4.B. Estructuras de selección.

Sitio: VIRGEN DE LA PAZ
Curso: Programación
Libro: 4.B. Estructuras de selección.

Imprimido por: Cristian Esteban Gómez
Día: miércoles, 31 de mayo de 2023, 11:01

Tabla de contenidos

1. Estructuras de selección.

¿Cómo conseguimos que nuestros programas puedan tomar decisiones? Para comenzar, lo haremos a través de las estructuras de selección. Estas estructuras constan de una sentencia especial de decisión y de un conjunto de secuencias de instrucciones.

El funcionamiento es sencillo, la sentencia de decisión será evaluada y ésta devolverá un valor (verdadero o falso), en función del valor devuelto se ejecutará una secuencia de instrucciones u otra. Por ejemplo, si el valor de una variable es mayor o igual que 5 se imprime por pantalla la palabra APROBADO y, si es menor, se imprime SUSPENSO. Para este ejemplo, la comprobación del valor de la variable será la sentencia especial de decisión. La impresión de la palabra APROBADO será una secuencia de instrucciones y la impresión de la palabra SUSPENSO será otra. Cada secuencia estará asociada a cada uno de los resultados que puede arrojar la evaluación de la sentencia especial de decisión.

Recomendación

En el lenguaje de programación C, verdadero o falso se representan mediante un literal entero. 0 representará Falso y 1 o cualquier otro valor, representará Verdadero. Como sabes, en Java las variables de tipo booleano sólo podrán tomar los valores true (verdadero) o false (falso).

La evaluación de las sentencias de decisión o expresiones que controlan las estructuras de selección, devolverán siempre un valor verdadero o falso.

Las estructuras de selección se dividen en:

1. Estructuras de selección simples o estructura `if`.
2. Estructuras de selección compuestas o estructura `if-else`.
3. Estructuras de selección basadas en el `operador condicional`.
4. Estructuras de selección múltiples o estructura `switch`.

A continuación, detallaremos las características y funcionamiento de cada una de ellas. Es importante que a través de los ejemplos que vamos a ver, puedas determinar en qué circunstancias utilizar cada una de estas estructuras. Aunque un mismo problema puede ser resuelto con diferentes estructuras e incluso, con diferentes combinaciones de éstas.

1.1. Estructura if / if-else.

La estructura `if` es una estructura de selección o estructura condicional, en la que se evalúa una expresión lógica o sentencia de decisión y en función del resultado, se ejecuta una sentencia o un bloque de éstas.

La estructura `if` puede presentarse de las siguientes formas:

<table>
<tr><th colspan="3">Estructura if e if-else.</th></tr>
<tr><th colspan="2">Estructura <code>if</code> simple.</th><th>Estructura <code>if</code> de doble alternativa.</th></tr>
<tr><td>Sintaxis:

if (expresión-lógica)

sentencia1;

sentencia2;

…;</td><td>Sintaxis:

if (expresión-lógica)

{

sentencia1;

sentencia2;

…;

sentenciaN;

}</td><td>Sintaxis:

if (expresión-lógica)

sentencia1;

else

sentencia2;</td><td>Sintaxis:

if (expresión-lógica)

{

sentencia1;

…;

sentenciaN;

}

else

{

sentencia1;

…;

sentenciaN;

}</td></tr>
<tr><td colspan="2">Funcionamiento:

Si la evaluación de la expresión-lógica ofrece un resultado verdadero, se ejecuta la <code>sentencia1</code> o bien el bloque de sentencias asociado. Si el resultado de dicha evaluación es falso, no se ejecutará ninguna instrucción asociada a la estructura condicional.</td><td colspan="2">Funcionamiento:

Si la evaluación de la expresión-lógica ofrece un resultado verdadero, se ejecutará la primera sentencia o el primer bloque de sentencias. Si, por el contrario, la evaluación de la expresión-lógica ofrece un resultado falso, no se ejecutará la primera sentencia o el primer bloque y sí se ejecutará la segunda sentencia o el segundo bloque.</td></tr>
</table>

Haciendo una interpretación cercana al pseudocódigo tendríamos que si se cumple la condición (expresión lógica), se ejecutará un conjunto de instrucciones y si no se cumple, se ejecutará otro conjunto de instrucciones.

