

# Programación

## Ejercicios Tema 5 Funciones

**Autores:**

**M. Paz Sesmero Lorente**  
**Paula de Toledo Heras**  
**Fco. Javier Ordoñez Morales**  
**Juan Gómez Romero**  
**Jose A. Iglesias Martínez**  
**Jose Luis Mira**



# SOLUCIONES

1. Escriba una función que devuelva el resultado de elevar un número (declarado como entero) a cualquier potencia (valor entero). El resultado de este cálculo debe ser de tipo long.

```
#include <stdio.h>

long potencia(int x, int n);

int main(void)
{
    int b;
    int e;

    /* Elevar a cualquier potencia */
    printf("Introduce base: ");
    scanf("%ld", &b);
    printf("Introduce exponente: ");
    scanf("%i", &e);
    printf("%d elevado a %i es %ld \n", b, e, potencia(b, e));

    system("PAUSE");
    return 0;
}

long potencia(int x, int n) {
    /* Funcion potencia:
    * Eleva el valor x al exponente n
    * Argumentos: x Base (int)
    *               n Exponente (int)
    * Valor de retorno: x^n (long)
    */

    long ret = x;
    int i;

    for(i=1; i<n; i++)
        ret = ret * x;

    return ret;
}
```

2. Escriba una función que reciba dos números reales positivos como parámetros y devuelva el máximo de los dos valores.

```
#include <stdio.h>

float max(float x, float y);

int main(void)
{
    float a, b;

    printf("Introduzca a: ");
    scanf("%f", &a);
```

```

printf("Introduzca b: ");
scanf("%f", &b);

printf("Mayor: %f\n", max(a, b));

system ("PAUSE");
return 0;
}

float max(float x, float y) {
/* Función max: Devuelve el mayor de los valores x, y
pasados como parámetro */

float max;
if(x >= y)
    max = x;
else
    max = y;
return max;
}

```

### 3. Escriba un programa en C que muestre el siguiente menú:

#### CALCULO DE AREAS

- ```

=====
1 - Calcular el área de un triángulo
2 - Calcular el área de un trapezio
3 - Calcular el área de un rectángulo

```

Y lea la opción introducida por el usuario. En función de la opción el programa deberá pedir los datos necesarios para calcular el área, realizar el cálculo y mostrar el resultado por pantalla.

Utilice una función para mostrar el menú, y otras tres para calcular el área de los distintos polígonos (triángulo, trapezio rectángulo). El valor de retorno de cada una de estas tres últimas funciones debe ser el área calculada.

```

#include <stdio.h>
int mostrarMenu(void);
float CalcularAreaTriangulo(void);
float CalcularAreaTrapezio(void);
float CalcularAreaRectangulo(void);

int main(void){
int opcion; //Para almacenar la opción elegida
float area; //Almacena el área calculadoa

opcion=mostrarMenu();

switch(opcion){
case 1:
area=CalcularAreaTriangulo();
printf("El area del triangulo es %4.2f\n",area);
break;

case 2:
area=CalcularAreaTrapezio();
printf("El area del trapezioes %4.2f\n",area);

```

```

        break;

    case 3:
        area=CalcularAreaRectangulo();
        printf("El area del cuadrado es %4.0f\n",area);
        break;

    default:
        printf("Fin del programa.");
    }
    system ("PAUSE");
    return 0;
}

int mostrarMenu(void) {
    /* Función que muestra por pantalla un menú y devuelve la
    opción seleccionada*/

    int opcion;

    printf("CALCULO DE AREAS  \n");
    printf("===== \n\n\n");
    printf("1 - Calcular el área de un triangulo \n");
    printf("2 - Calcular el área de un trapecio \n");
    printf("3 - Calcular el área de un rectangulo \n\n\n");

    printf("Opcion: ");
    scanf("%d",&opcion);
    return (opcion);
}

float CalcularAreaTriangulo(void) {
    /* Función que calcula y devuelve el área de un triángulo.*
    /* base y altura deben ser valores enteros */

    float area;
    int base, altura;

    printf("AREA DEL TRIÁNGULO  \n");
    printf("Base: ");
    scanf("%d",&base);
    printf("Altura: ");
    scanf("%d",&altura);
    //Para conseguir un valor real se hace una conversión explícita
    area=(float)base*altura/2;
    return (area);
}

float CalcularAreaTrapecio(void) {
    /* Función que calcula y devuelve el área de un trapecio */
    /* base_mayor, base_menor y altura deben ser valores enteros */

    float area;
    int base_mayor, base_menor, altura;
    printf("AREA DEL TRAPECIO  \n");
    printf("Base mayor: ");
    scanf("%d",&base_mayor);
    printf("Base menor: ");
    scanf("%d",&base_menor);

