciones, como se muestra a continuación: