Práctica 3. Estructuras de control

Objetivos

- Adquirir la capacidad de diseñar algoritmos para la resolución de problemas básicos.
- Conocer cómo se traduce un algoritmo escrito en pseudocódigo en un lenguaje de programación.

1. Introducción

En la práctica anterior vimos qué etapas teníamos que seguir cuando programamos en lenguaje C:

- En papel, diseñar el algoritmo que resuelve el problema que queremos resolver.
- Traducir el algoritmo a lenguaje C. Para ello usaremos un editor de textos como por ejemplo el *gedit*.
- Compilar el programa para obtener un ejecutable. Vamos a utilizar el compilador *gcc*. El comando a utilizar sería: `$gcc fuente.c -o ejecutable`
- Ejecutar el programa y probar que su funcionamiento es el deseado. Emplearemos el comando `$./ejecutable`

2. Operaciones

En prácticas anteriores hemos visto que se puede operar con las variables:

- `a=b+c`; //Guarda en `a` la suma de `b` y `c`
- `a=b-c`; //Guarda en `a` la resta de `b` y `c`

- `a=b*c`; //Guarda en `a` el producto de `b` y `c`
- `a=b/c`; //Guarda en `a` la división de `b` y `c`
- `a=b%c`; //Guarda en `a` el resto de la división entera de `b` entre `c`

Cuando el operando es el mismo que el destino, se puede abreviar de la siguiente forma:

- `a=a+b`;
- `a+=b`;

Ambas expresiones son equivalentes.

Otras expresiones abreviadas que se pueden realizar son las siguientes:

- `a++`; // Equivale a `a=a+1`;
- `a--`; // Equivale a `a=a-1`;
- `b=a++`; // Equivale a `b=a; a=a+1`;
- `b=++a`; // Equivale a `a=a+1; b=a`;

3. Estructuras de control selectivas

Nos permiten tomar decisiones. Hay varios tipos:

- Alternativa simple
- Alternativa doble
- Alternativa múltiple

3.1. Alternativa simple

Si se cumple una condición, se ejecutan unas instrucciones. Si no, no se ejecutan (ver Fig. 1).

```
if (condición){
     //instrucciones
}
//instrucciones
```

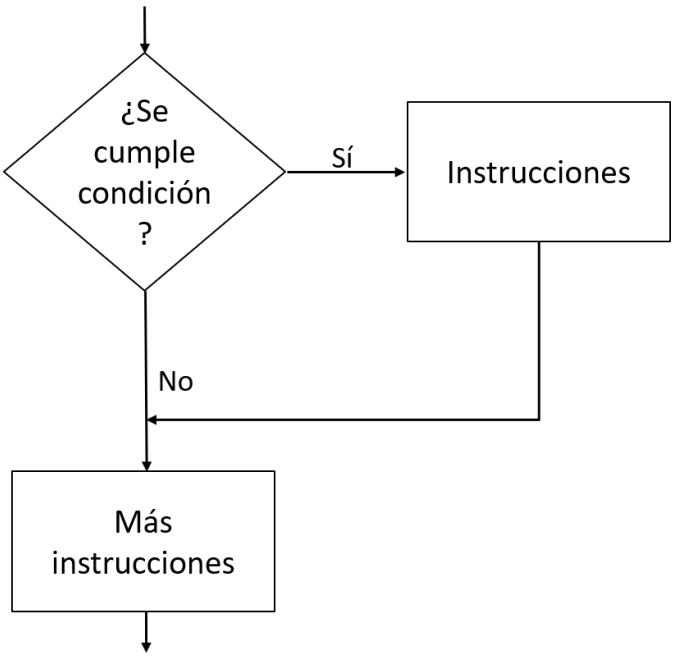

Figura 1: Flujograma de una estructura if

3.2. Alternativa doble

Si se cumple una condición, se ejecutan unas instrucciones. Si no se cumple, se ejecutan otras instrucciones (ver Fig. 2).

```
if (condición){
     //instrucciones
}
else{
     //instrucciones
}
//instrucciones
```