```

```

printf("Altura del trapecio: ");
scanf("%d",&altura);
//Para conseguir un valor real se utiliza 2.0
area=((base_mayor+base_menor)/2.0)*altura;
return (area);
}

float CalcularAreaRectangulo(void) {
    /* Función que calcula y devuelve el área de un trapecio */
    /* base y altura deben ser valores enteros */

    float area;
    int base, altura;
    printf("AREA DE UN RECTANGULO   \n");
    printf("Base: ");
    scanf("%d",&base);
    printf("Altura: ");
    scanf("%d",&altura);
    //En ese caso y dado que base y altura son reales el resultado
    //siempre será real
    area=(float)base*altura;
    return (area);
}

```

#### 4. Escriba una función que intercambie los valores de dos variables de tipo entero.

```

#include <stdio.h>

void swap(int *x, int *y);

int main(void)
{
    int a, b;

    printf("Introduzca a: ");
    scanf("%i", &a);
    printf("Introduzca b: ");
    scanf("%i", &b);

    printf("\nAntes de intercambio\n");
    printf("a: %i\n", a);
    printf("b: %i", b);

    swap(&a, &b);

    printf("\nDespues de intercambio\n");
    printf("a: %i\n", a);
    printf("b: %i\n", b);

    return 0;
}

void swap(int *x, int *y) {
    /* Función que intercambia el valor de dos variables. Para
    conseguir que los cambios sean visibles en el programa principal
    los parámetros se deben pasar por referencia*/
    int temp;
    temp = *x;
    *x = *y;
    *y = temp;
}

```

5. La nota del examen de una asignatura corresponde en un 20% al primer parcial, un 20% al segundo parcial y un 60% el examen final. La nota final es la mejor entre la nota del examen y la nota obtenida con la evaluación continua (teniendo en cuenta los dos parciales).

Escriba un programa que pida estas tres notas y calcule la nota final de un alumno. A continuación, el programa debe preguntar al usuario si quiere calcular una nueva nota o finalizar (pulsando la tecla 's'). Utilice una función para calcular la nota y otra para obtener el máximo (que puede ser la desarrollada en el ejercicio 2).

```
#include <stdio.h>

float calcularMax(float x, float y);
float calcularNota(float p1, float p2, float e);

int main(void)
{
    char c;
    do {
        float parcial1, parcial2, examen, final;

        /* Leer valores */
        printf("Introduzca nota parcial 1 [0, 10]: ");
        scanf("%f", &parcial1);
        printf("Introduzca nota parcial 2 [0, 10]: ");
        scanf("%f", &parcial2);
        printf("Introduzca nota examen final [0, 10]: ");
        scanf("%f", &examen);

        /* Calcular nota final*/
        final = calcularNota (parcial1, parcial2, examen);

        /* Imprimir resultados */
        printf("Nota final: %f", final);

        /* Continuar? */
        printf("\n\nSalir? (s/n): ");
        /* Para evitar conflictos con el salto de linea (\n) y
        evitar un bucle infinito se recurre al siguiente "truco" */
        do{
            scanf("%c", &c);
        } while (c!='\n');
        printf("\n");

    } while(c != 's');

    return 0;
}

float calcularNota (float p1, float p2, float notaExamen) {
    /* Función calcularNota: Calcula la nota final a partir de:
    - primer parcial 20% (p1)
    - segundo parcial 20% (p2)
    - examen final 60% (notaExamen)
    Si la nota final con la evaluación continua es menor a la del
    examen, la nota final es la del examen
    */
}
```

```

float notaEC, final;
notaEC    = p1*0.2+p2*0.2+notaExamen*0.6;
final = calcularMax(notaEC, notaExamen);
return final;
}

float calcularMax(float x, float y) {
/* Funcion calcularMax: Devuelve el mayor de los valores x, y
pasados como parámetro */

float max;
if(x >= y)
    max = x;
else
    max = y;
return max;
}

```

6. Realizar un programa en C que verifique si existe o no concordancia entre los datos de un NIF (DNI y letra) introducido por teclado. Para ello:

- a. Tras solicitar el DNI se debe comprobar que el valor introducido es positivo y tiene a lo sumo 8 dígitos. En caso de no cumplir estas condiciones se ha de indicar que el DNI introducido es incorrecto y se pedirá un nuevo valor.

En la fase de verificación se debe utilizar una función que reciba como parámetro el número del DNI y, tras hacer las comprobaciones pertinentes, devuelva el valor VERDADERO (1) o el valor FALSO (0).

- b. Una vez introducido un DNI correcto, se solicitará una letra y el programa debe determinar si DNI y letra son concordantes.

Para ello el alumno debe:

- b.1. Implementar una función que dado el número del DNI, devuelva la letra que le corresponde. Para calcular dicha letra hay que tener en cuenta que la misma es función del resto obtenido al dividir el número del DNI entre 23. La relación entre ambos parámetros es:

| Resto | Letra | Resto | Letra | Resto | Letra |
|-------|-------|-------|-------|-------|-------|
| 0     | T     | 8     | P     | 16    | Q     |
| 1     | R     | 9     | D     | 17    | V     |
| 2     | W     | 10    | X     | 18    | H     |
| 3     | A     | 11    | B     | 19    | L     |
| 4     | G     | 12    | N     | 20    | C     |
| 5     | M     | 13    | J     | 21    | K     |
| 6     | Y     | 14    | Z     | 22    | E     |
| 7     | F     | 15    | S     |       |       |

- b.2 Implementar una función que reciba, entre otros parámetros, la letra del DNI introducida por el usuario y devuelva si dicha letra corresponde o no al DNI introducido.

- c. Al final, el programa debe indicar al usuario si el NIF introducido es o no correcto.

**Nota:** En la resolución de este problema se sugiere hacer uso de las siguientes funciones contenidas en la biblioteca estándar ctype.h.

