INTRODUCCIÓN

El proceso de resolución de un problema con una computadora conduce a la escritura de un programa y a su ejecución en la misma. Aunque el proceso de diseñar programas es, esencialmente, un proceso creativo, se puede considerar una serie de fases o pasos comunes, que generalmente deben seguir todos los programadores

En esta sección trataremos brevemente estas fases y conceptualizaremos el concepto de algoritmo y sus características.

FASES EN LA RESOLUCIÓN DE PROBLEMAS CON COMPUTADORA

Las fases de resolución de un problema con computadora son:

- Análisis del problema.
- Diseño del algoritmo.
- Codificación.
- Compilación y ejecución.
- Verificación.
- Depuración.
- Mantenimiento.
- Documentación.

Las características más sobresalientes de la resolución de problemas son:

1. Análisis. El problema se analiza teniendo presente la **especificación de los requisitos** dados por el cliente de la empresa o por la persona que encarga el programa.
2. Diseño. Una vez analizado el problema, se diseña una solución que conducirá a un **algoritmo** que resuelva el problema.
3. Codificación (implementación). La solución se escribe en la sintaxis del lenguaje de alto nivel (por ejemplo, Java, C#, Processing, et) y se obtiene un/os archivos que representan el programa y que se denomina genéricamente **código fuente** que se traducirán al lenguaje de las computadoras en la siguiente fase.
4. Ejecución, verificación y depuración. El programa se ejecuta, se comprueba rigurosamente y se eliminan todos los errores (denominados “bugs”, en inglés) que puedan aparecer.
5. Mantenimiento. El programa se actualiza y modifica, cada vez que sea necesario, de modo que se cumplan todas las necesidades de cambio de sus usuarios.
6. Documentación. Escritura de las diferentes fases del ciclo de vida del software, esencialmente el análisis, diseño y codificación, unidos a manuales de usuario y de referencia, así como normas para el mantenimiento.

Las dos primeras fases conducen a un diseño detallado escrito en forma de algoritmo. Durante la tercera fase (codificación) se implementa[1] el algoritmo en un código escrito en un lenguaje de programación, reflejando las ideas desarrolladas en las fases de análisis y diseño.

Las fases de compilación y ejecución traducen y ejecutan el programa. En las fases de verificación y depuración el programador busca errores de las etapas anteriores y los elimina. Comprobará que mientras más tiempo se gaste en la fase de análisis y diseño, menos se gastará en la depuración del programa. Por último, se debe realizar la documentación del programa.

[1] Implementar: poner funcionamiento, aplicar métodos, medidas, etc para llevar a cabo algo

ALGORITMO Y METODOLOGÍA DE LA PROGRAMACIÓN

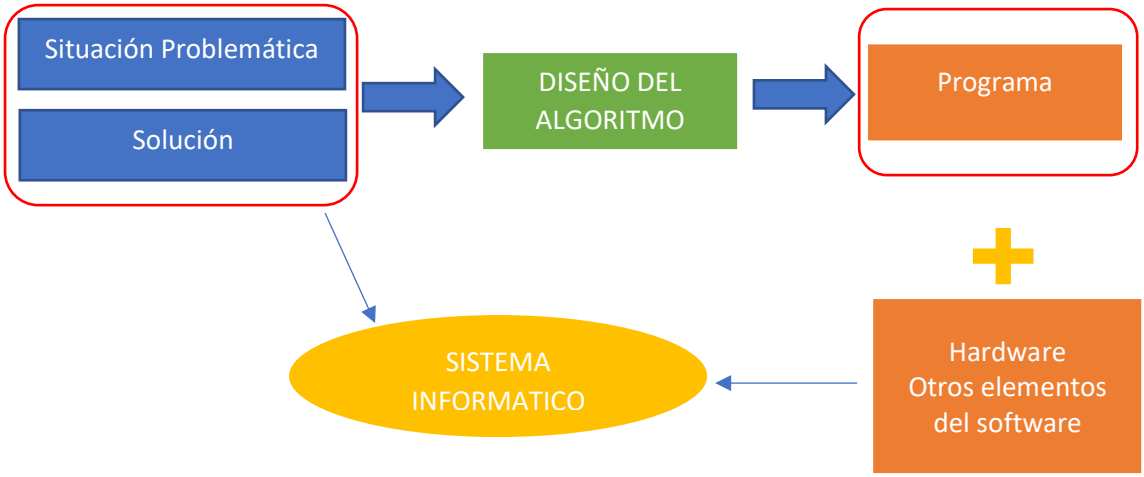
Un algoritmo es un método para resolver un problema mediante una serie de pasos precisos, definidos y finitos. Estas últimas tres palabras, son además las características de un algoritmo:

- Preciso: Indica el orden de realización de cada paso.
- Definido: Si se sigue dos veces, obtiene el mismo resultado cada vez.
- Finito: Tiene fin, un determinado número de pasos.

Un algoritmo debe producir un resultado en un tiempo finito. Los métodos que utilizan algoritmos se denominan **métodos algorítmicos**, en oposición a los métodos que implican algún juicio o interpretación que se denominan **métodos heurísticos**. Los métodos algorítmicos se pueden implementar en computadoras; mientras que las técnicas de inteligencia artificial han hecho posible la implementación del proceso heurístico en computadoras.

El programador es antes que nada una persona que resuelve problemas, por lo que para llegar a ser un programador eficaz se necesita aprender a resolver problemas de un modo riguroso y sistemático; esto es aplicar una **Metodología de la Programación**.

El eje central de esta metodología es el concepto de algoritmo. Así, la resolución de un problema aplicando la metodología de programación exige el diseño de un algoritmo que resuelva el problema propuesto. Esto lo puede observar en la figura 1:



Se observan todos los elementos de un problema representado como un sistema informático:

1. Los datos de entrada (situación problemática) y los datos de salida (la solución) son elementos importantes para poder diseñar un algoritmo.
2. El **DISEÑO DE UN ALGORITMO** hace referencia a una secuencia ordenada de pasos – sin ambigüedades – que conducen al desarrollo del proceso o estrategia. Así, el DISEÑO DEL ALGORITMO incluye tanto la fase de Análisis del Problema como la fase de Diseño del algoritmo. La fase del diseño del algoritmo tiene como objetivo crear un modelo del programa que aún no ha sido codificado.
3. A partir del DISEÑO DEL ALGORITMO, se puede construir el programa, que se corresponde con la fase de codificación o implementación del programa.
4. El programa, junto con la configuración del hardware y los documentos de desarrollo constituyen el software del sistema informático. Así, en este paso, el sistema puede ejecutarse, verificarse y depurarse, con lo cual se corresponde con la fase de Ejecución de la resolución de problemas mediante algoritmos.

Entonces, la idea central de la Metodología de la Programación es que para llegar a la realización de un programa es necesario el diseño previo de un algoritmo, de modo que sin algoritmo no puede existir un programa.

Los algoritmos son independientes tanto del lenguaje de programación en que se expresan como de la computadora que los ejecuta.

En cada problema el algoritmo se puede expresar en un lenguaje diferente de programación y ejecutarse en una computadora distinta; sin embargo, el algoritmo será siempre el mismo. Así, por ejemplo, en una analogía con la vida diaria, una receta de un plato de cocina se puede expresar en español, inglés o francés, pero cualquiera que sea el lenguaje, los pasos para la elaboración del plato se realizarán sin importar el idioma del cocinero.

En la ciencia de la computación y en la programación, los algoritmos son más importantes que los lenguajes de programación o las computadoras.

Un lenguaje de programación es tan sólo un medio para expresar un algoritmo y una computadora es sólo un procesador para ejecutarlo.

