UNIVERSIDAD AUTÓNOMA DE SINALOA

Facultad de Informática Culiacán

Estructura de Selección y Repetición en Java

Instructor:

MC. Gerardo Gálvez Gámez

gerardo.galvez@uas.edu.mx

Noviembre de 2016

Bloques de instrucciones

Se usan llaves para delimitar bloques

```
{
  // code
}
```

Un bloque y su bloque padre o pueden tener una variable con el mismo nombre

```
{
  int i;
  ...
  {
    int i;
    ...
  }
}
```

Bloques hermanos pueden tener variables con el mismo nombre

```
{
  int i;
  ...
}
...
{
  int i;
  ...
}
```

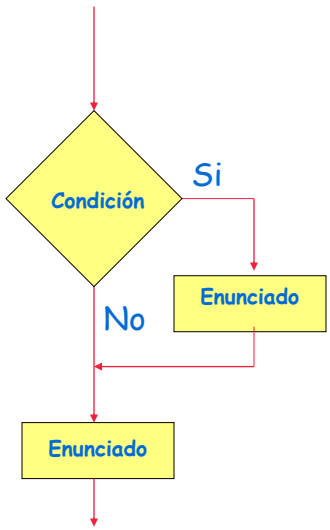
Tipos de instrucciones

Instrucciones Condicionales
Las instrucciones if y switch

Instrucciones de iteración
Las instrucciones while, do, for, y for-each

Instrucciones de salto
Las instrucciones goto, break, y continue

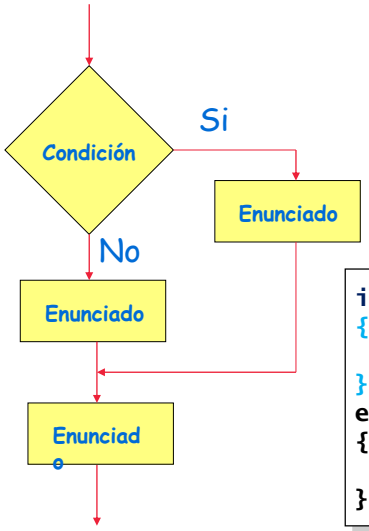
La instrucción if simple



Sintaxis

```
if ( expresión-booleana )
{
   instrucción-incrustada
}
```

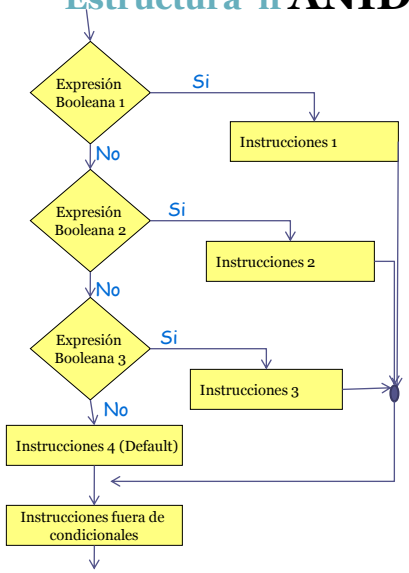
La instrucción if Doble



Sintaxis

```
if ( expresión-booleana )
{
  primera-instrucción-incrustada
}
else
{
  segunda-instrucción-incrustada
}
```


Estructura if ANIDADA



Permiten elegir entre dos o más opciones o alternativas posibles, en función del cumplimiento o no de las Expresiones Booleanas evaluadas.

Estructura de Selección if **Anidada**

Sintaxis

```
if ( expresión-booleana1 )
{
   Instrucciones1 (acciones a realizar) caso true
}
else
{
   if ( expresión-booleana2 )
   {
      Instrucciones2 (acciones a realizar) caso true
    }
    else
    {
          if ( expresión-booleana3 )
          {
             Instrucciones3 (acciones a realizar) caso true
          }
          else
          {
             Instrucciones (acciones a realizar) caso Default
           }
     }
}
```


Estructura if en Cascada

Sintaxis

```
if ( expresión-booleana1 )
{
   Instrucciones1 (acciones a realizar) caso true
}
else if ( expresión-booleana2 )
{
     Instrucciones2 (acciones a realizar) caso true
}
 else if ( expresión-booleana3 )
{
     Instrucciones3 (acciones a realizar) caso true
}
else
{
     Instrucciones (acciones a realizar) caso Default
}
```

La instrucción switch

- La instrucción **switch** proporciona un mecanismo elegante para expresar condiciones complejas que, de lo contrario, requerirían el uso de instrucciones **if** anidadas.
- Consta de bloques de varios casos, cada uno de los cuales especifica una sola constante y una etiqueta **case** asociada.
- En switch el valor de la expresión y de las constates tiene que ser de tipo **char, byte, short o int**. No hay lugar para boolean, reales ni long.