`int isalpha(int c);` Devuelve verdadero si c es una letra  
`int toupper (int c);` Devuelve el carácter en mayúscula correspondiente al carácter pasado por parámetro.

```
#include <stdio.h>
#include <ctype.h>
int validarDni(int n);
int verificarLetraNif(int dni, char letra);
char calcularLetraNif(int n);

int main(void)
{
    int dni;
    int ok;
    char letra;
    do{
        printf ("Introduzca su DNI: ");
        scanf("%d", &dni);
        ok=validarDni(dni);
        if (ok==0)
            printf("El numero de DNI introducido no es correcto\n");
    }while (ok==0);

    do{
        printf("Introduzca la letra del NIF: ");
        do{
            scanf("%c", &letra);
        } while (letra=='\n');
        ok=isalpha(letra);
        if (ok==0)
            printf("El caracter introducido no es letra. Por favor,
introduzca una letra\n");
        else
            letra=toupper(letra);
    }while (ok==0);

    ok=verificarLetraNif(dni, letra);
    if (ok==1)
        printf("Su NIF es %d%c y lo ha introducido correctamente\n",
dni, letra);
    else
        printf("La letra del NIF es incorrecta\n");

    system ("PAUSE");
    return (0);
}

int validarDni(int n){
    int valido=1;
    if ((n<0) || (n>99999999))
        valido=0;
    return (valido);
}
```

```

}

char calcularLetraNif(int n){
    int resto;
    char letra;
    resto=n%23;
    switch (resto){
        case 0: letra='T'; break;
        case 1: letra='R'; break;
        case 2: letra='W'; break;
        case 3: letra='A'; break;
        case 4: letra='G'; break;
        case 5: letra='M'; break;
        case 6: letra='Y'; break;
        case 7: letra='F'; break;
        case 8: letra='P'; break;
        case 9: letra='D'; break;
        case 10: letra='X'; break;
        case 11: letra='B'; break;
        case 12: letra='N'; break;
        case 13: letra='J'; break;
        case 14: letra='Z'; break;
        case 15: letra='S'; break;
        case 16: letra='Q'; break;
        case 17: letra='V'; break;
        case 18: letra='H'; break;
        case 19: letra='L'; break;
        case 20: letra='C'; break;
        case 21: letra='K'; break;
        case 22: letra='E'; break;
    }
    return (letra);
}

int verificarLetraNif(int dni, char letra){
    char letraCorrecta;
    int ok;

    letraCorrecta=calcularLetraNif(dni);
    if (letra==letraCorrecta)
        ok=1;
    else
        ok=0;
    return (ok);
}

```

7. Realizar un programa que pida al usuario información de un día/mes/año posterior al 1978 y a continuación le muestre al usuario el día de la semana con que se corresponde esa fecha. Para realizar este cálculo la única información disponible es que el 1 de Enero de 1978 fue domingo.

Para realizar el ejercicio conviene tener en cuenta que el día de la semana se puede obtener a partir del número de días transcurridos entre la fecha introducida por el usuario y la fecha de referencia (1 de enero de 1978). Una vez conocido este valor, el día de la semana vendrá dado por:

$$\text{dia\_semana} = \text{n\_dias\_transcurridos} \% 7$$

En la resolución de este ejercicio se deben usar las siguientes funciones:



- a) `void leerFecha(int *dia, int *mes, int *year);`  
 Objetivo: Solicita los datos relativos a una fecha hasta que éstos se correspondan con una fecha válida.
- b) `int validarFecha (int dia, int mes, int year);`  
 Valor de retorno: 1 si la fecha introducida es válida (posterior al 01/01/1978;  $0 < \text{mes} < 13$ ;  $0 < \text{d} \leq \text{días\_mes}$ ) y 0 en caso contrario.
- c) `int calcularDias(int dia, int mes, int year);`  
 Valor de retorno: Número de días que hay entre una determinada fecha y el 1 de Enero de 1978.  
 Para realizar correctamente este cálculo se requiere identificar los años bisiestos
- d) `int anioBisiesto(int year);`  
 Valor de retorno: 1 si el año es bisiesto y 0 en caso contrario.