METODOLOGIA PROPUESTA PARA LA CREACIÓN DE UN ALGORITMO

Como se expresó anteriormente, el resultado de la aplicación de las dos primeras fases de la resolución de problemas mediante algoritmos son los algoritmos que luego se programará. Por tal motivo a continuación detallaremos y ejemplificaremos la forma en la que se desarrollarán en la materia estas dos fases. Más adelante se actualizará

Dada la importancia del algoritmo en la ciencia de la computación, un aspecto muy importante será el diseño de algoritmos. A la enseñanza y práctica de esta tarea se dedica gran parte de esta materia.

ANÁLISIS DEL PROBLEMA

El análisis del problema involucra dos etapas: la definición de la situación problema y el análisis propiamente dicho.

1. En la etapa de definición se específica el propósito del algoritmo y una definición clara de lo que se desea resolver. Además, aquí se establece lo que se pretende lograr con su solución.
2. En la etapa de análisis se determinan las características del problema en términos de entradas y salidas. De igual manera, se realiza una investigación de los mecanismos que se deben seguir para resolver el problema. Estos mecanismos se denominan **procesos**. Si existen varios procesos que se podrían aplicar para resolver el problema, se determina cual es el que se va a utilizar. Si no existen procesos, o si estos no satisfacen de manera completa los requisitos para que la solución sea completa, se puede proponer procesos nuevos.

En la figura 1, lo anterior se representa de manera esquemática.

Figura 1. Esquema de la Fase de Análisis del Problema

Ejemplo 1: Se solicita desarrollar una calculadora que permita sumar dos números

Definición del Problema: Desarrollar una calculadora que permita sumar dos números

Como puede observar, en este caso el enunciado del problema es también la definición del problema. Se indica que cual es propósito (crear una calculadora). Además, se definió claramente el problema (la calculadora debe poder sumar 2 números)

Análisis:

- Datos de Entrada: Dos números, a los cuales denominaremos número A y número B
- Proceso:

  ¿Quién debe realizar el proceso?: Una calculadora

  ¿Cuál es el proceso que realiza la calculadora?

  suma = número A + número B.

  Donde lo que se ha aplicado es una ecuación matemática que la calculadora puede realizar. La variable dependiente suma almacena el resultado de sumar al número A, el número B.

- Datos de Salida: suma

Consideraciones previas a la representación de un algoritmo

La variable

Como regla general los datos de entrada y los datos de salida constituyen variables, por tanto, resulta conveniente describir este concepto.

Una variable es un contenedor para almacenar una información. Toda variable consta de:

- Un identificador: es el nombre de la variable. Este nombre debe ser único y no ambiguo.
- Un tipo de datos: tiene por objetivo determinar el rango de valores que puede almacenar la variable. Como consecuencia de esto, el tipo de datos también indica el conjunto de operaciones que se puede aplicar sobre los identificadores.

Existen diferentes tipos de datos. A continuación, se definirán un listado de los tipos de datos más comunes, los cuales comúnmente son denominados “Tipos de Datos Simples”. Los tipos de datos simples se caracterizan por estar presentes en los lenguajes de programación.

Tipos de Datos Simples

- Entero: representa un número entero. Por ejemplo 10
- Real o Flotante: representa un número real o coma flotante. Por ejemplo 10,5
- Carácter: Representa un carácter, por ejemplo ‘a’ o ‘c’.
- Cadena de caracteres o string: Representa un con conjunto de caracteres. Por ejemplo “Programa”.
- Fecha: representa una fecha, generalmente en formato dd/MM/yy. Por ejemplo 11/08/21.