3.3. Alternativa múltiple

En función del valor de una condición se ejecutan distintos tipos de instrucciones. En C se puede hacer de dos formas, con if-else anidados:

```
if (condición){
     //instrucciones
}
else if(condición){
```

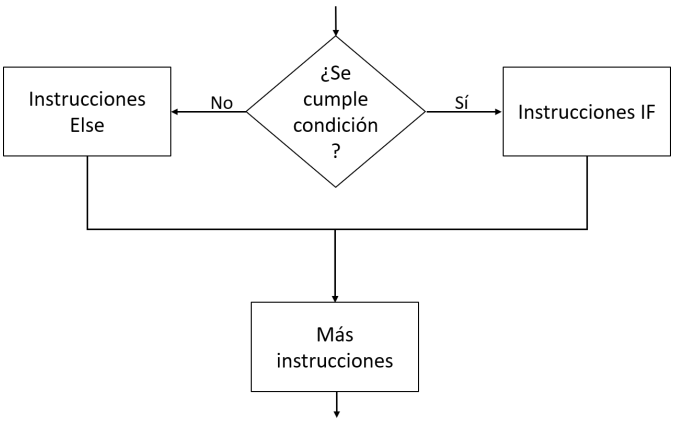

Figura 2: Flujograma de una estructura if-else

```
     //instrucciones
} else if(condición){
     //instrucciones
} else if(condición){
     //instrucciones
} else {
     //instrucciones
}
//instrucciones
```

O bien usando la estructura switch:

```
switch (variable){
     case 1:
         //instrucciones si variable vale 1
         break;
     case 2:
         //instrucciones si variable vale 2
         break;
     ...
     default:
         //instrucciones si variable no vale nada de lo anterior
}
//instrucciones
```

3.4. Condiciones

Para evaluar una condición, podemos usar los operadores de comparación:

- Mayor o igual que (a>=b)
- Mayor que (a>b)
- Igual que (a==b)
- Distinto que (a!=b)
- Menor o igual que (a<=b)
- Menor que (a<b)
- Que se cumpla una condición y otra condición (condición1) && (condición2)
- Que se cumpla una condición u otra condición(condición1) || (condición2)
- Que no se cumpla una condición !(condición)

4. Estructuras de control iterativas: bucles

Permiten definir fragmentos de un programa que se repiten (bucles). Hay que controlar que no sea un bucle infinito:

- Controlando una condición en cada iteración.
- Contando el número de iteraciones (uso de contadores).

La estructura de los bucles puede ser:

- Mientras se cumpla una condición, haz unas instrucciones.
- Haz unas instrucciones mientras se cumpla una condición.
- Haz unas instrucciones desde una condición inicial hasta que se alcance una condición final.

4.1. Bucle while

La estructura sería la que se muestra en la Fig. 3, siendo en pseudocódigo:

```
MIENTRAS condición HACER
 instrucciones
FINMIENTRAS
```

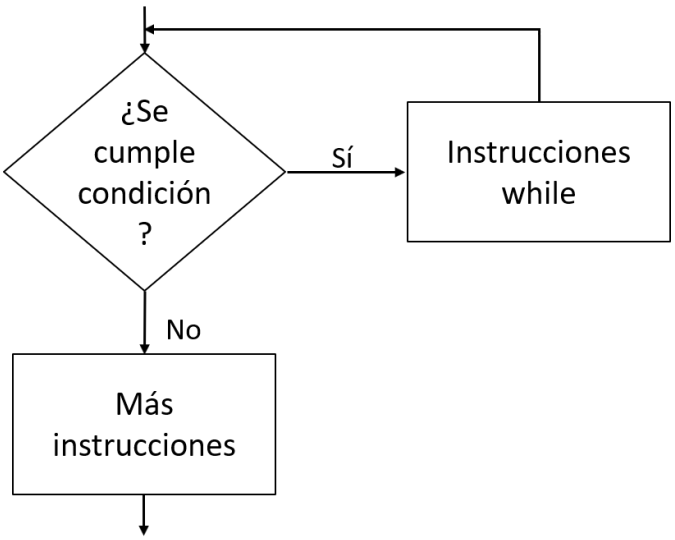

Figura 3: Flujograma de una estructura while

Y en C corresponde con:

```
while (condición){
   //instrucciones
}
```

Las instrucciones se ejecutan 0 o más veces. Un ejemplo sería el siguiente:

```
int n=3;
while (n>0){
   printf("Iteración%d", n);
   n--;
}
```

Se ejecuta 3 veces el bucle, para `n` con valor 3, 2 y 1.

4.2. Bucle do-while

La estructura sería (Fig. 4):

```
HACER
 instrucciones
MIENTRAS condición
```

Y en C corresponde con:

```
do{
   //instrucciones
}while (condición);
```

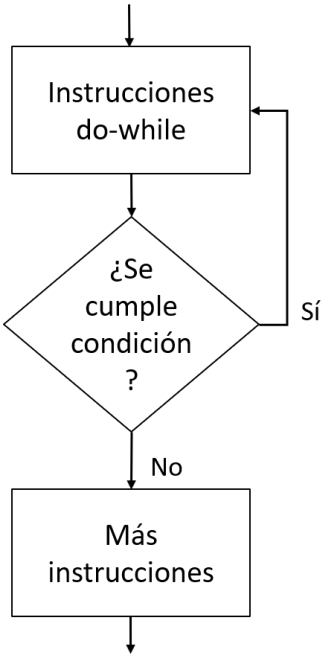

Figura 4: Flujograma de una estructura do-while

Las instrucciones se ejecutan 1 o más veces. Un ejemplo sería el siguiente:

```
int n=3;
do{
   printf("Iteración%d", n);
   n--;
}while (n>0);
```

Se ejecuta 3 veces el bucle, para `n` con valor 3, 2 y 1.

4.3. Bucle for

La estructura sería:

```
PARA condición_inicial MIENTRAS condición HACER
 instrucciones
FINPARA
```

Y en C corresponde con:

```
for(condición_inicial; condición; actualización_contadores){
   //instrucciones
}
```

Las instrucciones se ejecutan 0 o más veces.

Un ejemplo sería el siguiente:

```
int n;
for(n=3;n>0;n--){
   printf("Iteración%d", n);
}
```

Se ejecuta 3 veces el bucle, para `n` con valor 3, 2 y 1.

5. Algoritmos

5.1. Algoritmo Detecta_tres

El algoritmo siguiente detecta si el número introducido por teclado es el 3:

```
ALGORITMO Detecta_tres
 ENTRADAS:
  num: Entero ; Número que se lee
 SALIDAS:
 VARIABLES:
  num: Entero
 INICIO
  ESCRIBA "Escribe un número: "
  LEA num
  SI num == 3 ENTONCES
    ESCRIBA "El número es el 3"
  SINO
    ESCRIBA "El número no es el 3"
  FINSI
  ESCRIBA "Fin del algoritmo"
 FIN
```

5.2. Algoritmo Calcula_nota

El algoritmo siguiente convierte una nota numérica en su correspondiente calificación:

```
ALGORITMO Calcula_nota
 ENTRADAS:
  num: Entero ; Nota numérica
 SALIDAS:
 VARIABLES:
  num: Entero ; Nota numérica
```

```
INICIO
 ESCRIBA "Escribe la nota numérica: "
 LEA num
 SI num == 5 O num == 6 ENTONCES
   ESCRIBA "Aprobado"
 SINO SI num == 7 O num == 8 ENTONCES
         ESCRIBA "Notable"
      SINO SI num == 9 O num == 10 ENTONCES
              ESCRIBA "Sobresaliente"
           SINO
              ESCRIBA "Suspenso"
           FINSI
      FINSI
 FINSI
FIN
```

5.3. Algoritmo Calcula_nota_v2

El algoritmo siguiente convierte una nota numérica en su correspondiente calificación:

```
ALGORITMO Calcula_nota_v2
 ENTRADAS:
  num: Entero ; Nota numérica
 SALIDAS:
 VARIABLES:
  num: Entero ; Nota numérica
 INICIO
  ESCRIBA "Escribe la nota numérica: "
  LEA num
  SEGÚN num HACER
    CASO 5: ESCRIBA "Aprobado"
    CASO 6: ESCRIBA "Aprobado"
    CASO 7: ESCRIBA "Notable"
    CASO 8: ESCRIBA "Notable"
    CASO 9: ESCRIBA "Sobresaliente"
    CASO 10: ESCRIBA "Sobresaliente"
    SINO
      ESCRIBA "Suspenso"
 FINSEGUN
FIN
```

5.4. Algoritmo Suma_n_primeros_nums_v1

El algoritmo siguiente calcula la suma de los *n* primeros números enteros:

```
ALGORITMO Suma_n_primeros_nums_v1
 ENTRADAS:
  num: Entero ; Números a sumar
 SALIDAS:
  total: Entero ; Suma de los n primeros números
 VARIABLES:
  num: Entero
  total: Entero
  i: Entero ; Contador
 INICIO
  ESCRIBA "Escribe cuantos números quieres sumar: "
  LEA num
  SI num ≥ 1 ENTONCES
    total ← 0
    i ← 1
    MIENTRAS i ≤ num HACER
      Total ← total + i
      i ← i + 1
    FINMIENTRAS
    ESCRIBA "La suma es: "
    ESCRIBA total
  SINO
    ESCRIBA "El número ha de ser mayor o igual a 1"
  FINSI
 FIN
```

5.5. Algoritmo Potencia

El algoritmo siguiente calcula la potencia de un número elevado a un exponente:

```
ALGORITMO Potencia
 ENTRADAS:
  base: Entero ; Número leído (base)
  exp: Entero ; Número leído (exponente)
 SALIDAS:
  pot: Entero ; Potencia (Base elevado a Exp)
 VARIABLES:
  base: Entero
```

```
     exp: Entero
     pot: Entero
     i: Entero
    INICIO
     ESCRIBA "Escribe un número (base): "
     LEA base
     ESCRIBA "Escribe un número (exponente): "
     LEA exp
     SI exp ≥ 1 Y base ≥ 1 ENTONCES
       pot ← 1
       i ← 1
       MIENTRAS exp ≥ 1 HACER
         pot ← pot * base
         exp ← exp - 1
       FINMIENTRAS
       ESCRIBA "La potencia es: "
       ESCRIBA pot
     SINO
       ESCRIBA "La base y el exponente han de ser mayores o iguales a
1"
     FINSI
    FIN
```

6. Traducción a C

6.1. Programa Detecta_tres

Si traducimos el algoritmo Detecta_tres a lenguaje C nos quedaría el programa siguiente:

```
//Librería que contiene las funciones scanf y printf
#include <stdio.h>

//Función principal del programa
int main ()
{
  // Este programa indica si el número leído por teclado es 3

  // Declaro las variables de mi función
  int num;
```

```
  //Sustituyo la función ESCRIBA "cadena" por printf
  printf("Escribe un número: ");

  //Sustituyo la función LEA num por scanf ("%d", &variableEntera);
  scanf("%d", &num); //Guarda el número leído en la variable num

  //Compruebo si el número introducido es el 3
  if (num == 3){
    //Sustituyo la función ESCRIBA "cadena" por printf
    printf("El número introducido es el 3\n");
  }
  else {
    //Sustituyo la función ESCRIBA "cadena" por printf
    printf("El número introducido no es el 3\n");
  }

  //Sustituyo la función ESCRIBA "cadena" por printf
  printf("Fin del algoritmo\n");

  //Fin del programa
  return 0;
}
```

