



Introducción a la Programación

Grado en Ingeniería Informática

7. Más análisis de casos

Bloque 1. Léxico y organización de un algoritmo
7. Más análisis de casos

Dr. Isidro Verdú

Más análisis de casos

Ya conocemos...

Composición
Si - Entonces

```
SI cond ENTONCES  
  a  
FIN_SI;
```

Composición
Si - Entonces - Si_no

```
SI cond ENTONCES  
  a  
SI_NO  
  b  
FIN_SI;
```

donde:
cond: condición booleana
a,b: acciones

Más análisis de casos

Ya conocemos...

Composición Si - Entonces

```
LÉXICO
  x, y: Entero;
  max: Entero;
ALGORITMO
  Leer(x, y);

  SI (x ≥ y) ENTONCES
    max ← x;
  FIN_SI;

  SI (x < y) ENTONCES
    max ← y;
  FIN_SI;

  Escribir (max);
FIN.
```

Composición Si - Entonces - si_no

```
LÉXICO
  x, y: Entero;
  max: Entero;
ALGORITMO
  Leer(x, y);

  SI (x ≥ y) ENTONCES
    max ← x;
  SI_NO
    max ← y;
  FIN_SI;

  Escribir (max);
FIN.
```

3

Composición Segun

También existe...

Composición Segun

```
LÉXICO
  x, y: Entero;
  z: Entero;
ALGORITMO
  Leer(x, y);
  SEGÚN x, y
    x ≥ y : z ← x;
    x < y : z ← y;
  FIN_SEGÚN;
  Escribir (z);
FIN.
```

4

Composición Según

Sintaxis de la composición Según

```
SEGÚN  $c_1, c_2, \dots, c_n$   
   $e_1 : a_1$   
   $e_2 : a_2$   
  ...  
   $e_m : a_m$   
FIN_SEGÚN
```

```
SEGÚN  $c_1, c_2, \dots, c_n$   
   $e_1 : a_1$   
   $e_2 : a_2$   
  ...  
   $e_m : a_m$   
  EN_OTRO_CASO:  $a_{m+1}$   
FIN_SEGÚN
```

donde:

- c_i : variables que intervienen
- e_i : expresiones booleanas que expresan casos en función de los c_i
- a_i : acción que corresponde a e_i
- a_{m+1} se ejecuta si $e_1 \vee e_2 \dots \vee e_m = \text{FALSO}$

5

Composición Según

Según en C/C++: condiciones en cascada

```
SEGÚN  $c_1, c_2, \dots, c_n$   
   $e_1 : S_1$   
   $e_2 : S_2$   
  ...  
   $e_n : S_n$   
  EN OTRO CASO  $S_{n+1}$   
FIN_SEGÚN
```



En cascada:

```
if (condición1)  
  sentencia1;  
else if (condición2)  
  sentencia2;  
else if (condición3)  
  sentencia3;  
else  
  sentencia;
```

6

Composición Según

```
if (condición1)
{
    sentencia;
    sentencia;
    sentencia;
}
else if (condición2)
    sentencia;
else if (condición3)
{
    sentencia;
    sentencia;
    sentencia;
}
else
    sentencia;
```

Por supuesto, se usan
bloques cuando es
necesario

7

Composición Según

Según en C/C++: condiciones en cascada

```
SEGÚN  $c_1, c_2, \dots, c_n$ 
 $e_1: S_1$ 
 $e_2: S_2$ 
...
 $e_n: S_n$ 
EN OTRO CASO  $S_{n+1}$ 
FIN_SEGUN
```

```
If ( $e_1$ )  $S_1$ ;
else If ( $e_2$ )  $S_2$ ;
else If ( $e_3$ )  $S_3$ ;
...
else If ( $e_n$ )  $S_n$ ;
else  $S_{n+1}$ 
```

```
If ( $e_1$ ) { $S_1; S_2; \dots S_i$ }
else If ( $e_2$ ) { $S_1; S_2; \dots S_i$ }
else If ( $e_3$ ) { $S_1; S_2; \dots S_i$ }
...
else If ( $e_n$ ) { $S_1; S_2; \dots S_i$ }
else { $S_1; S_2; \dots S_i$ }
```