Nomenclatura del identificador

Se refiere al conjunto de reglas que se deben cumplir al momento de definir variables. Si bien muchas reglas dependen del lenguaje de programación, existen reglas comunes. Adicionalmente, con el objetivo de que el estudiante se acostumbre a la nomenclatura que se

usará en el lenguaje de programación elegido para la práctica de la materia se incluirán algunas reglas específicas:

- El nombre del identificador es único y no ambiguo. Esto significa que no debe existir otro identificador con el mismo nombre dentro del mismo ámbito de funcionamiento. Además, este nombre debe interpretar claramente lo que representa, y de hecho no debe lugar a doble interpretación. Por ejemplo, si una variable almacenará el promedio de las notas, un identificador válido sería *promedio*, mientras que identificadores incorrectos serían p o p1 (que no indican claramente que representan, o como se verá luego están reservadas para representar otro tipo de información) o calculo, o resultado_calculo (que no permiten identificar claramente el tipo de información que almacenarán).
- Las variables no inician con un número
- Si el nombre está constituido por varias palabras estas se separarán indicando la siguiente palabra en mayúscula. Por ejemplo, si queremos indicar la cantidad de vidas que le queda a un personaje, podríamos indicarlo de la siguiente manera cantidadVidas.
- El nombre de la variable no debería ser extremadamente largo, ya que luego dificulta la escritura y lectura de las operaciones en las que ellas intervienen.
- Se considera que variables con nombre i, j, k, indican variables enteras usadas para índices; en tanto, las variables de nombre a, b y c, por lo general se utilizan para valores numéricos reales (punto flotante); las variables llamadas p y q se emplean para apuntadores; las variables llamadas n y m son variables que contienen valores de tamaños de matrices.
- Las variables se escriben en minúscula. Por ejemplo, para expresar el puntaje acumulado de un personaje en un juego se puede usar el identificador puntaje.

Refinando el Ejemplo 1:

Con el concepto de variable, redefiniremos la Fase de Análisis de Problema para este ejemplo.

Recuerde que este este problema se ha expresado de la siguiente manera:

Se solicita desarrollar una calculadora que permita sumar dos números

Definición del Problema: Desarrollar una calculadora que permita sumar dos números

Análisis:

- Datos de Entrada:
  numeroA: Real
  numeroB: Real

- Datos de Salida: suma
  resultadoSuma: Real

- Proceso:

  ¿Quién debe realizar el proceso?: Una calculadora

  ¿Cuál es el proceso que realiza la calculadora?

  resultadoSuma = numeroA + numeroB

Observe que la definición de una variable se expresa indicando el identificador de la variable, seguido de dos puntos, un espacio y el tipo de datos.

DISEÑO DEL ALGORITMO

El objetivo de esta fase es formalizar el algoritmo que resuelve el problema. Existen diversas variantes de escritura de un algoritmo dependiendo del autor. Algunos persiguen que el algoritmo esté tan próximo a la forma de escritura del programa, de tal manera que sea casi directa la trascripción al lenguaje de programación. Otros estilos buscan que sean lo suficientemente general e independiente del lenguaje de programación.

En la cátedra vamos a adoptar este segundo enfoque, con el objetivo de no atarse a un lenguaje de programación, pero debido a que vamos a utilizar el paradigma orientado a objetos, vamos a ofrecer una representación desarrollada por el cuerpo docente para facilitar el pasaje hacia ese paradigma. Por consiguiente, a continuación, se compartirá la estructura básica de un algoritmo mediante su representación en el ejemplo 1:

ENTIDAD QUE RESUELVE EL PROBLEMA: Calculadora

VARIABLES

numeroA, numeroB: Real // almacenan los números a sumar

resultadoSuma: Real // almacena el resultado de la suma

NOMBRE ALGORITMO: sumar_numeros

PROCESO DEL ALGORITMO

1. *Leer* numeroA
2. *Leer* numeroB
3. resultadoSuma ← numeroA + numeroB // suma y asigna el resultado
4. *Mostrar* resultadoSuma

Donde la entidad que resuelve el problema es un sustantivo que representa a una cosa o persona que se encarga de ejecutar el algoritmo.

