

Práctica 5. Programación Modular

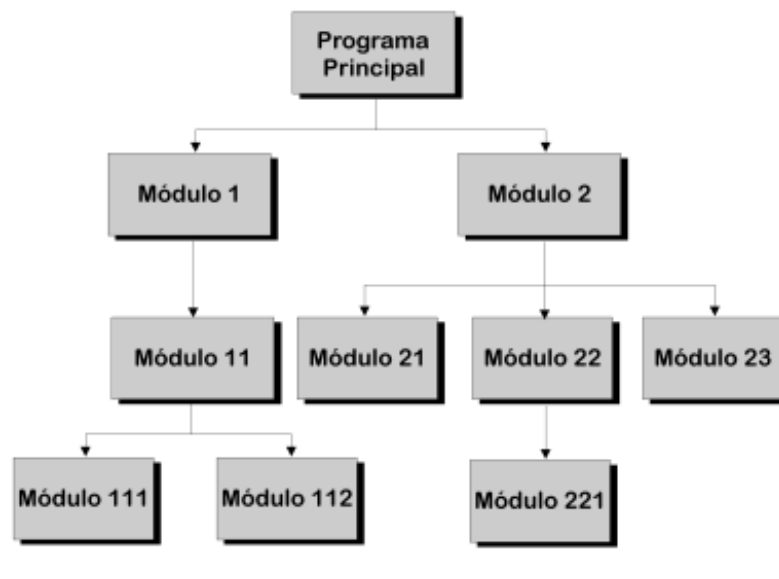
Objetivos:

- Familiarizarse con la descomposición descendente de problemas.
- Comprender las diferencias entre procedimientos y funciones.
- Comprender las diferencias entre el paso de parámetros por valor y por referencia.
- Comprender el concepto de retorno de valores en funciones.
- Afianzar el uso de subalgoritmos en la resolución de problemas.

1. Sentencias de control iterativas

- 📖 No olvides que antes de pasar a la fase de diseño deberemos analizar exhaustivamente el problema.

El diseño descendente resuelve un problema efectuando descomposiciones en otros problemas más sencillos a través de distintos niveles de refinamiento. La programación modular consiste en resolver de forma independientemente los subproblemas resultantes de una descomposición. El resultado de dividir reiteradas veces un problema en problemas más pequeños es una estructura jerárquica o en árbol.



2. Ejercicio resuelto 1. Repasa

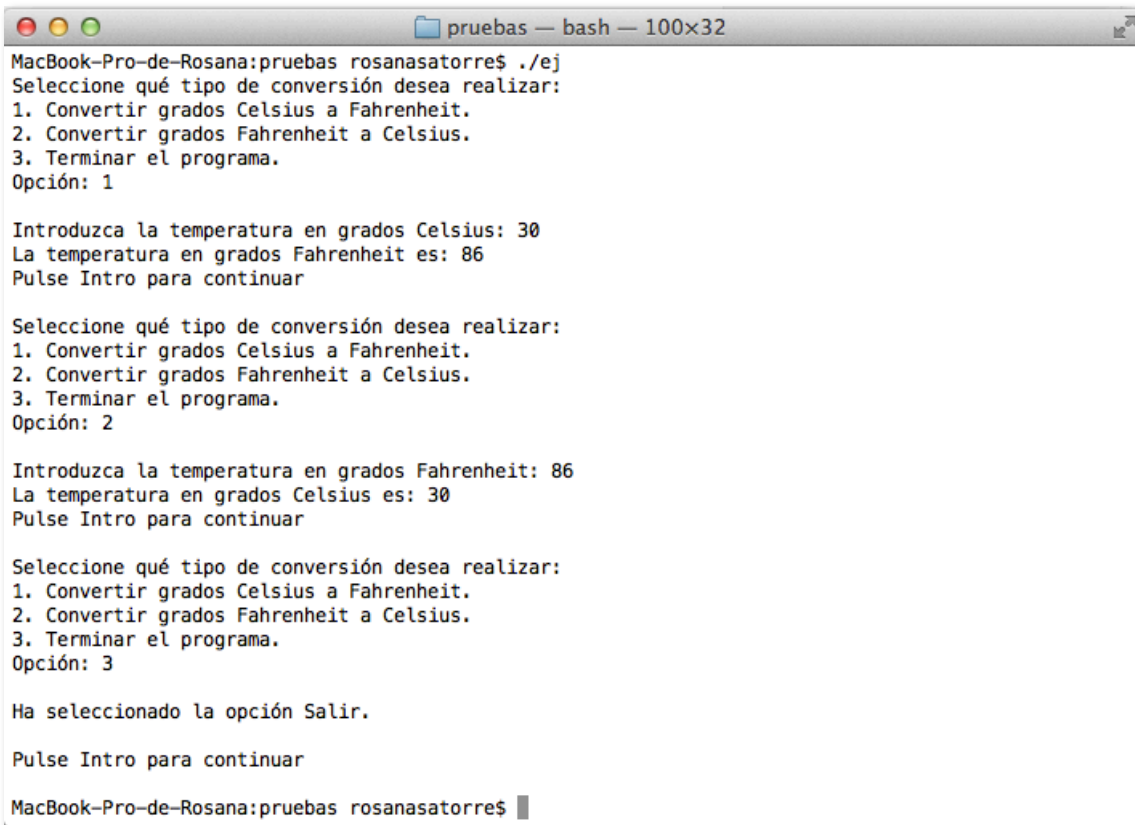
- Realiza un programa en C que permita convertir grados Celsius a Fahrenheit y viceversa. El programa debe mostrar un menú para poder seleccionar qué opción se desea (más la opción terminar). Y preguntar la temperatura que desea convertir.

Cada una de las conversiones se debe realizar por medio de una función.

La conversión de grados Celsius a grados Fahrenheit se obtiene multiplicando la temperatura en Celsius por 1,8 y sumando 32.

La conversión de grados Fahrenheit a grados Celsius se obtiene restándole 32 a la temperatura en grados Fahrenheit y dividiéndolo por 1,8.

Ejemplo de ejecución:



The screenshot shows a terminal window titled "pruebas — bash — 100x32". The user is at a prompt "MacBook-Pro-de-Rosana:pruebas rosanasatorre\$". They run a program that prompts them to select a conversion type. The user selects option 1 (Celsius to Fahrenheit), enters 30, and the program outputs 86. Then, the user selects option 2 (Fahrenheit to Celsius), enters 86, and the program outputs 30. Finally, the user selects option 3 (Exit), and the program outputs "Ha seleccionado la opción Salir." and prompts for a key press.

```
MacBook-Pro-de-Rosana:pruebas rosanasatorre$ ./ej
Seleccione qué tipo de conversión desea realizar:
1. Convertir grados Celsius a Fahrenheit.
2. Convertir grados Fahrenheit a Celsius.
3. Terminar el programa.
Opción: 1

Introduzca la temperatura en grados Celsius: 30
La temperatura en grados Fahrenheit es: 86
Pulse Intro para continuar

Seleccione qué tipo de conversión desea realizar:
1. Convertir grados Celsius a Fahrenheit.
2. Convertir grados Fahrenheit a Celsius.
3. Terminar el programa.
Opción: 2

Introduzca la temperatura en grados Fahrenheit: 86
La temperatura en grados Celsius es: 30
Pulse Intro para continuar

Seleccione qué tipo de conversión desea realizar:
1. Convertir grados Celsius a Fahrenheit.
2. Convertir grados Fahrenheit a Celsius.
3. Terminar el programa.
Opción: 3

Ha seleccionado la opción Salir.

Pulse Intro para continuar

MacBook-Pro-de-Rosana:pruebas rosanasatorre$
```

2.1 Solución propuesta

```
1  #include<iostream>
2  using namespace std;
3
4  void mostrar_menu();
5  float Celsius_to_Fahrenheit(float);
6  float Fahrenheit_to_Celsius(float);
7
8  main() {
9      int opcion;
10     float celsius, fahr;
11     do{
12         mostrar_menu();
13         cout<<"Opción: ";
14         cin>>opcion;
15         cout<<endl;
16         switch(opcion){
```

```
17     case 1: cout<<"Introducir temperatura en grados Celsius:
    ";
18         cin>>celsius;
19         fahr = Celsius_to_Fahrenheit(celsius);
20         cout<<"La temperatura en grados Fahrenheit es:
    "<<fahr;
21         break;
22     case 2: cout<<"Introduzca la temperatura en grados
    Fahrenheit: ";
23         cin>>fahr;
24         celsius = Fahrenheit_to_Celsius(fahr);
25         cout<<"La temperatura en grados Celsius es:
    "<<celsius;
26         break;
27     case 3: cout<<"Ha seleccionado la opción Salir."<<endl;
28         break;
29     default: cout<<"Debe introducir una opción de 1 a
    3"<<endl;
30     }
31     cout<<endl;
32     cin.get();
33     cout<<"Pulse Intro para continuar"<<endl;
34     cin.get();
35     }while(opcion!=3);
36 }
37
38 void mostrar_menu() {
39     cout<<"Seleccione qué tipo de conversión desea
    realizar:"<<endl;
40     cout<<"1. Convertir grados Celsius a Fahrenheit."<<endl;
41     cout<<"2. Convertir grados Fahrenheit a Celsius."<<endl;
42     cout<<"3. Terminar el programa."<<endl;
43 }
44
45 float Celsius_to_Fahrenheit(float cel){
46     float fah;
47
48     fah = cel * 1.8 + 32;
49     return(fah);
```

```
50 }  
51  
52 float Fahrenheit_to_Celsius(float fah)  
53 {  
54     float cel;  
55  
56     cel = (fah-32)/1.8;  
57     return (cel);  
58 }
```

3. Ejercicio resuelto 2. Repasa

► Dado el siguiente enunciado:

Calcular la representación en base 10 de un número a partir de su representación en base

2. Por ejemplo, $\text{bin2dec}(1010) = 10$. Se pide:

- Definir una función que calcule ese proceso.
- Utilizar dicha función correctamente en un programa que interaccione con el usuario.
- Definir un conjunto de pruebas para el problema.

3.1 Solución apartado 1

```
21 int BinarioADecimal(int n) {  
22     int n_decimal=0, coef=1;  
23  
24     while (n!=0) {  
25         n_decimal += coef*(n%10);  
26         coef *= 2;  
27         n = n / 10;  
28     }  
29     return (n_decimal);  
30 }
```

3.2 Solución apartado 2

```
1  #include <iostream>  
2  using namespace std;  
3  
4  int BinarioADecimal(int); //Declaración de la función
```

```
5
6  main()  {
7      // Declaración de variables
8      int num2,num10;
9
10     // Entrada de datos
11     cout << "Introduzca un numero en base 2: " << endl;
12     cin >> num2;
13     cout << endl;
14
15     // Llamada a la función
16     num10 = BinarioADecimal(num2);
17
18     // Mostrar Resultados
19     cout << "El numero " << num2 << " en base 2 equivale en
    base 10 a " << num10 << endl;
20 }
```

3.3 Solución apartado 3

Conjunto de pruebas:

Entrada (número en base 2 - binario)	Salida esperada (número en base 10 - decimal)
0	0
1	1
110	6
111	7
00110	16

4. Ejercicio resuelto 3. Repasa

- Realiza un programa que muestre un menú en el que a partir de un número entero n se elegirá una de las siguientes operaciones:
- Opción 1: **Calcular de cuántas cifras se compone.** (No contar los ceros a la izquierda)
 - Opción 2: **Mostrar la cifra i -ésima de dicho número.** La posición i debe pedirse al usuario.
 - Opción 3: **Salir.**

4.1 Solución

```
1  #include <iostream>
2  using namespace std;
3
4  //Declaración de constantes
5  const int  OPCION_SALIR = 3;
6
7  //Declaración de funciones
8  int  CantidadCifras(int);
9  int  Cifra_i(int, int);
10 void menu();
11
12 //Función principal
13 main()  {
14     int opcion, n, resultado, i;
15
16     do {
17         menu();
18         cout << "Introduzca la opción: ";
19         cin >> opcion;
20         switch(opcion)  {
21             case 1: cout << "Introduzca un número: ";
22                     cin >> n;
23                     resultado=CantidadCifras(n);
24                     cout <<"El número de cifras de "<<n<<" es
25                     "<<resultado<<endl;
26                     break;
27             case 2: cout << "Introduzca un número: ";
28                     cin >> n;
29                     cout <<"Introduce la i: ";
30                     cin >>i;
31                     resultado=Cifra_i(n,i);
32                     cout << "La cifra "<<i<<" de "<<n<<" es
33                     "<<resultado<<endl;
34                     break;
35             case OPCION_SALIR:
36                 cout << "FIN DEL PROGRAMA"<<endl;
37         }
38     }
39 }
```

```
36     cout<<"Pulsa Intro para continuar"<<endl;
37     cin.get();
38     cin.get();
39     } while(opcion != OPCION_SALIR);
40 }
41
42 //Función que calcula la cantidad de cifras
43 int CantidadCifras(int m) {
44     int contador=1;
45
46     while(m >=10) {
47         m=m/10;
48         contador ++;
49     }
50     return(contador);
51 }
52
53 //Función que calcula la cifra i-ésima del número
54 int Cifra_i(int m,int i) {
55     int contador=1,cifra=0,devolver;
56
57     cifra=m%10;
58     while(m >=10 && contador<i) {
59         m=m/10;
60         cifra=m%10;
61         contador ++;
62     }
63     if (contador==i)
64         devolver=cifra;
65     else
66         devolver=-1;
67     return(devolver);
68 }
69
70 //Función que muestra el menú
71 void menu() {
72     cout << "1.  Calcular el número de cifras de un
        número"<<endl;
```

```

73     cout << "2.  Mostrar la cifra i-ésima de un
        número"<<endl;
74     cout << "3.  SALIR"<<endl;
75 }

```

5. Ejercicio resuelto 4. Repasa

► Dados los siguientes algoritmos, responde a las cuestiones siguientes:

Algoritmo A

```
#include <iostream>
```

```
using namespace std;
```

```
void numMayor(int , int );
```

```
//Declaración de variable global
```

```
int mayor;
```

```
main() {
```

```
    int num1, num2;
```

```
    mayor = 0;
```

```
    cout << "Introduce dos números: ";
    "
```

```
    cin >> num1 >> num2;
```

```
    numMayor(num1,num2);
```

```
    cout << "El mayor es numMayor(num1,num2);
    "<<endl;
```

```
}
```

Algoritmo B

```
#include <iostream>
```

```
using namespace std;
```

```
void numMayor(int , int );
```

```
//Declaración de variable global
```

```
int resultado;
```

```
main() {
```

```
    int num1, num2;
```

```
    resultado = 0;
```

```
    cout << "Introduce dos números: ";
```

```
    cin >> num1 >> num2;
```

```
    resultado = num1 + num2 ;
```

```
    numMayor(num1,num2);
```

```
    cout << "Al sumar num1 y num2 da
    "<<resultado<<endl;
```

```
}
```

```
void numMayor(int n1, int n2)
```

```
{
```

```
    if (n1 > n2)
```

```
void numMayor(int n1, int n2)
```

```
{
```

```
    if (n1 > n2)
```

```
        mayor = n1;                                resultado = n1;
    else                                           else
        mayor = n2;                                resultado = n2;
    }
```

- 1. ¿Qué ocurre? ¿Funcionan correctamente? Realiza una traza de los algoritmos anteriores para comprobar su funcionamiento.
- 2. ¿Crees que deberíamos modificar los algoritmos anteriores? ¿Por qué? ¿Cómo?

5.1 Solución apartado 1

Traza del algoritmo A:

supuesta la entrada:	Programa Principal			procedimiento	
	num1	num2	mayor	n1	n2
	-	-	0		
	4	6		4	6
			6		

Llevaría a la ejecución:

```
MacBook-Pro-de-Rosana:pruebas rosanasatorre$ ./ej
Introduce dos números: 4 6
El mayor es 6
MacBook-Pro-de-Rosana:pruebas rosanasatorre$ █
```

En este programa desde el procedimiento numMayor se está modificando una variable global (mayor) sin que se hubiese pasado como parámetro a dicho procedimiento, aunque se produzca un resultado correcto podría haberse producido un efecto lateral. Esto es incorrecto.

