Módulo profesional 03
PROGRAMACIÓN

FEBRERO 2023

Innovación en Formación Profesional

De:

Planeta Formación y Universidades

UF1: Programación estructurada. Resumen.

1. Conceptos generales.
2. Estructura de un programa informático.
3. Tipos de datos simples y compuestos.

Conceptos generales

¿Qué es la informática?

- Gestionar información.
- Resolver problemas relacionados con la información.

¿Qué es un proceso?

- Un programa en ejecución.

¿Afecta a los recursos de un ordenador el diseño de un programa informático?

- Sí. La definición adecuada de variables en un programa influye en el consumo de los recursos de un ordenador.

Conceptos generales

- El proceso de la programación

- El conjunto de actividades realizadas para escribir un programa, guardarlo y prepararlo para su ejecución.
- Un programa puede ejecutarse sobre el mismo computador en el que fue escrito, o bien trasladarlo a otro diferente.
- El sistema operativo es el “software” básico que se carga en un computador. Algunos sistemas operativos (en adelante, S.O.) son específicos para un computador concreto, mientras otros pueden funcionar sobre computadores de diseño muy diferente.
- El proceso de la programación supone la creación de una serie de instrucciones para el computador, que se escriben en un lenguaje de programación determinado. El proceso se realiza por medio de un editor, pasando a un compilador para el caso de los lenguajes compilados. Este proceso de compilación comprueba la validez del programa en términos sintácticos y posteriormente lo traduce en instrucciones propias del procesador del computador (lenguaje máquina).
- Es importante diferenciar entre errores de compilación (aquellos producidos en el momento de codificar nuestro programa) y los errores de ejecución (aquellos producidos en el momento de ejecutarlo).
- Los lenguajes interpretados no incluyen compilador. Es en el momento de ejecutarlo cuando se traducen las instrucciones de alto nivel en instrucciones de bajo nivel (lenguaje máquina).

Conceptos generales

- Java

- El JDK (Java development kit o kit de desarrollo de Java) proporciona un compilador y un entorno de ejecución. También un directorio con documentación sobre todas las bibliotecas de Java (también llamadas API).
- Aunque el JDK es gratuito, hay en el mercado sistemas más elaborados que incluyen un conjunto de utilidades que facilitan la codificación de programas, por ejemplo Eclipse (www.eclipse.org)
- Entre estas utilidades tenemos: depuración de código para ver el estado de nuestro programa en un momento determinado de su ejecución, documentación en línea, facilidad para detección y solución de errores, generación automática de código, etc.
- Java es multiplataforma gracias a su arquitectura siguiendo el siguiente proceso:
  - Crear las instrucciones en un fichero con extensión *.java*
  - Compilar el programa utilizando el comando *javac* o con un entorno de desarrollo.
  - Se genera un fichero compilado (extensión *.class*) que se ejecuta en un entorno de ejecución (JRE) específico del sistema en el que se desea lanzar el programa.

Estructura de un programa informático

Bloque de declaración de variables

- Sección inicial en la que se definen las variables y constantes a utilizar.
- Importante definir correctamente cada variable con el tipo adecuado a su uso.
- Define cada variable con un nombre lo suficientemente descriptivo

Bloque de instrucciones

- Bloque de entrada de datos:
  - El programa recibe los datos con los que va a trabajar. Estos datos pueden venir directamente del usuario o bien de otro sistema.
- Bloque de transformación de la información:
  - El programa “hace algo” con estos datos. Los transforma, los modifica, crea otros datos nuevos.
- Bloque de salida de datos:
  - Una vez resuelto el problema, el programa da salida a estos datos mostrándoselos al usuario o enviándolos a otro sistema.

Estructura de un programa informático

Variables

- Dato que se modifica durante la ejecución del programa.

Constantes

- Dato que permanece inmutable durante la ejecución del programa.

Declaración de variables

- Se declaran en el bloque de declaración de variables siguiendo el siguiente patrón:

  Tipo nombre;

  Tipo nombre1, nombre2, nombre3;

  Tipo nombre = valor

Declaración de constantes:

- En java se declara una constante siguiendo el patrón:

  final tipo nombre = valor;

  La palabra reservada *final* indica que el contenido del campo no puede cambiarse durante el programa.

Estructura de un programa informático

Inicialización:

- Se puede hacer en el momento de declarar la variable.
- Si no se especifica en el momento de declarar la variable se establece lo que entendemos por inicialización por omisión. Por ejemplo, para los tipos numéricos este valor por omisión es cero.

Expresiones:

- Término dado a las fórmulas en los lenguajes de programación.

Multiplicación: *

División: /

Suma: +

Resta: -

Módulo: %

Estructura de un programa informático

Precedencia en los operadores

- Se aplican las reglas normales:
  - Los paréntesis preceden a la división y a la multiplicación que se ejecutan antes que la suma y la resta, quedando como sigue:

    Grupo 0: ( )

    Grupo 1: ++ — +

    Grupo 2: * / %

    Grupo 3: + - +

- Ejemplos:

```
int miVariable = (10 + 3) * 6;
```

Primero se suma 10 y 3, luego se multiplica por 6.

```
int miVariable2 = 10 + 3 * 6;
```

Primero se multiplica 3 y 6, luego se suma 10.

