

# PROGRAMACIÓN I

Comenzado el	sábado, 15 de diciembre de 2018, 15:42
Estado	Finalizado
Finalizado en	sábado, 15 de diciembre de 2018, 15:54
Tiempo empleado	11 minutos 43 segundos
Puntos	17,00/20,00
Calificación	8,50 de 10,00 (85%)

### Pregunta 1

Correcta

Puntúa 1,00  
sobre 1,00

¿Con qué formato deberíamos llamar a la función fopen para que cada vez que se abra el fichero y se escriba en él, se añada una nueva línea con el texto "hola", como en el ejemplo siguiente?

```
int main() {
```

```
FILE *f;
```

```
int i;
```

```
for (i=0;i<10;i++)
```

```
{
```

```
    f=fopen("fichero.txt", OPCION);
```

```
    fprintf(f, "hola\n");
```

```
    fclose(f);
```

```
}
```

```
return 0;
```

```
}
```

Seleccione una:

- ☒ "a" ✓
- ☐ "w"
- ☐ "w+"

#### EXPLICACIÓN:

El formato "a" abre el fichero de forma que la información que se escriba se añada a la ya existente, mientras que los formatos con "w" empiezan a escribir desde el principio del fichero, borrando todo lo que contuviese anteriormente.

La respuesta correcta es: "a"

## Pregunta 2

Correcta

Puntúa 1,00  
sobre 1,00

En un programa en C se pretende leer una fecha separada por blancos, de acuerdo con este ejemplo de ejecución:

```
Por favor, introduce una fecha: 11 3 1972
La fecha leida es 11/3/1972
```

Suponiendo que para almacenar la fecha se dispone de una variable llamada fecha de tipo Fecha definido así:

```
typedef struct {
    int dia;
    int mes;
    int anno;
} Fecha;
```

¿Cuál de las siguientes sentencias permitiría leer la fecha introducida de forma correcta

Seleccione una:

☐

```
scanf("%d %d %d", fecha.&dia, fecha.&mes, fecha.&anno);
```

☒

```
scanf("%d %d %d", &fecha.dia, &fecha.mes, &fecha.anno);
```

☐

```
scanf("%d %d %d", &fecha.&dia, &fecha.&mes, &fecha.&anno);
```

Respuesta correcta

La precedencia del operador "." es mayor que la del operador "&".

La respuesta correcta es:

```
scanf("%d %d %d", &fecha.dia, &fecha.mes, &fecha.anno);
```

### Pregunta 3

Correcta

Puntúa 1,00  
sobre 1,00

¿Qué escribe el siguiente programa?

```
#include <stdio.h>
typedef enum {verde=1, rojo, amarillo} Color;
int f (int a) {
    return a%2;
}
int g (int b) {
    return b%3;
}
int h (int c) {
    return c%5;
}
int main (){
    int m = 8, color=2;
    switch (color){
        case verde:
            printf ("%d", f (m));
            break;
        case rojo:
            printf ("%d", g (m));
            break;
        case amarillo:
            printf ("%d", h (m));
            break;
    }
    return 0;
}
```

Seleccione una:

- ☐ 3
- ☐ 1
- ☒ 2 

#### EXPLICACIÓN:

Como color es 2, se ejecuta el caso rojo, es decir, se escribe g(8), que es el resto de la división de 8 entre 3, es decir, 2.

La respuesta correcta es: 2

**Pregunta 4**

Correcta

Puntúa 1,00  
sobre 1,00

Si queremos implementar un menú de forma que se solicite una opción hasta que el usuario pulse 3 (opción de salir del programa), ¿qué instrucciones deberíamos utilizar en las líneas marcadas con (1) y (2) en el siguiente código para que funcione correctamente?

```
#include <stdio.h>
#include <stdlib.h>
```

```
typedef enum {altaUsuario=1, nuevaSesion, salir} MenuInicial;
```

```
int main() {
```

```
int opcion;
```

```
//(1)
```

```
{
```

```
printf ("\n");
```

```
printf ("1. Dar de alta a una nueva persona.\n");
```

```
printf ("2. Iniciar una sesion para un usuario.\n");
```

```
printf ("3. Salir del programa.\n\n");
```

```
printf ("Elija una opcion: ");
```

```
scanf ("%d", &opcion);
```

```
switch (opcion)
```

```
{
```

```
case altaUsuario:
```

```
printf("\nAlta usuario\n\n");
```

```
break;
```

```
case nuevaSesion:
```

```
printf("\nNueva sesion.\n\n");
```

```
break;
```

```
case salir:
```

```
printf ("\nAdios.\n\n");
```

```
break;
```

```
default:
```

```
printf ("\nOpcion incorrecta.\n\n");
```

```
}
```

```
} //(2)
```

```
return 0;
```

```
}
```