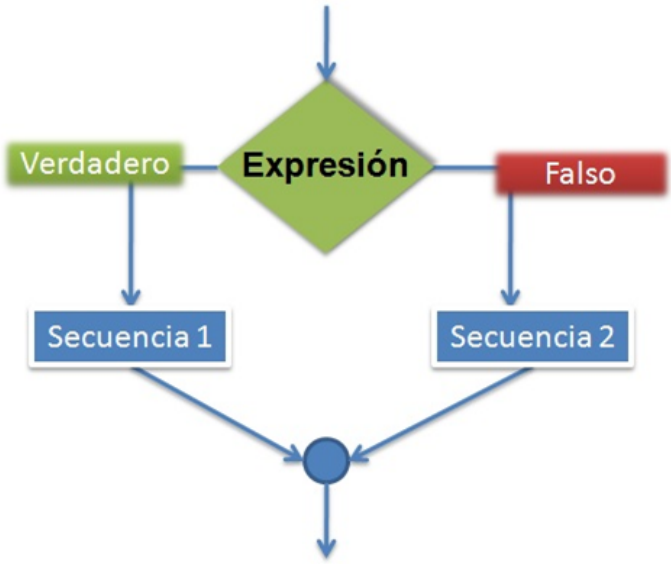

Imagen extraída de curso Programación del MECD.

Hay que tener en cuenta que la cláusula `else` de la sentencia `if` no es obligatoria. En algunos casos no necesitaremos utilizarla, pero sí se recomienda cuando es necesario llevar a cabo alguna acción en el caso de que la expresión lógica no se cumpla.

En aquellos casos en los que no existe cláusula else, si la expresión lógica es falsa, simplemente se continuarán ejecutando las siguientes sentencias que aparezcan bajo la estructura condicional `if` .

Los condicionales `if` e `if-else` pueden anidarse, de tal forma que dentro de un bloque de sentencias puede incluirse otro `if` o `if-else`. El nivel de anidamiento queda a criterio del programador, pero si éste es demasiado profundo podría provocar problemas de eficiencia y legibilidad en el código. En otras ocasiones, un nivel de anidamiento excesivo puede denotar la necesidad de utilización de otras estructuras de selección más adecuadas.

Cuando se utiliza anidamiento de este tipo de estructuras, es necesario poner especial atención en saber a qué `if` está asociada una cláusula else. Normalmente, un`else`estará asociado con el `if` inmediatamente superior o más cercano que exista dentro del mismo bloque y que no se encuentre ya asociado a otro `else`.

Debes conocer

Para completar la información que debes saber sobre las estructuras `if` e `if-else`, estudia el siguiente código. En él podrás analizar el código de un programa que realiza el cálculo de la nota de un examen de tipo test. Además de calcular el valor de la nota, se ofrece como salida la calificación no numérica de dicho examen. Para obtenerla, se combinarán las diferentes estructuras condicionales aprendidas hasta ahora.

Presta especial atención a los comentarios incorporados en el código fuente, así como a la forma de combinar las estructuras condicionales y a las expresiones lógicas utilizadas en ellas.

```
package sentencias_condicionales;

/**

 *

 * Ejemplos de utilización de diferentes estructuras

 * condicionales simples, completas y anidamiento de éstas.

 */

public class Sentencias_condicionales {

    /*Vamos a realizar el cálculo de la nota de un examen

     * de tipo test. Para ello, tendremos en cuenta el número

     * total de pregunta, los aciertos y los errores. Dos errores
```

```
    * anulan una respuesta correcta.
    *
    * Finalmente, se muestra por pantalla la nota obtenida, así
    * como su calificación no numérica.
    *
    * La obtención de la calificación no numérica se ha realizado
    * combinando varias estructuras condicionales, mostrando expresiones
    * lógicas compuestas, así como anidamiento.
    *
    */
   public static void main(String[] args) {
        // Declaración e inicialización de variables
       int num_aciertos = 12;
       int num_errores = 3;
       int num_preguntas = 20;
       float nota = 0;
       String calificacion="";

       //Procesamiento de datos
       nota = ((num_aciertos - (num_errores/2))*10)/num_preguntas;

       if (nota < 5)
       {
          calificacion="INSUFICIENTE";
       }
       else
       {
          /* Cada expresión lógica de estos if está compuesta por dos
           * expresiones lógicas combinadas a través del operador Y o AND
           * que se representa con el símbolo &&. De tal manera, que para
           * que la expresión lógica se cumpla (sea verdadera) la variable
           * nota debe satisfacer ambas condiciones simultáneamente
           */
           if (nota >= 5 && nota <6)
              calificacion="SUFICIENTE";
           if (nota >= 6 && nota <7)
              calificacion="BIEN";
           if (nota >= 7 && nota <9)
              calificacion="NOTABLE";
```

```
            if (nota >= 9 && nota <=10)

               calificacion="SOBRESALIENTE";

        }


        //Salida de información

        System.out.println ("La nota obtenida es: " + nota);

        System.out.println ("y la calificación obtenida es: " + calificacion);

    }


}
```