8

Según en C/C++: condicones en cascada

```
x=2;
scanf(" %f %d",&v);

if (v<0)
    f=2*x+4;
else if (v==0)
    f=x/2;
else if (v<=2.5)
    f=5*x;
else
    f=0;
```

9

Según en C/C++: condicones en cascada

```
x=2; scanf(" %f ",&v);

if (v<0)
    f=2*x+4;
else if (v==0)
    f=x/2;
else if ( v<=2.5)
    f=5*x;
else
    f=0;
```

```
x=2; scanf(" %f ",&v);

if (v<0) f=2*x+4;
if (v==0) f=x/2;
if ( v<=2.5) f=5*x;
if ( v>2.5) f=5*x;
```

¿Cuánto vale f si v = -1 ?

8 - 10

10

Condiciones anidadas en general

También se puede:

```
if (condición)
    if (condición2)
        sentencia;
    else
        sentencia;
else
    sentencia;
```

```
if (a>0)
    if (b>0) a=a+1;
    else
        if (c>0)
            if (a<5) b=b-1;
            else
                if (b<5)
                {
                    c=c+1;
                    a=2;
                }
                else a=a-1;
        else
            if (c<5) b=b-1;
            else c=c-1;
else
    a=0;
```

11

```
if (a>0)
    if (b>0) a=a+1;
    else
        if (c>0)
            if (a<5) b=b-1;
            else
                if (b<5) {c=c+1;a=2}
                else a=a-1;
        else
            if (c<5) b=b-1;
            else c=c-1;
else
    a=0;
```

Es muy importante
identificar
correctamente los
programas

12

```
if (a>0) if (b>0) a=a+1; else if (c>0) if (a<5)
b=b-1; else if (b<5) {c=c+1; a=2} else
a=a-1; else if (c<5) b=b-1; else c=c-1;
else a=0;
```

**Esto es lo mismo :
El ordenador lo entiende.
Nosotros no.**

13

Composición CASE: Selector

```
case expresión_ordinal of
  valor1 : sentencias1;
  valor2 : sentencias2;
  ...
  {otro_caso      sentenciasN}
end;
```

selector: para múltiples alternativas. Especialmente buena cuando es una variable o expresión que puede tomar distintos valores.

14

Selector (switch)

switch: para múltiples alternativas. Especialmente buena cuando es una variable o expresión que puede tomar distintos valores.

switch, case, break y *default* son palabras reservadas de C.

```
scanf("%d",opcion);  
switch (opcion+2)  
{  
    case 0:  
        a=a+1;  
        b=5;  
        break;  
    case 1:  
        c=a+1;  
        d=2;  
        break;  
    case 2:  
        c=b-1; b=4;  
        break;  
    default:  
        h=1; b=2;  
}
```

15

Selector (switch)

```
switch (selector)  
{  
    case etiqueta1 : sentencia; sentencia; ... sentencia; break;  
    case etiqueta2 : sentencia; sentencia; ... sentencia; break;  
    ...  
    case etiquetan : sentencia; sentencia; ... sentencia; break;  
    default : sentencia; sentencia; ... sentencia;  
}
```

El selector tiene que ser un *int* o *char*

default establece qué debe hacerse si no se da ningún caso.

break marca el final de las acciones de cada caso

16

Selector (switch)

break es opcional. Si no se pone, continúa ejecutando instrucciones hasta encontrar un break o el fin del switch.

Como no hay break, si introduzco un 0, la salida será:

Cero
Uno

Si introduzco un uno, la salida será:

Uno

```
printf("Mete 0 o 1:");  
scanf("%d",&a);
```

```
switch (a)  
{  
    case 0:  
        printf("Cero\n");  
    case 1:  
        printf("Uno\n");  
        break;  
    default:  
        printf("Otro\n");  
}
```

default es opcional, pero muy recomendable

17

Selector (switch)

Nos dice si el caracter es una vocal minúscula.

Cualquiera de los casos es vocal.

default en este caso es imprescindible.

```
scanf("%c",&caracter);  
switch (caracter)  
{  
    case 'a': case 'e': case 'i':  
    case 'o': case 'u':  
        printf("vocal");  
        break;  
    default:  
        printf("no vocal");  
}
```

18