Seleccione una:

- ☒ (1) do (2) while (opcion!=salir); ✓
- ☐ (1) do (2) while (opcion==altaUsuario || opcion==nuevaSesion);
- ☐ (1) while (opcion==salir) (2) -

#### SOLUCIÓN:

La respuesta (1) while (opcion==salir) (2) - no es correcta dado que el bucle termina cuando opcion es cualquier otra opción que no sea salir, es decir, justo el comportamiento contrario al que se busca.

La respuesta (1) do (2) while (opcion==altaUsuario || opcion==nuevaSesion); tampoco es válida ya que si el usuario introduce una opción incorrecta (que no sea altaUsuario ó nuevaSesion) el bucle termine, pero queremos que siga, y vuelva a pedir al usuario que introduzca una opción.

La respuesta correcta es: (1) do (2) while (opcion!=salir);



### Pregunta 5

Correcta

Puntúa 1,00  
sobre 1,00

Estudia las siguientes declaraciones:

```
typedef struct {  
    int c;  
} H;
```

```
typedef struct {  
    int a;  
} N;
```

```
typedef struct {  
    int y;  
    N b;  
    H s;  
} V;
```


```
typedef struct {  
    int z;  
    V c;  
} M;
```

```
typedef struct {  
    V x;  
    M a;  
    V r;  
} U;
```

U mi;

¿Cuál de los siguientes accesos es correcto?

Seleccione una:

- ☒ mi.a.c.y 
- ☐ mi.x.b.c
- ☐ mi.r.s.a

SOLUCIÓN:

mi es de tipo U

En mi, dentro de la estructura U, se puede referenciar al miembro a, que es de tipo M

En a, dentro de la estructura M, se puede referenciar al miembro c, que es de tipo V

En c, dentro de la estructura V, se puede referenciar al miembro y.

La respuesta correcta es: mi.a.c.y

## Pregunta 6

Correcta

Puntúa 1,00  
sobre 1,00

¿Qué código habría que añadir al siguiente programa para que guardase el nombre (no compuesto) y primer apellido del usuario en una sola instrucción?

```
#include <stdio.h>
```

```
int main() {
```

```
    char nombre[10];  
    char apellido[10];
```

```
    printf("Introduce tu nombre y tu apellido (separado por un espacio). Despues  
    pulsa intro: ");  
    //Leer el nombre (no compuesto) y primer apellido en una instrucción
```

```
    printf("Tu nombre es: %s\nY tu apellido: %s", nombre, apellido);
```

```
}
```

Seleccione una:



```
gets(nombre, apellido);
```



```
scanf("%s %s", nombre, apellido);
```



```
scanf("%s", nombre, apellido);
```

Respuesta correcta

SOLUCIÓN:

En este caso no se recomienda usar gets ya que éste lee toda la cadena seguida hasta el salto de línea. Por ello, se debe utilizar scanf. Además, el formato de lectura de scanf debe ser:

```
scanf("%s %s", nombre, apellido);
```

Para que lea las dos cadenas en la misma instrucción

La respuesta correcta es:

```
scanf("%s %s", nombre, apellido);
```

**Pregunta 7**

Correcta

Puntúa 1,00  
sobre 1,00

¿Cuál de las siguientes afirmaciones es correcta para el siguiente prototipo?

```
int func (int * p, int m);
```

Seleccione una:

- ☐ El primer argumento se pasa por valor y el segundo por referencia
- ☒ El primer argumento se pasa por referencia y el segundo por valor ✓
- ☐ Ninguna de las otras

**EXPLICACIÓN:**

En C una variable se pasa por referencia cuando se pasa su dirección como argumento de una función. Por lo tanto, como el primer parámetro es un puntero a int y el segundo un int, el primer argumento se pasa por referencia y el segundo por valor.

