UE#04

PROBLEMAS DE ANÁLISIS DE CASOS

CONTROL DE ALTERNATIVAS

Índice

- Ruptura de control.
- Sentencia `if` y sus variantes.
- Sentencia `switch`
- Usos y estilo

RUPTURA DE CONTROL

- En un bloque, la ejecución es secuencial: una orden detrás de otra según estén escritas de arriba a abajo.
- Los problemas de análisis de casos requieren la ruptura del control secuencial.
- Hay que elegir entre múltiples alternativas.
- Se ejecutará un código u otro en función de las condiciones preestablecidas.

SENTENCIA IF

- Formato del if_then:
  - **if (<<condición>>)**
  - ** <<Bloque>>;**
- La *condición* es una expresión booleana.
- Si el *bloque* contiene más de una sentencia debe encerrarse entre llaves.

SENTENCIA `IF`

- Funcionamiento del `if_then`:

- Si la *condición* se evalúa a `true`, se ejecuta *bloque* y si no (si se evalúa a `false`) no se ejecuta.

SENTENCIA IF

- Formato del if_then_else:

```
if (<<condición>>)
   <<Bloque1>>;
else
   <<Bloque2>>;
```

SENTENCIA `IF`

- Funcionamiento del `if_then_else`:
- Si la *condición* se evalúa a `true`, se ejecuta *bloque1* y si no (si se evalúa a `false`) se ejecuta *bloque2*.

SENTENCIA IF

- Anidamientos if_then_else:

```
if (<<condición1>>)
   <<Bloque1>>;
else if (<<condición2>>)
   <<Bloque2>>;
. . .
else
   <<BloqueN>>;
```

SENTENCIA `IF`

- Funcionamiento del `if_then_else` anidado:

- Si la *condición1* se evalúa a `true`, se ejecuta *bloque1* y si no, si la *condición2* se evalúa a `true` se ejecuta *bloque2* y así sucesivamente con todas las ramas que haya. El *bloqueN* se ejecutará solo si no se han satisfecho ninguna de las condiciones precedentes.

EJEMPLO DE USO

- Problema: “El mayor de tres números, todos distintos”. Modelo Heavy.

```
if ((x >= y) && (x >= z))
  return x;
else if ((y >= x) && (y >= z))
  return y;
else if ((z >= x) && (z >= y))
  return z;
else  // No hay nada qué hacer
  return z;
```

EJEMPLO DE USO

- Problema: “El mayor de tres números, todos distintos”. Modelo Hiper.

```
if (x >= y)
  if (x >= z)
    return x;
  else
    return z;
else if (y >= z)
  return y;
else
  return z;
```

EJERCICIOS DE `if`

- Ejercicio1: “El mayor de tres números, todos distintos”. Especificación y realización
- Ejercicio2:”Valor absoluto de un número”. Especificación y realización.

EJERCICIOS DE `if`

```
FUNCION max3Fino (Entero x, y, z) → Entero
PRE: cierto
                      | x  si (x >= y) /\ (x >= z)
POST: resultado =     | y  si (y >= y) /\ (y >= z)
                      | z  si (z >= x) /\ (z >= y)

int max3Fino (int x, int y, int z)
{
  if ((x >= y) && (x >= z))
    return x;
  else if (y >= z)
    return y;
  else
    return z;
}
```

EJERCICIOS DE `if`

- **FUNCION valorAbsoluto (Entero x) → Entero**
- **PRE: cierto**
- **| x si x >= 0**
- **POST: resultado = |**
- **| -x e.o.c**

```
int valorAbsoluto (int x)
{
  if (x >= 0)
    return x;
  else
    return -x;
}
```

SENTENCIA switch

```
switch (<<expresión>>)
{
   case <<valor1>> :
      <<Bloque1>;
      break;
   case <<valor2>> :
      <<Bloque2>;
      break;
   . . .
   default
      <<BloqueN>>;
}
```

SENTENCIA `switch`

- La *expresión* puede ser `int` o `char`.
- Los respectivos *bloques* no necesitan llaves.
- La cláusula `default` es opcional.
- Una orden `return` equivale a un `break`. Cuando exista la una, sobra la otra.

SENTENCIA `switch`

- Funcionamiento:
- Se evalúa la *expresión*.
- Si ajusta con alguno de los *valores*, se ejecuta el *bloque* correspondiente (explora de arriba abajo).
- Si no ajusta con ninguno, ejecuta el *bloque* de la cláusula `default`.

EJEMPLOS DE `switch`

- Problema:”Visualizar los nombres de los puntos cardinales”
- Se da nombre a cada punto:

```
final int     norte = 1;
final int  noreste = 2;
final int      este = 3;
final int  sureste = 4;
final int       sur = 5;
final int suroeste = 6;
final int     oeste = 7;
final int noroeste = 8;
```

EJEMPLOS DE `switch`

```
FUNCION verRumbo (Entero r) ---> Cadena
PRE: cierto
POST: res es la cadena de caracteres de la constante
      relativa a "r"
String verRumbo (int r)
{
  switch (r)
  {
     case 1  : return "Norte";
     case 2  : return "Noreste";
     case 3  : return "Este";
     case 4  : return "Sureste";
     case 5  : return "Sur";
     case 6  : return "Suroeste";
     case 7  : return "Oeste";
     default : return "Noroeste";
  }
}
```

EJERCICIOS DE Selección

- Ejercicio3: “Girar un rumbo 90º”
- Ejercicio4: “Girar un rumbo 45º”

EJERCICIOS DE Selección

- FUNCION gira90 (Entero rumbo) → Entero
- PRE: cierto
- POST: resultado es la constante que se corresponde con avanzar
- a la derecha 90 grados desde "rumbo".

```
int gira90 (int rumbo)
{
  if (rumbo == oeste)
    return norte;
  else if (rumbo == noroeste)
    return noreste;
  else
    return rumbo + 2;
}
```

EJERCICIOS DE Selección

- FUNCION gira45 (Entero rumbo) → Entero
- PRE: cierto
- POST: resultado es la constante que se
- corresponde con avanzar a la
- derecha 45 grados desde "rumbo".

```
int gira45 (int rumbo)
{
    return rumbo % 8 + 1;
}
```