Autoevaluación

¿Cuándo se mostrará por pantalla el mensaje incluido en el siguiente fragmento de código?

```
If (numero % 2 == 0);

System.out.print("El número es par /n");
```

- ○ Nunca.
- ○ Siempre.
- ○ Cuando el resto de la división entre 2 del contenido de la variable numero, sea cero.

1.2. Operador condicional.

El operador condicional o también llamado operador ternario no es más que una forma contraída o reducida de la estructura **if-else**.

Se respresentará medianto el uso conjunto de los símbolos **?** y **:**

La estructura del operador condicional o ternario es como sigue:

```
(expresion_a_evaluar)?valor_si_verdadero:valor_si_falso;
```

Si la expresión evaluada toma un valor de **true** se devolverá el valor asociado al caso de verdadero (primer valor). Si por el contrario, toma un valor de **false** , entonces devolverá el valor asociado al caso de falso (segundo valor).

Veamos un ejemplo de uso:

```
int num1 = 10;
int num2 = 5;
int numMayor;

numMayor = (num1 > num2) ? num1 : num2;

System.out.println("El número mayor es " + numMayor);
```

Aquí sucederá que sin num1 es mayor que num2, la variable numMayor tomará el valor de num1. Si no, numMayor tomará el valor de num2.

1.3. Estructura switch.

¿Qué podemos hacer cuando nuestro programa debe elegir entre más de dos alternativas?, una posible solución podría ser emplear estructuras `if` anidadas, aunque no siempre esta solución es la más eficiente. Cuando estamos ante estas situaciones podemos utilizar la estructura de selección múltiple `switch`. En la siguiente tabla se muestra tanto la sintaxis, como el funcionamiento de esta estructura.

Estructura `switch`

Sintaxis:

```
switch (expresion) {
case valor1:
sentencia1_1;
sentencia1_2;
….
break;
….
….
case valorN:
sentenciaN_1;
sentenciaN_2;
….
break;
default:
sentencias-default;
}
```

Condiciones:

- Donde expresión debe ser del tipo `char`, `byte`, `short` o `int`, y las constantes de cada case deben ser de este tipo o de un tipo compatible.
- La expresión debe ir entre paréntesis.
- Cada `case` llevará asociado un valor y se finalizará con dos puntos.
- El bloque de sentencias asociado a la cláusula `default` puede finalizar con una sentencia de ruptura `break` o no.

Funcionamiento:

- Las diferentes alternativas de esta estructura estarán precedidas de la cláusula `case` que se ejecutará cuando el valor asociado al `case` coincida con el valor obtenido al evaluar la expresión del `switch`.
- En las cláusulas `case`, no pueden indicarse expresiones condicionales, rangos de valores o listas de valores. (otros lenguajes de programación sí lo permiten). Habrá que asociar una cláusula `case` a cada uno de los valores que deban ser tenidos en cuenta.
- La cláusula `default` será utilizada para indicar un caso por defecto, las sentencias asociadas a la cláusula `default` se ejecutarán si ninguno de los valores indicados en las cláusulas `case` coincide con el resultado de la evaluación de la expresión de la estructura `switch`.
- La cláusula `default` puede no existir, y por tanto, si ningún `case` ha sido activado finalizaría el `switch`.
- Cada cláusula `case` puede llevar asociadas una o varias sentencias, sin necesidad de delimitar dichos bloques por medio de llaves.
- En el momento en el que el resultado de la evaluación de la expresión coincide con alguno de los valores asociados a las cláusulas `case`, se ejecutarán todas las instrucciones asociadas hasta la aparición de una sentencia `break` de ruptura. (la sentencia `break` se analizará en epígrafes posteriores)

En resumen, se ha de comparar el valor de una expresión con un conjunto de constantes, si el valor de la expresión coincide con algún valor de dichas constantes, se ejecutarán los bloques de instrucciones asociados a cada una de ellas. Si no existiese coincidencia, se ejecutarían una serie de instrucciones por defecto.