En tanto la sección de variables se utiliza para definir cada una de las variables que intervienen en el algoritmo que ejecuta la entidad. Aquí se expresan las variables de entrada, las variables de salida y toda aquella variable que se necesite durante la ejecución del algoritmo.

Observe que se ha utilizado //, esto significa comentario de línea. Todo aquello que se ubique a la derecha de este comentario no se ejecutará, sirve solamente para clarificar las instrucciones del algoritmo.

Finalmente, en la tercera sección se describen uno o más algoritmos que la entidad puede realizar. A cada uno de ellos se les debe brindar un nombre representativo del proceso realizará. En este caso dado que el proceso sumará dos números, se ha utilizado el nombre `sumar_números`. Observe que inicia con un verbo para indicar acción, y se separan las palabras con un _.

En los algoritmos existen un conjunto de palabras clave reservadas, que son escritas en cursiva. Generalmente representan instrucciones predefinidas o primitivas.

Las primitivas que más van a usar al diseñar algoritmos son *Leer* y *Escribir*. La primera permite señalar que una variable va a almacenar un valor que es indicado por el usuario. La segunda indica que se va a mostrar el valor de una variable.

El operador de asignación (←) sirve para indicar que el resultado de una operación o proceso se almacenará en una variable.

Con toda la información anterior vamos a transcribir el significado del algoritmo del ejercicio 1. Supongamos que se desea sumar los valores 5 y 4:

a) Si se necesita sumar dos números se lo deberemos pedir a la entidad CALCULADORA.
b) Para que CALCULADORA sume dos números se debe invocar el algoritmo sumar_numeros.
c) En el paso 1 del proceso del algoritmo, se lee un valor que se asigna a la variable numeroA. Aquí es donde se ingresa el valor 5, por lo cual numeroA = 5.
d) En el paso 2 se lee un valor que se asigna a la variable numeroB. Aquí es donde se ingresa el valor 4, por lo cual numeroB = 4.
e) En el paso 3 se realiza la operación suma (numeroA + numeroB). Como numeroA almacena el valor 5 y numeroB almacena el valor 4, la operación realiza lo siguiente: 5 + 4. A continuación en el mismo paso el resultado de la operación (5 + 4 = 9) se asigna a la variable resultadoSuma debido al uso del operador de asignación (←). A partir de ese momento resultadoSuma contiene el valor 9. Observe que el operador de asignación se aplica de derecha a izquierda.
f) En el paso 4 se muestra el valor de resultadoSuma como consecuencia de aplicar la primitiva Mostrar.

Existen otras primitivas importantes que se irán incorporando a medida que la complejidad del algoritmo lo requiera.

CONCUSION

Dada la importancia del algoritmo en la ciencia de la computación, un aspecto muy importante será el diseño de algoritmos. A la enseñanza y práctica de esta tarea se dedica gran parte de esta materia.

El diseño de la mayoría de los algoritmos requiere creatividad y conocimientos profundos de la técnica de la programación. En esencia, la solución de un problema se puede expresar mediante un algoritmo.

Finalmente, los algoritmos se pueden representar de diferentes maneras: diagrama de flujos, ecuaciones, pseudocódigo, etc. En la asignatura, veremos una representación conveniente al tipo de programación que se estudiará: La programación orientada a objetos. Para el desarrollador inicial se recomienda seguir estos pasos ya que le permitirán aplicar las técnicas de abstracción y desarrollo de algoritmos (comprobado científicamente), mientras que para los desarrolladores avanzados, les permitirá ser ordenados en el desarrollo de aplicaciones. En ambos casos la idea central es que el desarrollador se tome el tiempo conveniente para analizar y diseñar una solución antes de ir directamente a codificarlas, lo cual en general conduce a codificaciones que pueden poseer grandes inconvenientes.

BIBLIOGRAFIA

Fundamentos de Programación: Algoritmos, estructuras de datos y objetos. Cuarta edición. Luis Goyanes Aguilar. ISBN: 978-84-481-6111-8