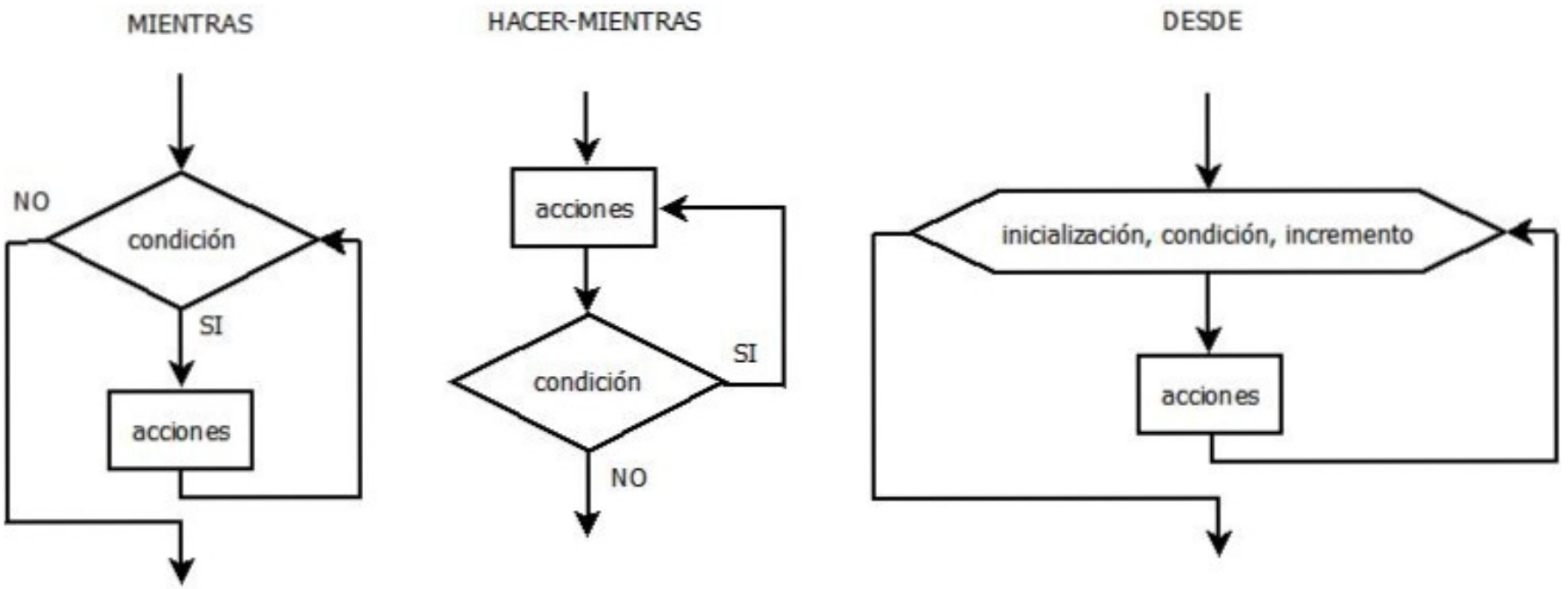
PROGRAMACIÓN

Unidad 2. Estructuras de control de flujo en Java



José L. Berenguel

Tabla de Contenidos

Estructuras de control de flujo

- Se puede demostrar matemáticamente que un algoritmo puede describirse usando solo 3 estructuras de control de flujo:
  - **Secuencial**. Las sentencias se ejecutan una detrás de la otra.
  - **Condicional**. Se ejecutan o no, sentencias en función del valor de verdad de una condición.
  - **Cíclica**. Se ejecutan repetidamente unas sentencias mientras se cumpla una determinada condición.

Estructuras de control de flujo

- Instrucciones condicionales:
  - SI-SI_NO (***if-else***).
  - SEGÚN_SEA (***switch***).
- Instrucciones cíclicas, iterativas o bucles:
  - DESDE/PARA (***for***).
  - MIENTRAS (***while***).
  - HACER-MIENTRAS (***do-while***).