**Nota: Un año es bisiesto si es múltiplo de 4 excepto aquellos divisibles por 100 a no ser que lo sean por 400. Así, por ejemplo, los años 1500, 1700, 1800, 1900 y 2100 son años no bisiestos mientras que los años 1600, 1200 ó 2400 sí son años bisiestos.**

```
#include <stdio.h>
int anioBisiesto(int year);
int validarFecha (int d, int m, int a);
int calcularDias(int d, int m, int a);
void leerFecha(int *d, int *m, int *a);

int main(void)
{
    int dia, mes, anio;
    int totalDias;
    int resto;
    printf("Por favor, introduzca una fecha posterior al 1 de Enero
de 1978\n");
    leerFecha(&dia, &mes, &anio);
    totalDias=calcularDias(dia, mes, anio);
    resto=totalDias%7;
    printf("El dia %d/%d/%d será ", dia, mes, anio);
    switch (resto){
        case 0: printf("domingo\n");break;
        case 1: printf("lunes\n");break;
        case 2: printf("martes\n");break;
        case 3: printf("miercoles\n");break;
        case 4: printf("jueves\n");break;
        case 5: printf("viernes\n");break;
        case 6: printf("sabado\n");
    }

    system ("PAUSE");
    return (0);
}

int anioBisiesto(int year){
    /* Dado un año determina si éste es bisiesto o no
    Para ello hay que saber que los años bisiestos son aquellos
    que son divisibles por 4 pero no por 100 o
    divisibles por 4 y por 400 */
    int bisiesto=0;
```

```

    if ( ((year%4) == 0) && (((year%100)!=0) || ((year%400)==0)) ) {
        bisiesto=1; //Si el año no es divisible por 4 no es bisiesto
    }
    return bisiesto;
}

int validarFecha (int d, int m, int a){
    /* Comprueba que los datos introducidos corresponden a una
    fecha válida:
        año > 1978
        1>mes<=12
        1>=dia<=dias mes
    Para el último cálculo es necesario conocer los días que
    tiene el mes introducido (31, 30, 29 ó 28)*/
    int diasMes;
    int bisiesto;
    int correcta;

    correcta=1;
    /*Se comprueba la validez del año*/
    if (a<1978){
        printf("La fecha introducida debe ser posterior al 1 de
Enero de 1978\n");
        correcta=0;
    }
    /*Si el año es válido se comprueba la validez del mes*/
    else
        if ((m<1) || (m>12)){
            printf("El mes introducido no es válido\n");
            correcta=0;
        }
        /*Si el mes es válido se determina el núm. de días del mes*/
        else{
            switch (m){
                case 1: case 3: case 5: case 7: case 8: case 10: case
12:
                    diasMes=31;
                    break;
                case 4: case 6: case 9: case 11:
                    diasMes=30;
                    break;
                case 2:
                    /*Para determinar si febrero tiene 28 o 29 días se
                    analiza si el año es bisiesto*/
                    bisiesto=anioBisiesto(a);
                    if (bisiesto==1)
                        diasMes=29;
                    else
                        diasMes=28;
                } //switch

                if ((d<0) || (d>diasMes)){
                    correcta=0;
                    printf("El día introducido no es valido\n");
                } //if ((d<0) || (d>diasMes)
            } //else
        }
    return (correcta);
}

void leerFecha(int *d, int *m, int *a){

```

```

    /*Lee una fecha y devuelve los valores leídos
    *d: día
    *m: mes
    *a: año
    al programa principal
    Para conseguir este objetivo los parámetros se pasan por
    referencia*/
    int valida;
    do{
        printf ("Día: "); scanf("%d", d);
        /*Puesto que en este caso d, m y a son punteros se omite
        El operador dirección (&) en la llamada a scanf*/
        printf ("Mes: "); scanf("%d", m);
        printf ("Year: "); scanf("%d", a);
        valida=validarFecha(*d, *m, *a);
    }while (valida==0);
}

int calcularDias(int d, int m, int a){
    /*Determina el número de días transcurridos entre la fecha
    introduciday el 1 de enero de 1978*/
    int i;
    int bisiesto;
    int dias=0;

    /*Por cada año transcurrido hasta llegar al de la fecha
    introducida se suman 365 ó 366 días según corresponda*/
    for (i=1978; i<a; i++){
        bisiesto=anioBisiesto(i);
        if (bisiesto==1)
            dias=dias+366;
        else
            dias=dias+365;
    }
    /*Por cada mes transcurrido hasta llegar al de la fecha
    introducido se suman 31, 30, 29 ó 28 días según corresponda*/
    for (i=1; i<m; i++)
    {
        switch (i){
            case 1: case 3: case 5: case 7: case 8: case 10: case 12:
                dias=dias+31;
                break;
            case 4: case 6: case 9: case 11:
                dias=dias+30;
                break;
            case 2:{
                bisiesto=anioBisiesto(a);
                if (bisiesto==1)
                    dias=dias+29;
                else
                    dias=dias+28;
            }
        }
    }
    return (dias);
}