Traza del algoritmo B:

supuesta la entrada:

Programa Principal			procedimiento	
num1	num2	resultado	n1	n2
-	-	0		
4	6			
		10		
			4	6
		6		

llevaría a la ejecución:

```
MacBook-Pro-de-Rosana:pruebas rosanasatorre$ ./ej
Introduce dos números: 4 6
Al sumar num1 y num2 da 6
MacBook-Pro-de-Rosana:pruebas rosanasatorre$
```

En este programa desde el procedimiento `numMayor` se está modificando una variable global sin que se hubiese pasado como parámetro a dicho procedimiento, como resultado se ha modificado la variable `resultado` que contenía un valor anteriormente (el de la suma de dos números) y por tanto se visualiza por pantalla un valor incorrecto. A esto se le denomina efecto lateral. Esto es incorrecto.

5.2 Solución apartado 2

Sí, los dos, aunque sólo uno produzca un resultado incorrecto. Deberíamos pasar como parámetro por referencia una variable que almacenase el número mayor tras la ejecución del subalgoritmo para evitar efectos laterales.

Algoritmo A	Algoritmo B
<code>#include <iostream></code>	<code>#include <iostream></code>
<code>using namespace std;</code>	<code>using namespace std;</code>
<code>int numMayor(int , int);</code>	<code>void numMayor(int , int, int &);</code>
<code>main() {</code>	<code>main() {</code>
<code>int num1, num2, mayor;</code>	<code>int num1, num2, mayor, resultado;</code>
<code>mayor = 0;</code>	<code>resultado = 0;</code>
<code>cout << "Introduce dos números:</code>	<code>cout << "Introduce dos números: ";</code>

```
    ";
    cin >> num1 >> num2;                cin >> num1 >> num2;

    mayor = numMayor(num1,num2);          resultado = num1 + num2 ;
    cout << "El mayor es numMayor(num1, num2, mayor);
    "<<mayor<<endl;
}                                          cout << "Al sumar num1 y num2 da
                                          "<<resultado<<endl;

                                          }

int numMayor(int n1, int n2) {          void numMayor(int n1, int n2, int
                                          &resultadp) {

    int max;

                                          if (n1 > n2)

    if (n1 > n2)                          resultado = n1;

        max = n1;                        else

    else                                  resultado = n2;

        max = n2;                        }

    return (max) ;

}
```

6. Ejercicio resuelto 5. Repasa

- Realiza un programa en C que calcule el área de diferentes figuras geométricas. Concretamente de un rectángulo, un triángulo y un círculo. El programa deberá mostrar un menú con las diferentes opciones (más la opción de terminar) y posteriormente preguntar los datos necesarios para el cálculo del área.

El área de un rectángulo se calcula como:

`area_rectangulo = base * altura`

El área de un triángulo se calcula como:

`area_triangulo = (base * altura)/2`

El área de un círculo se calcula como:

`area_circulo = pi * radio^2`

Ejemplo de ejecución:

```
pc-rosana:pruebasC rosana$ ./ej
```

```
Seleccione la opción deseada
```

1. Área de un rectángulo
2. Área de un triángulo
3. Área de un círculo
4. Terminar

```
Elige una opción(1/2/3/4): 1
```

```
Área del rectángulo
```

```
Introduce la base: 50
```

```
Introduce la altura: 20
```

```
Resultado = 1000
```

```
Seleccione la opción deseada
```

1. Área de un rectángulo
2. Área de un triángulo
3. Área de un círculo
4. Terminar

```
Elige una opción(1/2/3/4): 2
```

```
Área del triángulo
```

```
Introduce la base: 40
```

```
Introduce la altura: 30
```

```
Resultado = 600
```

```
Seleccione la opción deseada
```

1. Área de un rectángulo
2. Área de un triángulo
3. Área de un círculo
4. Terminar

```
Elige una opción(1/2/3/4): 3
```

```
Área del círculo
```

```
Radio : 45
```

```
Resultado = 6075
```

```
Seleccione la opción deseada
```

1. Área de un rectángulo
2. Área de un triángulo
3. Área de un círculo
4. Terminar

```
Elige una opción(1/2/3/4): 6
```

```
Opción errónea
```

```
Seleccione la opción deseada
```

1. Área de un rectángulo
2. Área de un triángulo
3. Área de un círculo
4. Terminar

```
Elige una opción(1/2/3/4): 4
```

```
Hasta otra
```

```
Fin del programa
```

```
pc-rosana:pruebasC rosana$ █
```

