Tema 6. Conceptos básicos de programación

Prof. María Alejandra Quintero

Informática

Año 2014-2015

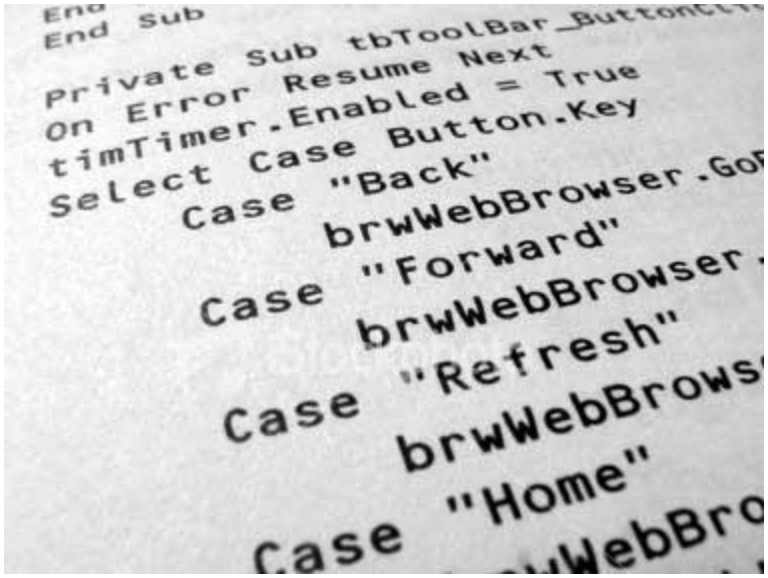
¿Qué es la programación?

Es la acción de escribir programas de computación con el objetivo de resolver un determinado problema.

Implica escribir instrucciones para indicarle a la computadora cómo procesar los datos para producir la información deseada.

Programa:

Secuencia de instrucciones que indica las acciones o tareas que la computadora debe ejecutar para dar solución a un problema determinado.



Lenguaje de programación

- Usados para escribir programas
- Conjunto de reglas ó normas, símbolos y palabras especiales utilizadas para construir un programa. Tienen una sintaxis bien definida.
- Clasificación
  - Lenguaje de máquina
  - Lenguaje ensamblador
  - Lenguaje de alto nivel

Lenguaje de máquina

- Secuencias de números (0's y 1's)
- Lenguaje propio de cada computadora

Lenguaje ensamblador

- Utiliza códigos parecidos al inglés
- Varía de acuerdo al tipo de procesador
- Ejemplo: LOAD X ADD Y STORE Z

Lenguajes de alto nivel

- Instrucciones en un lenguaje familiar
- Usa notaciones matemáticas conocidas
  Ejemplo: Z = X+Y
- Independiente de la máquina

Ejemplos de lenguajes de alto nivel:

- FORTRAN
- ALGOL
- COBOL
- BASIC
- PL/I
- PROLOG
- Pascal
- C
- Turbo C
- Turbo Basic
- Turbo Pascal
- C++
- Visual C
- Visual Basic
- Delphi
- Java
- C#
- Python

Datos

Un dato es la representación de un hecho, evento o elemento del mundo real.

Ejemplo

Una persona puede tener varios datos que permiten identificarla, como:

Nombre, Cédula de Identidad

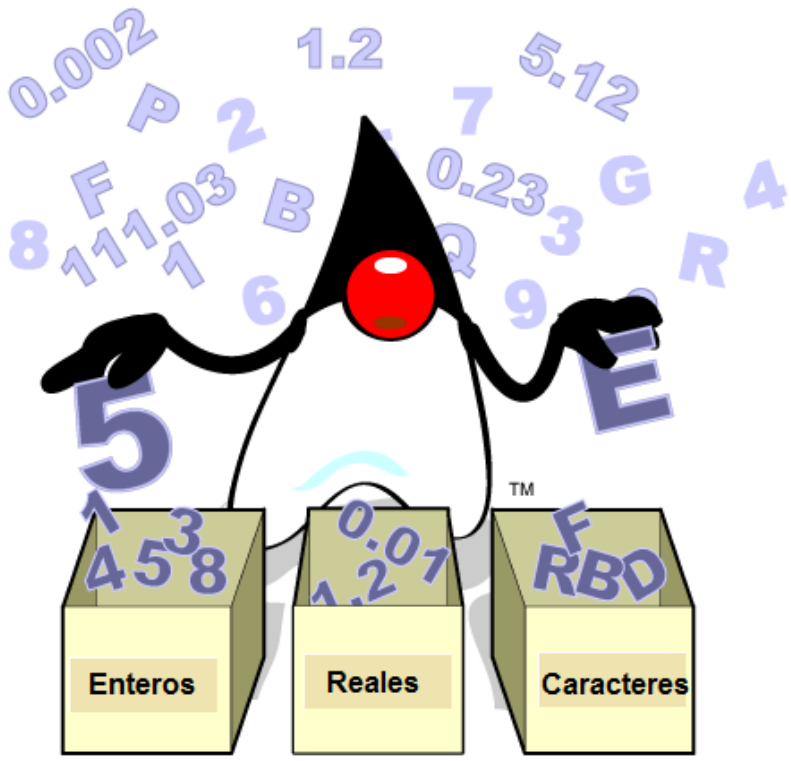
Edad, Sexo, Profesión

¿ Qué datos pudieran usarse para identificar a un árbol?

Tipos de datos

Los tipos de datos básicos utilizados en computación son los siguientes:

- Entero
- Real
- Carácter
- Cadena de caracteres
- Lógicos



Datos de tipo entero

Números que no tienen componentes fraccionarios o decimales. Pueden ser negativos o positivos.

..... –5, –4, –3, –2, –1, 0, 1, 2, 3, 4, 5, ...

Ejemplos: edad de una persona, número de estudiantes en un salón.

Datos de tipo real

Números que pueden tener punto decimal. Pueden ser negativos o positivos.

Sirven para representar valores dentro del conjunto de los números reales .

Ejemplo: altura de un árbol, salario de una persona, impuesto a pagar por la compra de un artículo.

Datos de tipo carácter

Son símbolos que el computador reconoce.
Un carácter puede ser:

Una letra: A, B, …., Z, a, b, c, …., z
Un dígito: 0, 1, 2, 3, …, 9
Un símbolo: ! , $, %, &, *, /, @, ….

Ejemplos: sección de una asignatura, tipo de sangre, calidad de un producto.

Datos de tipo cadena de caracteres

Contienen una sucesión de caracteres delimitadas por comillas.

Ejemplos de cadenas de caracteres:
"Ingeniería Forestal", " 2 de enero de 2013", "M & R computación"

Ejemplo de datos tipo cadena de caracteres:
Nombre de una persona, CI, dirección.

Datos de tipo lógico

Son datos que pueden ser verdaderos o falsos

Ejemplo:

Se desea saber si una persona es soltera. La respuesta puede ser representada por un dato tipo lógico.

Respuesta = Falso o

Respuesta = Verdadero

Los datos pueden ser:

Constantes

- Valores o datos cuyo valor es fijo

Variables

- Son datos cuyo valor cambia cada vez que se usa el programa

Constante

Valor o dato que no puede cambiar en la ejecución de un programa. Son valores fijos.

Ejemplos:

Constante	Tipo de constante
PI = 3.1416	Real
Máximo = 50	Entera
Profesión = “Ing. Forestal”	Cadena de caracteres
Ocupado = Falso	Lógica
Clase= “ A”	Caracter

Variable

Valor o dato que puede cambiar durante la ejecución de un programa. Representa una dirección de memoria donde se guarda un dato.

Todo dato que vaya a ser introducido en la computadora y todo valor que se calcule a partir de otros datos en un programa, deben definirse (declararse) como una variable.

Atributos de las variables

Nombre: usado para identificar la variable

Tipo: corresponde al tipo de dato que describe su uso.

Ejemplos:

Nombre	Tipo
Peso	Real
Apellidos	Cadena de caracteres
Numero_hijos	Entero

Metodología para la construcción de un programa

Paso 1. Análisis

Tiene como finalidad conocer y comprender el problema.

En esta fase se definen los cuáles son los datos necesarios, qué debe hacer el programa y cuáles son los resultados que debe arrojar.