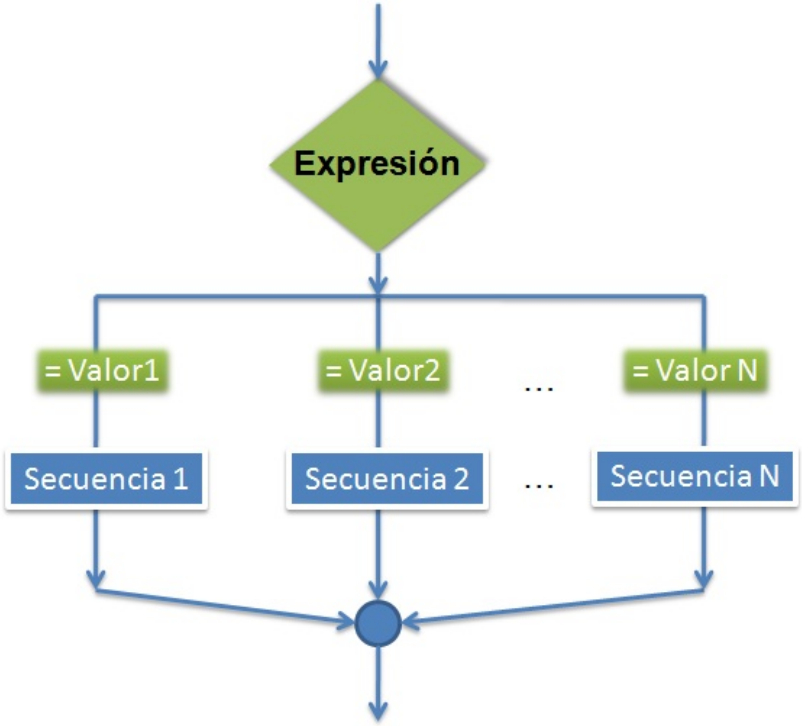

Imagen extraída de curso Programación del MECD.

Debes conocer

Estudia el siguiente fragmento de código en el que se resuelve el cálculo de la nota de un examen de tipo test, utilizando la estructura `switch`.

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package sentencias_condicionales;
/**
 *
 * @author Pc
 */
public class condicional_switch {

    /*Vamos a realizar el cálculo de la nota de un examen
     * de tipo test. Para ello, tendremos en cuenta el número
     * total de preguntas, los aciertos y los errores. Dos errores
     * anulan una respuesta correcta.
     *
     * La nota que vamos a obtener será un número entero.
     *
     * Finalmente, se muestra por pantalla la nota obtenida, así
     * como su calificación no numérica.
```

```
    *
    * La obtención de la calificación no numérica se ha realizado
    * utilizando la estructura condicional múltiple o switch.
    *
    */
   public static void main(String[] args) {
        // Declaración e inicialización de variables
       int num_aciertos = 17;
       int num_errores = 3;
       int num_preguntas = 20;
       int nota = 0;
       String calificacion="";

       //Procesamiento de datos
       nota = ((num_aciertos - (num_errores/2))*10)/num_preguntas;

       switch (nota) {
           case 5: calificacion="SUFICIENTE";
                   break;
           case 6: calificacion="BIEN";
                   break;
           case 7: calificacion="NOTABLE";
                   break;
           case 8: calificacion="NOTABLE";
                   break;
           case 9: calificacion="SOBRESALIENTE";
                   break;
           case 10: calificacion="SOBRESALIENTE";
                   break;
           default: calificacion="INSUFICIENTE";
       }

       //Salida de información
       System.out.println ("La nota obtenida es: " + nota);
       System.out.println ("y la calificación obtenida es: " + calificacion);
   }
}
```