6.1 Solución propuesta

```
#include <iostream>
#include <math.h>
using namespace std;

//Declaramos la constante que necesitamos
const int kpi=3.14159;

//Declaramos los módulos que vamos a usar después del main
void intr_datos(float &, float &);
int menu();

//Declaramos el main
main(){
    int opcion;
    float base, altura, radio, resultado;
    do{
        opcion = menu();
        switch(opcion){
            case 1: cout << "Área del rectángulo " << endl;
                    intr_datos(base, altura);
                    resultado = base * altura;
                    cout << "Resultado = " << resultado << endl;
                    break;
            case 2: cout << "Área del triángulo " << endl;
                    intr_datos(base, altura);
                    resultado = (base * altura) / 2;
                    cout << "Resultado = " << resultado << endl;
                    break;
            case 3: cout << "Área del círculo " << endl;
                    cout << "Radio : ";
                    cin >> radio;
                    resultado = kpi * pow(radio,2);
                    cout << "Resultado = " << resultado << endl;
                    break;
            case 4: cout << "Hasta otra " << endl;
```

```
        break;

        default:cout << "Opción errónea " << endl;
    }

    }while(opcion!=4);
    cout << "Fin del programa" << endl;
}

//Declaramos el menú del programa
int menu(){
    int op;

    cout << endl << "Seleccione la opción deseada " << endl;
    cout << "1. Área de un rectángulo" << endl;
    cout << "2. Área de un triángulo" << endl;
    cout << "3. Área de un círculo" << endl;
    cout << "4. Terminar" << endl;
    cout << "Elige una opción(1/2/3/4): ";
    cin >> op ;
    return(op);
}

//Este módulo introducirá los datos
void intr_datos(float &b, float &a){

    cout << "Introduce la base: ";
    cin >> b;
    cout << "Introduce la altura: ";
    cin >> a;
}
```

7. Ejercicio propuesto 1. Resuelve

- Realiza un programa en C que solicite un número entero y utilizando el módulo “capicua” determine si un número es capicúa o no y se lo indique al usuario. El módulo tomará el número como parámetro de entrada y como salida devolverá un 1 o un 0 para indicar si es o no capicúa. A partir de este dato el programa principal indicará al usuario si el número introducido es o no capicúa.

8. Ejercicio propuesto 2. Resuelve

- Realiza un programa en C que dibuje por pantalla un cuadrado. El programa debe tener además de la función main() los siguientes módulos:
- leePar(): lee un carácter c1, un carácter c2 que debe ser 'R' o 'V' y un entero n entre 4 y 20.
 - dibuja(): recibe como argumento los parámetros leídos en el módulo anterior y visualiza un cuadrado de lado n utilizando el carácter c1. Si el carácter c2 es 'R' el cuadrado debe estar relleno, en caso de que sea 'V' sólo debe dibujar el contorno. Este módulo debe devolver el número de caracteres dibujado y el tamaño del espacio interior (expresado como número de espacios en blanco del interior de la figura).

El main() debe mostrar en pantalla el número de caracteres dibujado y el tamaño del espacio interior.

Ejemplo de ejecución:

```
pc-rosana:pruebasC rosana$ ./ej
Introduce un caracter : *
¿Rellene o hueco? (R/H) R
Introduce el tamaño 4

****
****
****
****

El número de caracteres dibujado es 16
El tamaño del espacio interior es 0
pc-rosana:pruebasC rosana$ █
```

Otro ejemplo de ejecución:

```
pc-rosana:pruebasC rosana$ ./ej
Introduce un caracter : +
¿Rellene o hueco? (R/H) H
Introduce el tamaño 5

+++++
+  +
+  +
+  +
+++++

El número de caracteres dibujado es 16
El tamaño del espacio interior es 9
pc-rosana:pruebasC rosana$ █
```


9. Ejercicio propuesto 3. Resuelve

► Debes implementar un programa en C que permita jugar al juego Piedra, Papel, Tijera. El juego constará de varias rondas y ganará el que gane 2 rondas. El juego terminará cuando el usuario ya no quiera seguir jugando.

El programa deberá mostrar en cada ronda, las rondas que lleva ganadas cada jugador. Y al final de la partida, las partidas que lleva ganadas cada jugador.

Módulos:

Para implementar el juego, además de la función `main()`, el programa debe contener los siguientes módulos:

- **Módulo “apuestaUsuario”**. Este módulo debe pedir al usuario si quiere Piedra, Papel o Tijera. Se tendrá en cuenta si comprueba que la elección del usuario es correcta. El módulo debe mostrar la apuesta del usuario y **debe devolver la elección** del usuario (Piedra (p), Papel(a) o Tijera(t)).
- **Módulo “piedraPapelTijera”**. Este módulo mostrará y devolverá si la máquina ha elegido Piedra, Papel o Tijera. Debe calcularlo de manera aleatoria, para ello debe utilizar la función `rand()` (*consulta la explicación sobre la generación de números aleatorios en C que aparece al final de este ejercicio*)

la variable “valor”, almacenará un número entre 0 y 2. Por lo tanto, la función devolverá Piedra (p) si “valor” es 0, devolverá Papel (a) si “valor” es 1 o devolverá Tijera (t) si “valor” es 2.

- **Módulo “comprobar”**. Este módulo comprobará cuál de los dos jugadores ha ganado la ronda y mostrará los resultados. También comprobará si alguno de los dos jugadores ha ganado ya 2 rondas, y por tanto, la partida. Si, es así, mostrará quién ha ganado y las partidas que lleva ganadas cada jugador.