Técnica a utilizar: Análisis E-P-S (Entrada-Proceso-Salida)

Análisis E-P-S

Entrada:

Se especifican cuáles son los datos necesarios para resolver el problema y de qué tipo son.

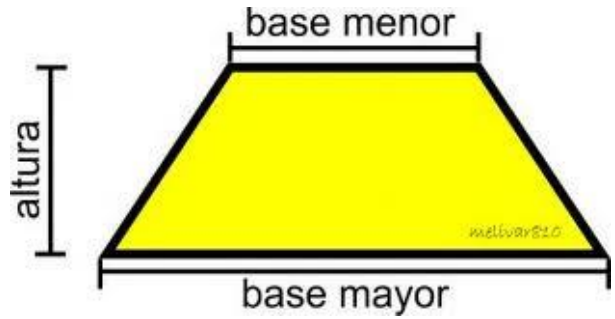
Proceso:

Se indican las operaciones o cálculos que se van a realizar con los datos de entrada para encontrar la solución del problema (ecuaciones).

Salida:

Se definen cuáles son los resultados esperados.

Ejemplo: realizar el análisis E-P-S para calcular el área de un trapecio.



$$Area = \frac{(base\ mayor + base\ menor) \times h}{2}$$

Entrada

Los datos necesarios para resolver el problema son:

- B1: base menor. Tipo: Real
- B2: base mayor. Tipo: Real
- h: altura. Tipo: Real

Proceso

Calcular el área del trapecio usando la ecuación:

$$A = \frac{(B1 + B2) \times h}{2}$$

Salida

A: área del trapecio. Tipo: Real.

Paso 2. Diseño

Consiste en especificar cómo se resuelve el problema.

En esta fase se establece la secuencia de pasos que debe seguirse para obtener la solución del problema.

Esta secuencia es la base para escribir el código en un lenguaje de programación.

Herramientas: Algoritmos
Diagramas de flujo

Algoritmos (diseño)

Un algoritmo es una secuencia ordenada de pasos que llevan a la solución de un problema o a la ejecución de una tarea.

Características de un buen algoritmo:

- Los pasos deben ser precisos y claros.
- Debe seguir un orden lógico.
- Debe tener un principio y un fin (número finito de pasos)
- Debe resolver correctamente el problema

Ejemplo: realizar un algoritmo para calcular el área de un trapecio.

Algoritmo Área del trapecio

0. Inicio
1. Solicitar base menor del trapecio (B1)
2. Solicitar base mayor del trapecio (B2)
3. Solicitar altura del trapecio (h)
4. $A = \frac{(B1 + B2) \times h}{2}$
5. Mostrar el área (A)
6. Fin

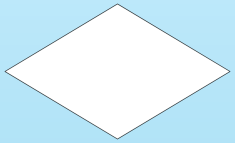
Nota:
La instrucción "solicitar" también puede escribirse como: "obtener" o "leer". Es una instrucción de entrada de datos.

Diagramas de flujo (diseño)

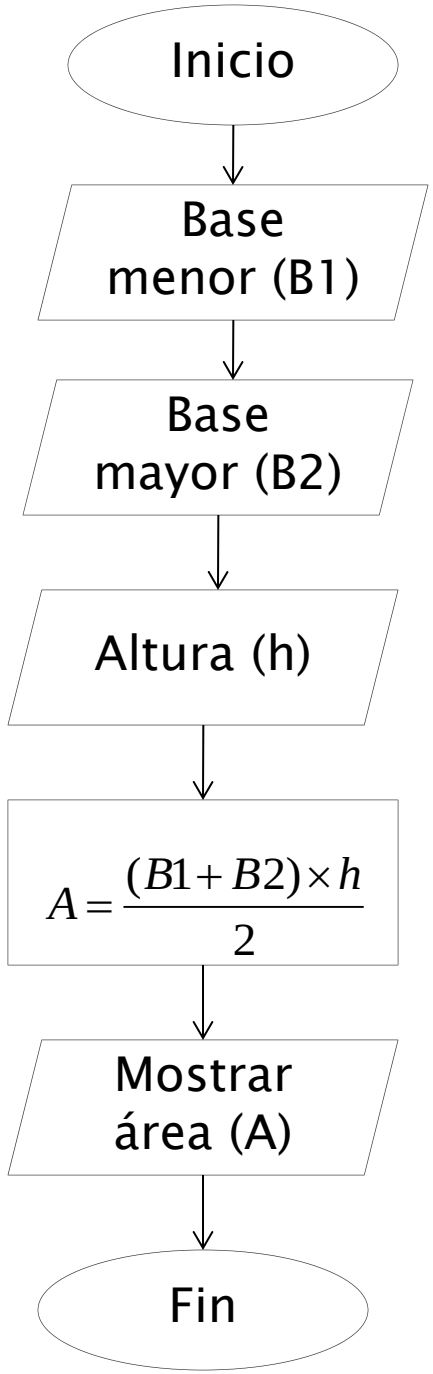
Un diagrama de flujo es la representación gráfica de un algoritmo. Utiliza símbolos para indicar acciones y estos se conectan a través de flechas que muestran el flujo o secuencia del programa.