Estructura secuencial

Flujograma:

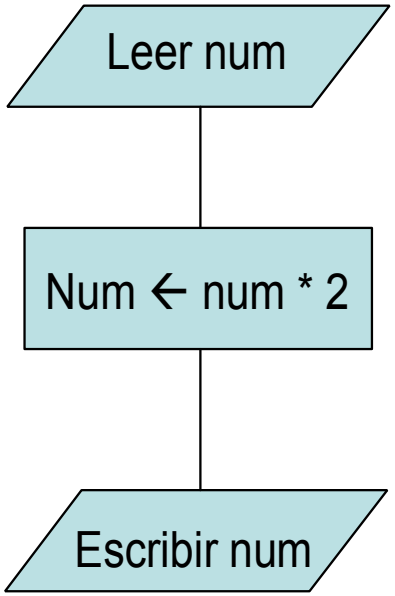
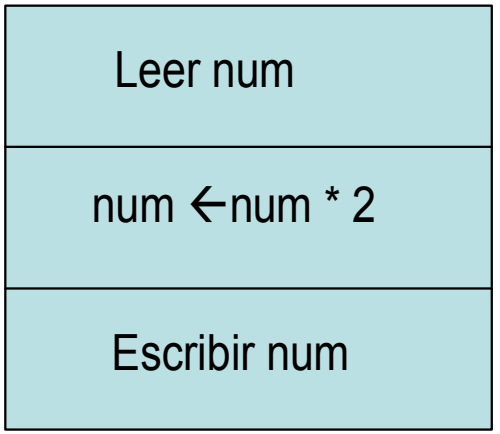


Diagrama N-S:



Pseudocódigo:

```
Leer num
Num ← num * 2
Escribir num
```

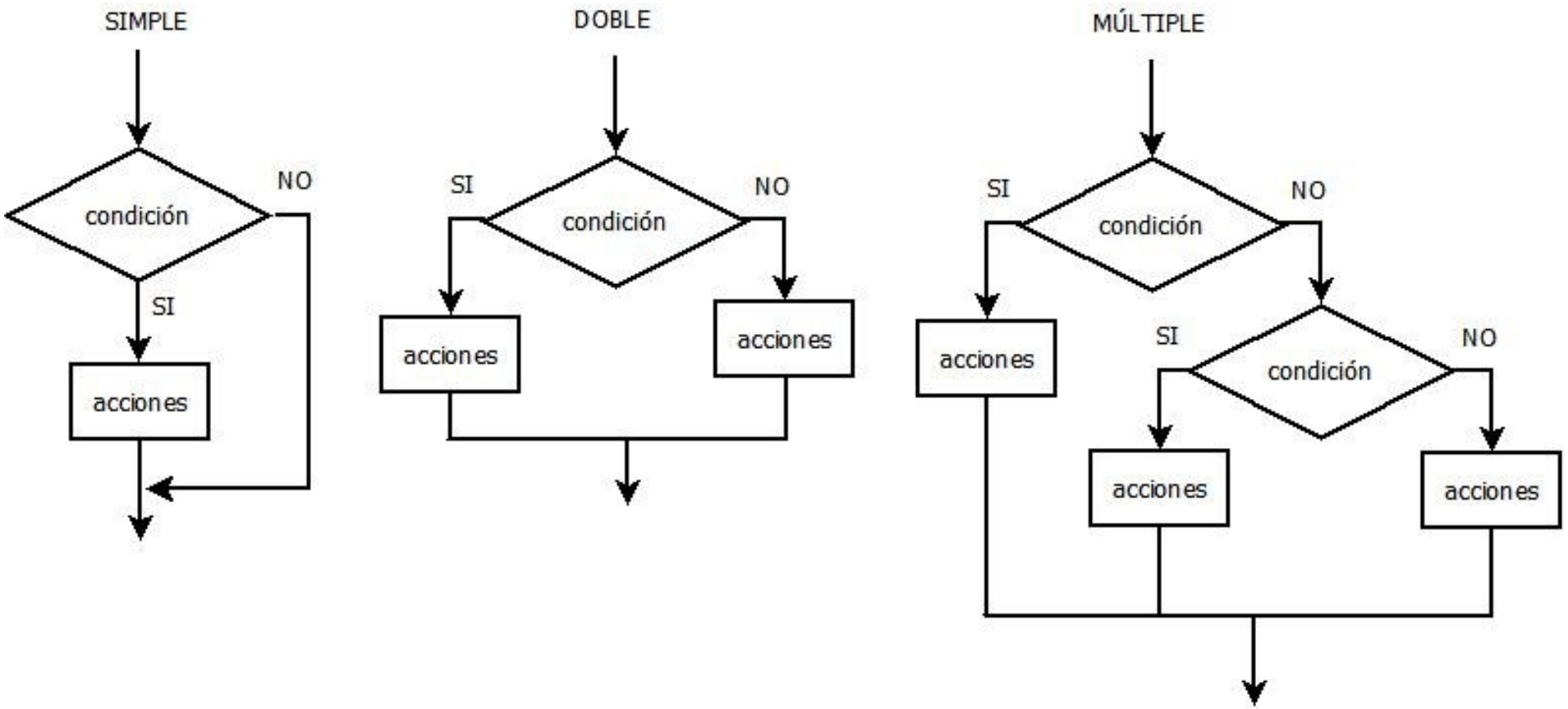
Estructura secuencial en Java

- Las instrucciones se ejecutan en el orden en el que aparecen en el programa.
- Pueden escribirse en una línea o en varias.

```
{sentencia1; sentencia2;...; sentenciaN;}
```

```
{
  sentencia1;
  sentencia2;
  …;
  sentenciaN;
}
```

Estructuras condicionales o selectivas



La instrucción *if-else* en Java

- La condición debe ser una expresión de tipo *boolean*.
- El bloque ***else*** es opcional.
- Las llaves delimitadoras se pueden omitir cuando solo hay una única instrucción, pero se aconseja su uso.
- Las sentencias ***if*** se pueden anidar.

```
if(condición){
     sentencias
} else {
     sentencias
}
```

```
if(a>b){
     System.out.println("El mayor es"+a);
} else if(a<b){
     System.out.println("El mayor es"+b);
} else{
     System.out.println("Los dos números son iguales");
}
```

La instrucción *switch* en Java

- Permite ejecutar diferentes bloques de instrucciones en función del resultado de una expresión.
- Se puede sustituir por instrucciones *if-else* anidadas.
- Puede contener cualquier número de ***case*** pero no puede haber dos con el mismo valor.
- La sentencia ***break*** se emplea para provocar la finalización del *switch*.
- El bloque ***default*** es opcional y se ejecuta si el resultado de la expresión no coincide con ningún *case*.

```
switch(expresión){
     case valor1:
          sentencias;
          break;
     case valor2:
          sentencias;
          break;
     default:
          sentencias;
}
```

La instrucción *switch* en Java

- Ejemplo de instrucción *switch*:

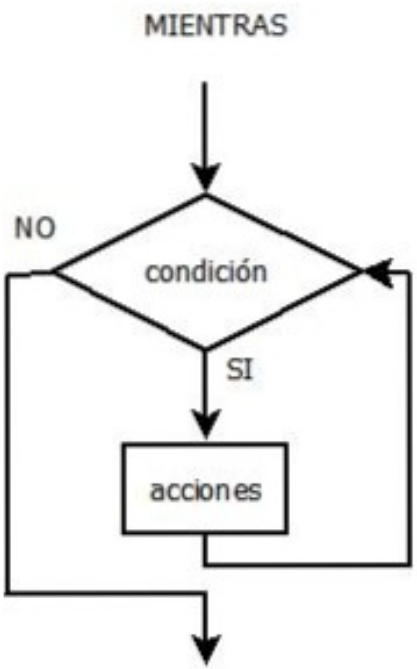
```
switch(nota){
      case 0:
      case 1:
      case 2:
      case 3:
      case 4:
            System.out.println(“Suspenso”);
            System.out.println(“Ánimo”);
            break;
      case 5:
            System.out.println(“Suficiente”);
            break;
      case 6:
            System.out.println(“Bien”);
            break;
      case 7:
      case 8:
            System.out.println(“Notable”);
            break;
      case 9:
      case 10:
            System.out.println(“Sobresaliente”);
            System.out.println(“Enhorabuena”);
            break;
      default:
            System.out.println(“Nota incorrecta”);
}
```