Ejemplo de ejecución básico:

```
pc-rosana:pruebasC rosana$ ./ej
*** PIEDRA (p) PAPEL (a) TIJERA (t) ***
¿Quieres jugar? (s/n) s
¿Piedra, papel, tijera? (p/a/t) p
Papel
Gana la máquina.
Rondas ganadas:
Máquina: 1
Usuario: 0
¿Piedra, papel, tijera? (p/a/t) t
Papel
Tú ganas.
Rondas ganadas:
Máquina: 1
Usuario: 1
¿Piedra, papel, tijera? (p/a/t) a
Tijera
Gana la máquina.
Rondas ganadas:
Máquina: 2
Usuario: 1

¡Oh! ¡Has perdido!
Partidas ganadas:
Máquina: 1
Usuario: 0

¿Quieres jugar? (s/n) n
Juego terminado

pc-rosana:pruebasC rosana$ █
```

Cómo generar números aleatorios en C

- Necesitaréis poner al principio del programa

```
#include <ctime>
#include <cstdlib>
```

- La primera instrucción dentro del `main()` debe ser `srand(time(0));` para inicializar el generador de números aleatorios. Si esto no se hiciera, cada vez que se ejecuta el programa se obtendría la misma secuencia de números. Esta instrucción solo se debe ejecutar una vez, no la metáis dentro de ningún bucle.
- Generar un entero al azar entre 0 y 2 es bastante sencillo, simplemente hay que usar `rand() % 3`. Como norma general, para obtener un entero entre 0 y n hay que usar `rand() % (n+1)`. Esto devuelve un valor que podéis asignar a una variable o devolver como resultado de una función, por ejemplo

```
int tirada;
tirada = rand() % 3;
```

- En general, para obtener un entero al azar entre a y b se usaría `rand() % (b-a+1) + a`. Esto parece un poco complicado, pero para los casos más habituales es

sencillo. Por ejemplo, para generar un número entre 1 y 20 la expresión anterior se convertiría en `rand() % (20-1+1) + 1`. O sea, simplemente `rand() % 20 + 1`.

10. Ejercicio propuesto 4. Resuelve

► **Implementar una versión simplificada del juego de cartas del BlackJack, también llamado 21. Se van repartiendo cartas de una en una y el objetivo del juego es llegar lo más cerca posible de 21 puntos sin pasarse. Habitualmente participa la banca y uno o más jugadores. La mecánica del juego será la siguiente:**

- Para simplificar, en nuestra versión del juego solo participa un jugador y la banca.
- La banca no va a ir obteniendo cartas como el jugador, simplemente se le asignará una puntuación aleatoria entre 1 y 21.
- Al jugador se le irán “repartiendo” cartas de 1 en 1. Es decir, se generará un valor al azar entre 1 y 12 y un valor al azar para el palo (bastos, copas, espadas u oros). Aclaración: esto no es muy realista porque se puede repetir la misma carta varias veces.
- Tras “repartirle” carta al jugador se le dirá qué carta ha salido, cuántos puntos lleva, y, si no se ha pasado de 21, se le preguntará si quiere otra carta (tendrá que introducir una 's' o una 'n'). Si no introduce ninguna de las dos, se le volverá a preguntar.
- Cuando el jugador diga que no quiere más cartas, se mostrará el resultado final. Ganará el jugador si tiene más puntos que la banca y no se ha pasado de 21 (fijaos en que la banca no puede pasarse tal y como funciona nuestra versión del juego)

El programa deberá tener al menos los siguientes módulos:

- **reparteCarta:** debe devolver el número y el palo de la carta “repartida” (o sea, generada al azar). Tendréis que generar un entero al azar entre 1 y 12 para el número y otro entre 1 y 4 para el palo. Si no sabes cómo generar un número al azar entre 1 y un valor máximo, consulta la explicación del ejercicio anterior.
 - Se interpretará 1 como bastos, 2 como copas, 3 como espadas y 4 como oros.
 - El módulo devolverá el palo como un carácter, no como un entero. O sea 'B' para bastos, 'C' para copas, etc.
 - No es misión de este módulo imprimir en pantalla qué carta ha salido, esto es trabajo del main o de otro módulo.
- **calculaTotal:** recibe como parámetro los puntos actuales del jugador y el número de la carta que ha salido. Devuelve el nuevo total, sumando los puntos actuales más el valor de la nueva carta teniendo en cuenta que:
 - El as vale 11 puntos.

- Las figuras (cartas mayores que 9) valen 10.
- El resto de cartas tienen su valor habitual.
- No es misión de este módulo comprobar si nos hemos pasado o no de 21, simplemente debe calcular el resultado (sea el que sea).

Ejemplos de ejecución:

```
pc-rosana:pruebasC rosana$ ./ej  
  
Te ha salido un 9 de E  
Por ahora tienes 9. ?Quieres carta? (s/n) s  
Te ha salido un 11 de C  
Por ahora tienes 19. ?Quieres carta? (s/n) n  
La banca ten?a 13  
Has ganado!!!  
  
pc-rosana:pruebasC rosana$ █
```

```
pc-rosana:pruebasC rosana$ ./ej  
  
Te ha salido un 3 de B  
Por ahora tienes 3. ?Quieres carta? (s/n) s  
Te ha salido un 9 de E  
Por ahora tienes 12. ?Quieres carta? (s/n) s  
Te ha salido un 10 de C  
Por ahora tienes 22. Te has pasado  
La banca ten?a 17  
Ha ganado la banca  
  
pc-rosana:pruebasC rosana$ █
```

11. Ejercicio propuesto 5. Resuelve

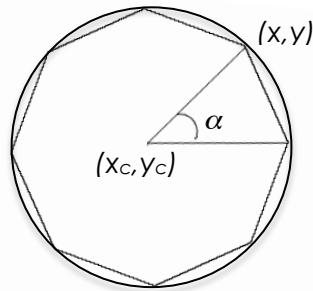
- Recupera el ejercicio propuesto 5 del tema 3 (dibujaba un cuadrado o un rectángulo). Ayudándote de dicho ejercicio escribe un programa en C que dibuje un polígono regular de n lados inscrito en una circunferencia, empleando la librería gráfica gfx. El programa solicitará las coordenadas (x_c, y_c) del centro del polígono y su radio r , así como el número de lados a dibujar. Una vez comprobado que no se excede de la ventana gráfica, el programa dibujará el polígono centrado en la posición (x_c, y_c) .