6.2. Programa Calcula_nota

La traducción del algoritmo Calcula_nota a lenguaje C sería la siguiente:

```
//Librería que contiene las funciones scanf y printf
#include <stdio.h>

//Función principal del programa
int main ()
{
  // Este programa convierte una nota numérica a su
  // calificación correspondiente

  // Declaro las variables de mi función
  int num;

  //Sustituyo la función ESCRIBA "cadena" por printf
```

```
  printf("Escribe la nota numérica: ");

  //Sustituyo la función LEA num por scanf ("%d", &variableEntera);
  scanf("%d", &num); //Guarda el número leído en la variable num

  //Compruebo qué nota es para imprimir la cadena correspondiente
  if ((num == 5) || (num == 6)){
    //Sustituyo la función ESCRIBA "cadena" por printf
    printf("Aprobado");
  }
  else{
      if ((num == 7) || (num == 8)){
          //Sustituyo la función ESCRIBA "cadena" por printf
          printf("Notable");
      }
      else{
          if ((num == 9) || (num == 10)){
              //Sustituyo la función ESCRIBA "cadena" por printf
              printf("Sobresaliente");
          }
          else{
              //Sustituyo la función ESCRIBA "cadena" por printf
              printf("Suspenso");
          }
      }
  }

  //Fin del programa
  return 0;
}
```

6.3. Programa Calcula_nota_v2

La traducción del algoritmo Calcula_nota_v2 a lenguaje C sería la siguiente:

```
//Librería que contiene las funciones scanf y printf
#include <stdio.h>

//Función principal del programa
int main ()
{
```

```
  // Este programa convierte una nota numérica a su
  // calificación correspondiente

  // Declaro las variables de mi función
  int num;

  //Sustituyo la función ESCRIBA "cadena" por printf
  printf("Escribe la nota numérica: ");

  //Sustituyo la función LEA num por scanf ("%d", &variableEntera);
  scanf("%d", &num); //Guarda el número leído en la variable num

  //Compruebo qué nota es para imprimir la cadena correspondiente
  switch (num){
    //Sustituyo la función ESCRIBA "cadena" por printf

    //Imprimo lo mismo si la nota es 5 o si es 6
    case 5:
    case 6: printf("Aprobado\n");
            break; //Sale del switch porque ya entró en un caso

    //Imprimo lo mismo si la nota es 7 o si es 8
    case 7:
    case 8: printf("Notable\n");
            break; //Sale del switch porque ya entró en un caso

    //Imprimo lo mismo si la nota es 9 o si es 10
    case 9:
    case 10: printf("Sobresaliente\n");
            break; //Sale del switch porque ya entró en un caso
    //Si no es ningún valor de los anteriores, Suspenso
    default:
            printf("Suspenso\n");
  }

  //Fin del programa
  return 0;
}
```

6.4. Programa Suma_n_primeros_nums_v1

Si traducimos el algoritmo Suma_n_primeros_nums_v1 a lenguaje C nos quedaría el programa siguiente:

```
//Librería que contiene las funciones scanf y printf
#include <stdio.h>

//Función principal del programa
int main ()
{
  // Este programa calcula la suma de los primeros n números

  // Declaro las variables de mi función
  int num, total, i;

  //Sustituyo la función ESCRIBA "cadena" por printf
  printf("Escribe cuantos números quieres sumar: ");

  //Sustituyo la función LEA num por scanf ("%d", &variableEntera);
  scanf("%d", &num); //Guarda el número leído en la variable num

  //Inicializo las variables
  total = 0;
  i = 1;

  //Compruebo si el número introducido es mayor o igual que 1
  if (num >= 1) {
    //Hago el bucle para sumar hasta que i valga n
    while (i <= num){
      //Calculo la suma parcial
      total = total + i;
      i = i + 1;
    }

    //Sustituyo la función ESCRIBA "cadena" por printf
    printf("La suma es:%d \n", total);
  }
  else{
    //Sustituyo la función ESCRIBA "cadena" por printf
    printf("El número ha de ser mayor o igual a 1");
  }
```

```
    //Fin del programa
    return 0;
  }
```