En el diseño del programa se pueden usar algoritmos o diagramas de flujo, de acuerdo al gusto del programador.

Símbolos usados en los diagramas de flujo

Símbolo	Significado
	Inicio/Fin del programa
	Entrada / salida de datos
	Procesos
	Decisión
	Conector de una misma página
	Conector de página diferente

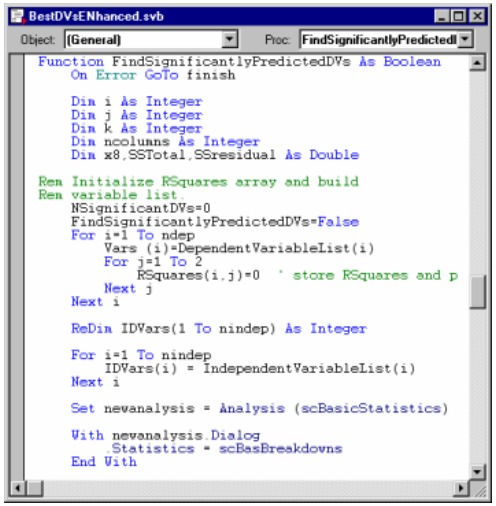
Ejemplo: diagrama de flujo para calcular el área de un trapecio.



Paso 3. Codificación

Traducción de cada uno de los pasos especificados en el diseño a un lenguaje de programación, siguiendo las reglas de sintaxis del mismo.

El resultado de esta fase es un programa escrito en el lenguaje de programación seleccionado, el cual se denomina código fuente.



En la codificación, cada paso escrito en el diseño (algoritmo o diagrama de flujo), se escribe como una instrucción o sentencia.

Tipos básicos de instrucciones *:

- Instrucciones de entrada/salida: permiten obtener datos y mostrar resultados.

- Instrucciones aritmético-lógicas: ejecutan operaciones aritméticas (suma, resta, multiplicación, …) o lógicas.

* Barrios J. 2011. Apuntes de programación I. webdelprofesor.ula.ve/ingenieria/ijudith/

- Instrucciones de decisión: permiten a un programa elegir entre diferentes cursos de acción (tema 8, estructuras de decisión).

- Instrucciones de repetición: permiten repetir una misma tarea o sentencia cierta cantidad de veces (tema 9, estructuras de repetición).

Paso 4. Ejecución

Consiste en poner a funcionar el programa (ejecutar o correr el programa).

En esta etapa el compilador del lenguaje de programación traduce el código fuente a lenguaje de máquina (código objeto), siempre y cuando no tenga errores de sintaxis.

Ejemplo:

X = a /(b+5 En esta instrucción hay un error de sintaxis, falta un paréntesis.

Si hay errores, es necesario corregirlos y volver a compilar el programa.

Paso 5. Pruebas y depuración

En esta etapa se identifican y se corrigen los errores del programa.

Hay dos tipos de errores:

-Errores de sintaxis: ocurren cuando se violan las reglas del lenguaje de programación.

-Errores de lógica: el programa funciona pero los resultados son incorrectos.