Para hallar los vértices del polígono puedes utilizar las siguientes ecuaciones:

$$x = x_c + r \cdot \cos(\alpha)$$

$$y = y_c + r \cdot \sin(\alpha)$$

donde α es el ángulo correspondiente al vértice, como se observa en la figura.



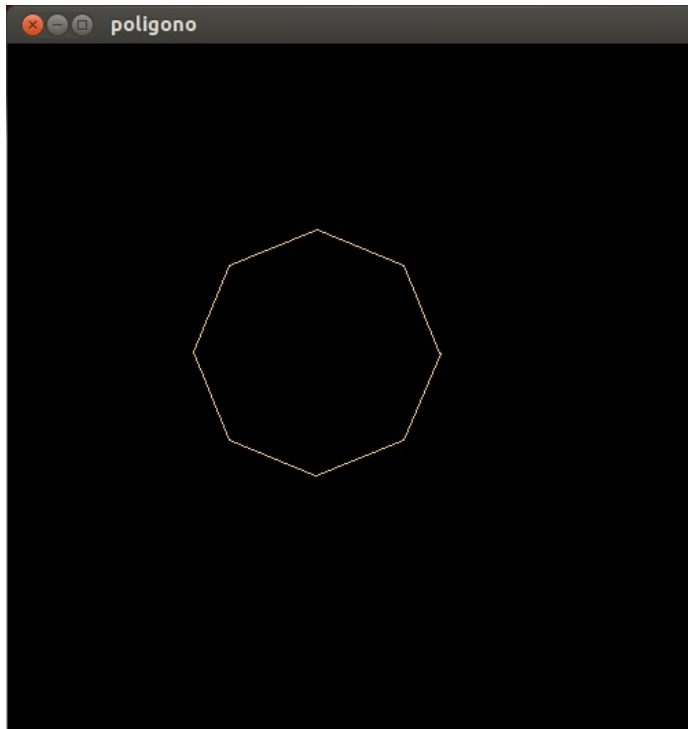
Puedes hacer uso de la rutina `gfx_line()` para dibujar los lados del polígono.

El programa debe tener como mínimo los siguientes módulos:

- **validaCoordenadas:** módulo que comprueba si algún vértice del polígono se sale de la zona de trazado. El programa debe dibujar el polígono sólo si todos los vértices se encuentran en la zona de trazado
- **validaLados:** función que se encarga de pedir el número de lados del polígono y validar que se encuentra entre 4 y 15. Esta función debe obligar al usuario a introducir un número correcto.

Por ejemplo:

```
pc-rosana:pruebasC rosana$ ./ej
Introduce el punto central 200 250
Introduce el tamaño del radio 90
Introduce el número de lados 8
```



12. Errores de programación comunes

- **Utilizar punto y coma en la definición de una función.** Cuando se define una función, su cabecera no termina con punto y coma. **El punto y coma se utiliza en las declaraciones**, es decir, en los prototipos de las funciones. Lo mismo se puede decir si se trata de un procedimiento (función tipo void en C).
- **Llamar a una función con un número diferente de parámetros.** Hay que asegurarse de que se pasa el número exacto de parámetros cuando se invoca a una función o a un procedimiento.
- **Devolver un valor que no coincide con el tipo de la función.** Cuando se devuelve un valor utilizando return, hay que asegurarse de que el resultado de la expresión situada dentro de return coincide con el tipo de la función.
- **No utilizar return en una función que devuelve un valor.** Cuando una función devuelve un valor, hay que utilizar return para devolverlo.
- **Pasar un parámetro real con distinto tipo al parámetro formal.** Cuando llame a una función o a un procedimiento, el tipo de los parámetros reales debe coincidir con el tipo de los correspondientes parámetros formales.

- **Olvidar los paréntesis en una función que no acepta parámetros.** Aunque una función no acepte parámetros, cuando se realiza la llamada a la misma deben utilizarse paréntesis, sin nada entre ellos. Si no se colocan los paréntesis, se obtiene un error de compilación.
- **Declarar un prototipo que no coincide con la definición de la función.** Cuando se define una función, hay que asegurarse de que tanto la definición como su declaración tienen el mismo prototipo, es decir, devuelven el mismo tipo y aceptan el mismo número y tipo de parámetros.