```

8. Escribir un programa que solicite al usuario un número natural y el número de decimales que se quiere contemplar al calcular su raíz. Una vez introducidos estos datos, se debe mostrar por pantalla el resultado de:

- a) Realizar la raíz cuadrada usando el método de las aproximaciones sucesivas.
- b) Calcular el factorial de ese número.
- c) Indicar si el número introducido es primo o no. Un número primo es aquel que sólo es divisible por sí mismo y por el número 1. Por convenio, el número 1 no se considera primo

```
#include <stdio.h>

double calcularRaiz(int n, int decimales);
long calcularFactorial(int n);
int comprobarPrimo(int n);

int main(void)
{
    int n;
    int decimales;
    double raiz;
    long factorial;
    int primo;
    //Solicitamos un número hasta que éste sea positivo
    do{
        printf("Introduzca el número del que quiere calcular la
raiz\n");
        scanf ("%d", &n);
        if (n<0)
            printf("ERROR: No se puede calcular la raiz de un
número negativo\n");
        }while (n<0);

        //Solicitamos el número de decimales que tendrá la
aproximación. Este número debe ser positivo:
        do{
            printf ("Introduza el número de decimales que tendrá la
aproximación\n");
            scanf ("%d", &decimales);
        }while (decimales<0);

        raiz=calcularRaiz(n, decimales);
        factorial=calcularFactorial(n);
        primo=comprobarPrimo(n);

        //Mostramos resultados
        printf ("La raiz cuadrada de %d es %lf\n", n, raiz);
        printf ("Su Factorial es %ld\n", factorial);
        if (primo==1)
            printf ("Ademas, el numero es primo\n");
        else
            printf ("Ademas, el numero NO es primo\n");

        return (0);
    }
}
```

```

double calcularRaiz(int n, int decimales){
    /*PARAMETROS
    int n;           //Numero del que queremos calcular la raíz
    int decimales;  //Numero de decimales a los que vamos a
                    aproximar*/

    //Variables locales
    double raiz;    /*Numero con el que probaremos si es la
                    raíz. Valor de Retorno*/

    float incremento;
    int dec;        //Numero de decimales a los que vamos a aproximar
                    //y número de decimales con los que estamos
                    //realizando el cálculo

    //Inicializamos raiz al menor entero posible, es decir, a 0:
    raiz=0;
    incremento=1.0;
    /*Inicialmente buscamos el entero más próximo a la raíz;
    Es decir, probaremos con números que difieren entre sí en
    1 unidad. Por tanto, el número de decimales para la
    Primera aproximación es 0 */
    dec=0;
    while ((n>(raiz*raiz) )&&(decimales>=dec)){
        while(raiz*raiz<n) {
            raiz=raiz+incremento;
        }
        //Al salir del bucle, el valor almacenado en raiz será mayor
        //o igual que el número buscado.
        //Si es mayor (nos hemos pasado), actualizamos el valor de
        //raiz
        if (raiz*raiz!=n){
            raiz=raiz-incremento;
            //Ajustamos a un nuevo decimal
            incremento=incremento/10.0;
            dec++;
        }
    }//while

    return (raiz);
}

long calcularFactorial(int n){
    long factorial;
    factorial=1;
    int i;

    //Dado que el factorial tanto de 0 como de 1 es 1, el caso
    //genérico se da para n>=2. En este caso:
    //n!=1*2*3*....*n

    for (i=2; i<=n; i++)
        factorial=factorial*i;

    return (factorial);
}

comprobarPrimo(int n){
    int primo;
    int i;

```

```
int divisor;

if (n==1)
    primo=0;
else
    primo=1; //Se admite que el número es primo
divisor=2;
//Se intenta localizar algún divisor del número dado
while ((primo==1)&&(divisor<n)){
    if (n%divisor==0)
        primo=0;
    else
        divisor=divisor+1; //Si el número sigue siendo
                           //primo probamos con
                           //otro divisor
}
return (primo);
}
```