Selección y Repetición Java• FIUAS

La instrucción switch

- No está permitido agrupar varias constantes en una sola etiqueta **case**, sino que cada constante debe tener la suya propia
- Un bloque **switch** puede contener declaraciones.
- El ámbito de una constante o variable local declarada en un bloque **switch** se extiende desde su declaración hasta el final del bloque **switch**

Sintaxis de la instrucción switch

```
switch (<expresión>)
{
      case <valor1>:  <bloque1>
                      <siguienteAcción>
                        break;

      case <valor2>:  <bloque2>
                      <siguienteAcción>
                      break;
      ...

      default:  <bloqueDefault>
                <siguienteAcción>
                break;
 }
```


Sintaxis de la instrucción switch

```
switch (expresión)
{

      case  valor1:
      case  valor2:
      case  valor3:
                  instrucciones;
                  break;

      case valor4:
                  instrucciones;
                  break;
      .
      .
      .

      default:
              sentencias;
              break;
 }
```

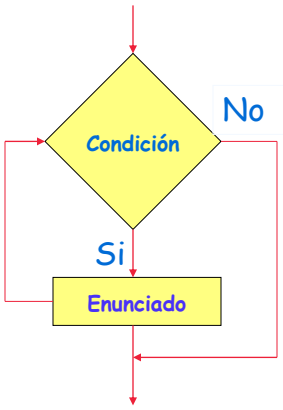
Uso de instrucciones Iterativas

- Se ejecutan repetidamente mientras se cumple una condición. También se conocen como instrucciones de bucle.
- Cada una de estas instrucciones está pensada para un estilo de iteración distinto.
  - La instrucción while
  - La instrucción do
  - La instrucción for
  - La instrucción for-each

La instrucción while

- Ejecuta instrucciones en función de un valor booleano
- Evalúa la expresión booleana al principio del bucle
- Ejecuta las instrucciones mientras el valor booleano sea True

Fujo de ejecución de while



Sintaxis:

inicialización
while (*expresión-booleana* **)**
{
  instrucción-incrustada
  actualización
}

Flujo de ejecución

Una instrucción **while** se ejecuta de la siguiente manera:

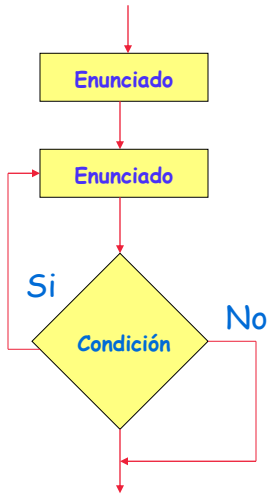
1. Se evalúa la expresión booleana que controla la instrucción **while**.
2. Si la expresión booleana se cumple (**true**), el control pasa a la instrucción incrustada. Al llegar al final de la misma, el control se transfiere implícitamente al inicio de la instrucción **while** y se vuelve a evaluar la expresión booleana.
3. Si la expresión booleana no se cumple (**false**), el control pasa al final de la instrucción **while**. Por lo tanto, el programa ejecuta repetidamente la instrucción incrustada mientras la expresión booleana de control sea **true**.

La expresión booleana se prueba al inicio del bucle **while**, por lo que es posible que la instrucción incrustada no se llegue a ejecutar.

La instrucción do

- Ejecuta instrucciones en función de un valor booleano
- Evalúa la expresión booleana al final del bucle
- Ejecuta las instrucciones mientras el valor booleano sea True

La instrucción do



Sintaxis:

inicialización
do {

<instrucciones>

} while(<condición>**);**

Flujo de ejecución

Una instrucción **do** se ejecuta de la siguiente manera:

1. El control pasa a la instrucción incrustada.
2. Al llegar al final de la instrucción incrustada, se evalúa la expresión booleana.
3. Si la expresión booleana se cumple (**true**), el control pasa al inicio de la instrucción **do**.
4. Si la expresión booleana no se cumple (**false**), el control pasa al final de la instrucción **do**.

La instrucción for

Sintaxis:

```
for ( inicialización ; condición ; actualización )
{
      instrucción-incrustada
}
```

Como en las demás instrucciones iterativas, la condición en un bloque **for** debe ser una expresión booleana que funciona como condición para la continuación, no para la terminación.

Declaración de variables

Una sutil diferencia entre las instrucciones **while** y **for** es que una variable declarada en el código de inicialización de una instrucción **for** sólo tiene validez dentro de ese bloque **for**.

Por ejemplo:

el siguiente código generará un error en tiempo de compilación:

```
for (int i = 0; i < 10; i++)
{
  Console.WriteLine(i);
}
Console.WriteLine(i); // Error: i está fuera de ámbito
```


La instrucción “for extendido” o “for each”.

Facilita el recorrido de objetos existentes en una colección sin necesidad de definir el número de elementos a recorrer.

Sintaxis:

```
for ( TipoARecorrer nombreVariableTemporal : nombreDeLaColección ) {
        Instrucciones
}
```

Nota: Para saber si un for es un for extendido o un for normal hemos de fijarnos en la sintaxis que se emplea.

La instrucción “for extendido” o “for each”.

- Elige el tipo y el nombre de la variable de iteración
- Ejecuta instrucciones incrustadas para cada elemento de la clase collection

```
public void listarTodosLosNombres () {
      for (String Nombre: ListaNombres) {
       //Muestra cada uno de los nombres dentro de ListaNombres
      System.out.println (Nombre);
   }
 }
```


Uso de instrucciones de salto

se usan para transferir el control incondicionalmente a otra instrucción.

- Las instrucciones:
  - break y
  - continue

Las instrucciones break and continue

- La instrucción break salta fuera de una iteración
- La instrucción continue salta a la siguiente iteración

```
int i = 0;
while (true) {
    Console.WriteLine(i);
    i++;
    if (i < 10)
    {
        continue;
    }
    break;
    Console.Write("FIN");
}
```

Selección y Repetición Java• FIUAS

Codificación de Algoritmos en Pseudocódigos, con Estructuras de Selección y Repetición.

Preguntas?

Selección y Repetición Java• FIUAS

Actividades

ExtraClases

Objetivo:

El alumno demostrara la habilidad alcanzada en clases, para codificar pseudocódigos de diversos problemas, que utilizan procedimientos de solución de toma de decisiones y Estructuras Repetitivas.