ANEXO. Ejemplos de código en Java

Código del ejercicio resuelto 1

Realiza un programa en C que permita convertir grados Celsius a Fahrenheit y viceversa. El programa debe mostrar un menú para poder seleccionar qué opción se desea (más la opción terminar). Y preguntar la temperatura que desea convertir.

```
import java.util.Scanner;

class ejRes1{

    public static void mostrar_menu() {

        System.out.println("Seleccione qué tipo de conversión desea realizar:");

        System.out.println("1. Convertir grados Celsius a Fahrenheit.");

        System.out.println("2. Convertir grados Fahrenheit a Celsius.");

        System.out.println("3. Terminar el programa.");

    }

    public static float Celsius_to_Fahrenheit(float cel) {

        float fah;

        fah = (float)(cel * 1.8) + 32;

        return(fah);

    }

    public static float Fahrenheit_to_Celsius(float fah) {
```

```
float cel;

cel = (fah-32)/(float)1.8;

return (cel);
}

public static void main(String args[]) {

    int opcion;

    float celsius, fahr;

    Scanner sc = new Scanner(System.in);

    do{

        mostrar_menu();

        System.out.println("Opción: ");

        opcion = sc.nextInt();

        System.out.println();

        switch(opcion){

            case 1: System.out.println("Introduzca la temperatura en grados Celsius: ");

                    celsius = sc.nextFloat();

                    fahr = Celsius_to_Fahrenheit(celsius);

                    System.out.println("La temperatura en grados Fahrenheit es: "+fahr);

                    break;

            case 2: System.out.println("Introduzca la temperatura en grados Fahrenheit: ");

                    fahr = sc.nextFloat();

                    celsius = Fahrenheit_to_Celsius(fahr);

                    System.out.println("La temperatura en grados Celsius es: "+celsius);

                    break;

            case 3: System.out.println("Ha seleccionado la opción Salir.");
```



```

        break;

        default: System.out.println("Debe introducir una opción de 1 a 3");
    }

    System.out.println();

    System.out.println("Pulse C para continuar");

    sc.next();

}while(opcion!=3);
}
}

```

Código del ejercicio resuelto 2

Calcular la representación en base 10 de un número a partir de su representación en base 2. Por ejemplo, $\text{bin2dec}(1010) = 10$. Se pide:

1. Definir una función que calcule ese proceso.
2. Utilizar dicha función correctamente en un programa que interaccione con el usuario.
3. Definir un conjunto de pruebas para el problema.

```

import java.util.Scanner;

class ejRes2{

    public static int BinarioADecimal(int n) {
        int n_decimal=0, coef=1;

        while (n!=0) {
            n_decimal += coef*(n%10);
            coef *= 2;
            n = n / 10;
        }

        return (n_decimal);
    }
}

```

```
}
```

```
public static void main(String args[]) {
```

```
    // Declaración de variables
```

```
    int num2, num10;
```

```
    Scanner sc = new Scanner(System.in);
```

```
    // Entrada de datos
```

```
    System.out.println("Introduzca un numero en base 2: ");
```

```
    num2 = sc.nextInt();
```

```
    System.out.println();
```

```
    // Llamada a la función
```

```
    num10 = BinarioADecimal(num2);
```

```
    // Mostrar Resultados
```

```
    System.out.println("El numero "+num2+" en base 2 equivale en base 10  
a "+num10);
```

```
}
```

```
}
```

Código del ejercicio resuelto 3

Realiza un programa que muestre un menú en el que a partir de un número entero n se elegirá una de las siguientes operaciones:

Opción 1: Calcular de cuántas cifras se compone. (No contar los ceros a la izquierda)

Opción 2: Mostrar la cifra i-ésima de dicho número. La posición i debe pedirse al usuario.

Opción 3: Salir.

```
import java.util.Scanner;
```

```
class ejRes3{
```

```
static final int OPCION_SALIR=4;
```

```
//Función que calcula la cantidad de cifras
```

```
public static int CantidadCifras(int m) {
```

```
    int contador=1;
```

```
    while(m >=10) {
```

```
        m=m/10;
```

```
        contador ++;
```

```
    }
```

```
    return(contador);
```

```
}
```

```
//Función que calcula la cifra i-ésima del número
```

```
public static int Cifra_i(int m,int i) {
```

```
    int contador=1,cifra=0,devolver;
```

```
    cifra=m%10;
```

```
    while(m >=10 && contador<i) {
```

```
        m=m/10;
```

```
        cifra=m%10;
```

```
        contador ++;
```

```
    }
```

```
    if (contador==i)
```

```
        devolver=cifra;
```

```
    else
```

```
        devolver=-1;
```

```
    return(devolver);
```

```
}
```

```
//Función que muestra el menú
```

```
public static void menu() {
```

```
    System.out.println("1. Calcular el número de cifras de un número");
```

```
    System.out.println("2. Mostrar la cifra i-ésima de un número");
```

```
    System.out.println("3. Mostrar el reverso de un número");
```

```
    System.out.println("4. SALIR");
```

```
}
```

```
public static void main(String args[]) {  
    // Declaración de variables  
    Scanner sc = new Scanner(System.in);  
    int opcion, n, resultado, i;  
  
    do {  
        menu();  
        System.out.println("Introduzca la opción: ");  
        opcion = sc.nextInt();  
  
        switch(opcion) {  
            case 1: System.out.println("Introduzca un número: ");  
                    n = sc.nextInt();  
                    resultado=CantidadCifras(n);  
                    System.out.println("El número de cifras de "+n+" es  
"+resultado);  
                    break;  
            case 2: System.out.println("Introduzca un número: ");  
                    n = sc.nextInt();  
                    System.out.println("Introduce la i: ");  
                    i = sc.nextInt();  
                    resultado=Cifra_i(n,i);  
                    System.out.println("La cifra "+i+" de "+n+" es  
"+resultado);  
                    break;  
            case OPCION_SALIR:  
                System.out.println("FIN DEL PROGRAMA");  
        }  
  
        System.out.println("Pulsa C para continuar");  
        sc.next();  
    } while(opcion != OPCION_SALIR);  
}
```