La respuesta correcta es: El primer argumento se pasa por referencia y el segundo por valor

**Pregunta 8**

Correcta

Puntúa 1,00  
sobre 1,00

Dado el siguiente programa, seleccione que muestra por pantalla:

```
#include <stdio.h>
void funcion1 (int * p)
{
    (*p)++;
    p++;
    (*p)--;
}
int main ()
{
    int num[6] = {10,14,23,17,15,8};
    int *p = &num[4];
    funcion1(p);
    printf ("%d", num[4]-num[5]);
    return 0;
}
```

Seleccione una:

- ☒ 9 ✓
- ☐ 8
- ☐ 4

**EXPLICACIÓN:**

Tras llamar a funcion1 la tabla num quedaría de la forma {10,14,23,17,16,7}

La respuesta correcta es: 9

**Pregunta 9**

Correcta

Puntúa 1,00  
sobre 1,00

1.¿Cómo completarías la instrucción scanf en el siguiente programa?

```
#include <stdio.h>
```

```
int main() {
```

```
    int numero;
```

```
    printf( "Introduce un numero: ");
```

```
    scanf( "%d",
```

```
    printf( "\nHas introducido el numero %d\n", numero);
```

```
    return 0;
```

```
}
```

Seleccione una:

- ☒ &numero); ✓
- ☐ );
- ☐ numero);

**SOLUCIÓN:**

Se ha de especificar la variable donde se va a guardar la información precedida por &amp;

La respuesta correcta es: &amp;numero);

**Pregunta 10**

Correcta

Puntúa 1,00  
sobre 1,00

¿Cuántas veces se ejecuta el siguiente bucle for?

```
for (i=1; i<=10; i*=2)  
    printf("Hola\n");
```

Seleccione una:

- ☐ 5
- ☐ 10
- ☒ 4 ✓

**SOLUCIÓN:**

El bucle se ejecuta 4 veces, para i=1, 2, 4 y 8

La respuesta correcta es: 4

### Pregunta 11

Correcta

Puntúa 1,00  
sobre 1,00

¿Cuál es la salida de la instrucción `printf("*****\n** hola **\n*****\n");`?

Seleccione una:

☐

\*\*\*\*\*

☐

\*\*\*\*\*

\*\* hola \*\*

\*\*\*\*\*

☐

\*\*\*\*\* hola \*\*\*\*\*

#### SOLUCIÓN:

Los saltos de línea hacen que se muestre un mensaje con tres líneas

La respuesta correcta es:

\*\*\*\*\*

\*\* hola \*\*

\*\*\*\*\*

## Pregunta 12

Incorrecta

Puntuaje -0,50  
sobre 1,00

El problema del siguiente proyecto se pone de manifiesto durante ...

1. El archivo receta.h:

```
double receta (int x, int * y);
```

2. El archivo receta.c:

```
#include "receta.h"
double receta (int x, int y) {
    return (x+y)/2.0;
}
```

3. El archivo main.c:

```
#include <stdio.h>
#include "receta.h"
int main () {
    double resultado = receta (1,2);
    printf ("%2lf\n", resultado);
    return 0;
}
```

Seleccione una:

- ☐ la compilación de receta.c
- ☒ la compilación de main.c ✗
- ☐ el enlazado del proyecto

### EXPLICACIÓN:

El prototipo de la función receta en el archivo receta.h

```
double receta (int x, int * y);
```

no coincide con la definición del archivo receta.c (el tipo del segundo parámetro es int en la declaración e int \* en la definición):

```
double receta (int x, int y) {
    return (x+y)/2.0;
}
```

Esto da lugar a un error al compilar el archivo receta.c.