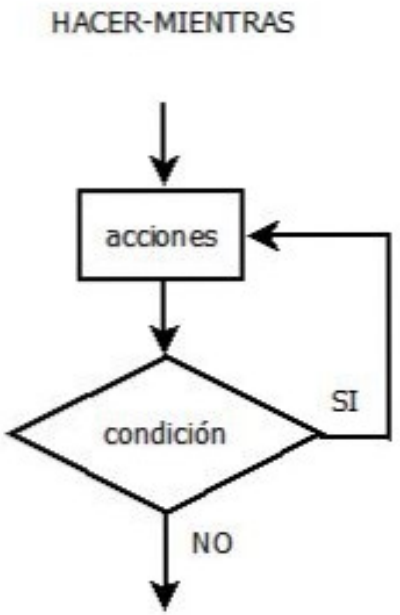
La instrucción *switch* en Java

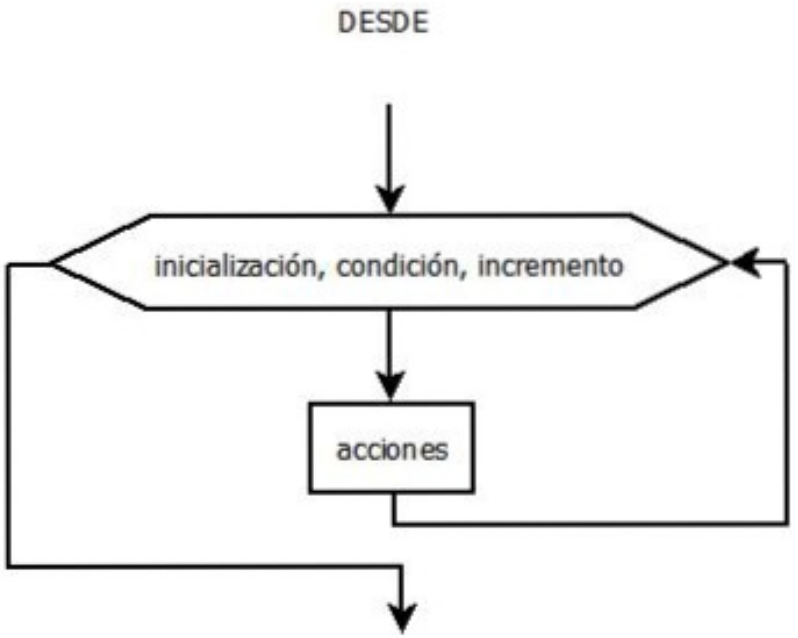
- En versiones recientes del lenguaje se elimina la obligatoriedad de usar *break*, para distinguirlo de la anterior versión, en lugar de : se utiliza -> y bloques { }

```
switch(nota){
     case 0,1,2,3,4 -> {
          System.out.println(“Suspenso”);
          System.out.println(“Ánimo”);
     }
     case 5 ->
          System.out.println(“Suficiente”);
     case 6 ->
          System.out.println(“Bien”);
     case 7,8 ->
          System.out.println(“Notable”);
     case 9,10 -> {
          System.out.println(“Sobresaliente”);
          System.out.println(“Enhorabuena”);
     }
     default:
          System.out.println(“Nota incorrecta”);
}
```

Estructuras cíclicas







La instrucción for en Java

- Permite ejecutar un conjunto de instrucciones un número determinado de veces.
  - Las **instrucciones de control** (*inicialización*, *condición* e *incremento*) son opcionales, pero el ; debe estar presente. Por ejemplo: ***for(int i=0;;)***
  - Si se declara una variable en la inicialización esta será accesible solo en el interior del bucle.

```
for(inicialización;condición;incremento){
     sentencias;
}
```

```
for(int i=1;i<=10;i++){
     //Muestra los números del 1 al 10
     System.out.println("El número es "+i);
}
```

La instrucción *while* y *do-while*

- Ejecutan un bloque de instrucciones mientras se cumple la condición.
  - ***while***: condición antes de ejecutar el bucle. Puede no ejecutarse ninguna vez.
  - ***do-while***: condición al final del bucle. Al menos se ejecuta una vez.

```
while(condición){
     sentencias;
}
```

```
do{
     sentencias;
}while(condición);
```

Salida forzada de un bucle

- Los bucles ***for***, ***while*** y ***do-while*** pueden abandonar la ejecución del bloque de instrucciones antes de su finalización, mediante las instrucciones ***break*** y ***continue***:
  - ***break***: Se vio su utilidad en la instrucción *switch*. Provoca la salida forzada del bucle, continuando la ejecución por la primera sentencia situada después del mismo.
  - ***continue***: provoca que el bucle detenga la iteración actual y pase a iniciar una nueva iteración.

Se desaconseja el uso de las sentencias break y continue ya que rompe el flujo normal del programa.

Unidad 2. Estructuras de control de flujo en Java

DUDAS Y PREGUNTAS