Tipos de datos simples y compuestos

Booleanos

- Las condiciones controlan las decisiones que se toman en los programas en cuanto a los diferentes caminos alternativos a seguir. Lo que llamamos el flujo de un programa.
- Las condiciones también se conocen como expresiones booleanas. El resultado puede ser *true* o *false*.
- Usan seis operadores de comparación para comparar los resultados de expresiones numéricas:
  - == igual a
  - != distinto de
  - > mayor que
  - < menor que
  - >= mayor o igual que
  - <= menor o igual que

Ejemplos:

```
boolean esMayorDeEdad;
int edad=22;
esMayorDeEdad = edad >= 18;
//edad vale 22, luego edad >=18 es true que se asigna a la variable
//esMayorDeEdad
```

Tipos de datos simples y compuestos

Booleanos

- Operadores booleanos
  - & y
  - | o
  - ^ o exclusivo
  - ! Negación

Ejemplo:

x & y: Para que sea cierta, tanto x como y deben ser ciertas.
x | y: Para que sea cierta, o x o y, o las dos deben ser ciertas.

- Tabla de verdad

X	Y	X && Y	X || Y	!X
V	V	V	V	F
V	F	F	V	F
F	V	F	V	V
F	F	F	F	V

Tipos de datos simples y compuestos

Booleanos

- Ejemplo en Java:

```
int contador = 0;
int i=10;
boolean esMenor;

esMenor = (contador < i) && (i>1);
//esMenor es true si y solo si se cumplen las dos condiciones.

esMenor = (contador <i) || (i>1);
//esMenor es true si se cumplen una de las dos condiciones, o las dos.
```

- Se pueden construir expresiones más complejas:

```
boolean esMenor = ( ( (contador < i) || < (i < 10) ) && (i > 1) );
```

Tipos de datos simples y compuestos

Caracteres

- El tipo ***char*** abarca todas las letras, dígitos y símbolos que aparecen en el teclado.

- Podemos representar también algún otro carácter del teclado con el prefijo de escape:

\b retroceso
\t tabulador
\n salto de línea
\f cambio de página
\r volver al principio de la línea

- Los caracteres en Java se escriben con una única comilla, por ejemplo ‘a’ y las siguientes operaciones son válidas automáticamente para ellos:
  - Asignación
  - Comparación (los seis operadores descritos anteriormente). Por ejemplo: ’g’ < ‘z‘
  - Entrada y salida

Tipos de datos simples y compuestos

Tipos numéricos

- *byte* se usa normalmente en programas que tratan con datos a nivel de máquina.
- *short* puede utilizarse para ahorrar espacio teniendo en cuenta su rango de valores.
- *int* es el tipo normal para declarar enteros.
- *float* es la elección natural para números reales.

Los tipos numéricos tienen acceso a todas las operaciones ya mencionadas.

Tipos de datos simples y compuestos

Estructuras básicas de control

- ***if/else***

Sintaxis:

```
if (condición)
   Instrucción;
else instruction;
```

Primero, se evalúa la condición. Si el resultado es cierto, se ejecuta la parte del *if*, si es falso se ejecuta la parte del *else*.

- ***while***

Sintaxis:

```
Inicializar las condiciones
While (condiciones) {
  Instrucciones del bucle
  Actualizar las condiciones
}
```

Después de las instrucciones que inicializan las variables involucradas en las condiciones, empieza el bucle en sí comprobando las condiciones. Si el resultado de evaluar la condición es cierta, se ejecuta el cuerpo de instrucciones. Se sale del bucle cuando las condiciones son falsas.

- ***do-while***

Sintaxis:

```
Inicializar las condiciones
do {
  Instrucciones del bucle
  Actualizar las condiciones
} While (condiciones)
```

El cuerpo de instrucciones se ejecuta como mínimo una vez ya que las condiciones se evalúan al final

- ***for***

Sintaxis:

```
for (inicio; test; actualizacion) {
  Conjunto de instrucciones
}

Inicio: define e inicializa una o varias variables
```

Tipos de datos simples y compuestos

Estructuras básicas de control

- ***switch***

Sintaxis:

```
switch (expresion) {
   case valor: Instrucción; break;
   case valor: Instrucción; break;
    …
   default: Instrucción; break;
}
```

La instrucción *switch* toma el valor de la expresión y, empezando con el primer valor, intenta encontrar una correspondencia. Si se encuentra, se ejecuta la instrucción asociada y el *break* hace que el control pase al final de la instrucción *switch*. La palabra clave *default* captura todos los valores que no han sido mencionados.

Sin un *break* el control pasa al siguiente caso.

La expresión debe producir un valor que sea un entero o un carácter.