6.5. Programa Potencia

La traducción del algoritmo Potencia a lenguaje C sería la siguiente:

```
  //Librería que contiene las funciones scanf y printf
  #include <stdio.h>

  //Función principal del programa
  int main ()
  {
    // Este programa calcula la potencia de un número

    // Declaro las variables de mi función
    int base, exponente, pot, i;

    //Sustituyo la función ESCRIBA "cadena" por printf
    printf("Escribe un número (base): ");

    //Sustituyo la función LEA base por scanf ("%d", &variableEntera);
    scanf("%d", &base); //Guarda el número leído en la variable base


    //Sustituyo la función ESCRIBA "cadena" por printf
    printf("Escribe un número (exponente): ");

    //Sustituyo la función LEA exp por scanf ("%d", &variableEntera);
    scanf("%d", &exponente); //Guarda el número leído en la variable
exponente

    //Compruebo que la base y el exponente sean mayores que 1
    if ((base >= 1) && (exponente >= 1)){
      //Inicializo las variables
      pot = 1;
      i = 1;
      while (exponente >= 1){
        pot = pot * base;
        exponente = exponente - 1;
```

```
      }
      //Sustituyo la función ESCRIBA por printf
      printf("La potencia es:%d", pot);
    }
    else{
      //Sustituyo la función ESCRIBA "cadena" por printf
      printf("La base y el exponente han de ser mayores o iguales a
1");
    }

    //Fin del programa
    return 0;
  }
```

7. Ejercicios propuestos

7.1. Ejercicio 1

Escribe en un archivo con extensión `.c` el programa `Detecta_tres`. Compílalo con el compilador `gcc`. Después, ejecútalo para comprobar su funcionamiento.

7.2. Ejercicio 2

Escribe en un archivo con extensión `.c` el programa `Calcula_nota`. Compílalo con el compilador `gcc`. Después, ejecútalo para comprobar su funcionamiento.

7.3. Ejercicio 3

Escribe en un archivo con extensión `.c` el programa `Calcula_nota_v2`. Compílalo con el compilador `gcc`. Después, ejecútalo para comprobar su funcionamiento.

7.4. Ejercicio 4

Crea un programa que lea un número y escriba el mes al que corresponde imprimiendo su nombre por pantalla. Si es el 1, habrá que imprimir Enero, si es el 2, Febrero, etc. Si no es un número del 1 al 12 debe imprimir un mensaje de error.

7.5. Ejercicio 5

Escribe en un archivo con extensión `.c` el programa `Suma_n_primeros_nums_v1`. Compílalo con el compilador `gcc`. Después, ejecútalo para comprobar su funcionamiento. ¿Cómo sería el código usando una estructura de tipo *do-while*? ¿Y un bucle *for*?

7.6. Ejercicio 6

Escribe en un archivo con extensión `.c` el programa `Potencia`. Compílalo con el compilador `gcc`. Después, ejecútalo para comprobar su funcionamiento. ¿Cómo sería el código usando una estructura de tipo *do-while*? ¿Y un bucle *for*?

7.7. Ejercicio 7

Realice un programa `volumen.c` en lenguaje C que calcule volúmenes de figuras geométricas.

- La función `main` presentará un menú para saber si se quiere calcular el volumen de un cono (1) o el volumen de ortoedro (2). En función de la opción elegida, se leerán los datos necesarios y se calculará el volumen del cono o el volumen del ortoedro, imprimiendo el resultado por pantalla.
- Volumen del cono = $(1/3 * PI * radio^2 * altura)$
- Volumen del ortoedro = $(lado1 * lado2 * altura)$

7.8. Ejercicio 8

Modifica el programa anterior para que el menú incluya una opción 3 “Salir”. Se mostrará el menú y mientras la opción elegida no sea la 3 no se saldrá del programa. Es decir, se muestra el menú, si la opción es 1 ó 2 se calcula el volumen, se imprime por pantalla y vuelve a mostrar el menú. Si la opción es distinta de 1, 2 y 3 da un mensaje de error y vuelve a mostrar el menú. Si la opción es 3, sale del programa.

7.9. Ejercicio 9

Realice un programa en C que lea dos números y compruebe que el primero es menor que el segundo. Si no es así dará un mensaje de error y los volverá

a leer. Si los números son correctos, imprimirá todos los números pares desde el mayor hasta el menor por pantalla. Por ejemplo si se lee el 3 y el 7, ha de imprimir 6, 4.