La respuesta correcta es: la compilación de receta.c

**Pregunta 13**

Correcta

Puntúa 1,00  
sobre 1,00

El siguiente programa escrito en C no compila bien por culpa de la función *iniContacto()*:

```
#define DIM 64
```

```
typedef struct {  
    int dia;  
    int mes;  
    int anyo;  
} Fecha;
```

```
typedef struct {  
    char nombre [DIM];  
    char apellidos [DIM];  
    Fecha fecha_nacimiento;  
} Contacto;
```

```
void iniContacto(Contacto* cnt, char* nmb, char* apII, int dn, int m  
n, int an){
```

```
    strcpy(cnt->nombre, nmb);  
    strcpy(cnt->apellidos, apII);  
    cnt->fecha_nacimiento->dia= dn;  
    cnt->fecha_nacimiento->mes= mn;  
    cnt->fecha_nacimiento->anyo= an;  
    return;  
}
```

```
int main(int argc, char** argv) {
```

```
    Contacto amigo;
```

```
    iniContacto(&amigo, "Manuel", "Molina", 3, 3, 1999);
```

```
    return 0;  
}
```

¿Cuál de las siguientes afirmaciones es correcta?

Seleccione una:

- ☒ Dentro de la función *iniContacto()* deberían sustituirse todos los "*fecha\_nacimiento->*" por "*fecha\_nacimiento.*". ✓
- ☐ Dentro de la función *iniContacto()* deberían sustituirse todos los "*cnt->*" por "*cnt.*".
- ☐ Dentro de la función *iniContacto()* deberían sustituirse todos los "->" por ".".

Respuesta correcta

El problema es que se pretende acceder a los campos de la fecha de nacimiento como si esta fuera un puntero a *Fecha*, pero el campo *fecha\_nacimiento* de *Contacto* es de tipo *Fecha*.

La respuesta correcta es: Dentro de la función *iniContacto()* deberían sustituirse todos los "*fecha\_nacimiento->*" por "*fecha\_nacimiento.*".

#### Pregunta 14

Correcta

Puntúa 1,00  
sobre 1,00

¿Qué efecto tiene la instrucción `break` cuando aparece dentro de un bucle?

Seleccione una:

- ☒ Abandona el bucle y sigue ejecutando la siguiente instrucción tras el bucle ✓
- ☐ Permite volver a evaluar la condición de salida del bucle
- ☐ Termina la ejecución del programa

SOLUCIÓN:

La instrucción `break` en un bucle hace que se abandone el bucle y se siga ejecutando la siguiente instrucción tras el bucle

La respuesta correcta es: Abandona el bucle y sigue ejecutando la siguiente instrucción tras el bucle



**Pregunta 15**

Correcta

Puntúa 1,00  
sobre 1,00

En el siguiente programa en C:

```
#define MESESA 32
#define MAX_MSJ 32
```

```
typedef struct {
    int dia;
    int mes;
    int anno;
} Fecha;
```

```
int main()
{
    Fecha fecha;
    int diasMes[MESESA]={31,29,31,30,31,30,31,31,30,31,30,31};
    int fechaOk = 0;

    /***** Codigo para obtener una fecha adecuada del usuario *****/

    printf("\nLa fecha es correcta\n\n");

    return 0;
}
```

Se considera utilizar una de las siguientes alternativas para conseguir una fecha adecuada del usuario :

**Alternativa A**

```
do{
    printf("Fecha (dd mm aa): ");
    scanf("%d %d %d", &fecha.dia, &fecha.mes, &fecha.anno);
    if ((0 > fecha.anno) ||
        (0 >= fecha.mes) || (MESESA < fecha.mes) ||
        (0 >= fecha.dia) || (diasMes[fecha.mes-1]<fecha.dia))
        printf("Fecha erronea\n");
    else fechaOk = 1;
} while (!fechaOk);
```

**Alternativa B**

```
while (!fechaOk){
    printf("Fecha (dd mm aa): ");
    scanf("%d %d %d", &fecha.dia, &fecha.mes, &fecha.anno);
    if ((0 > fecha.anno) ||
        (0 >= fecha.mes) || (MESESA < fecha.mes) ||
        (0 >= fecha.dia) || (diasMes[fecha.mes-1]<fecha.dia))
        printf("Fecha erronea\n");
    else fechaOk = 1;
};
```

¿Cuál de las siguientes afirmaciones es correcta?

Seleccione una:

- ☒ Las dos alternativas son correctas ✓
- ☐ La alternativa B es correcta pero no la A
- ☐ La alternativa A es correcta pero no la B

Respuesta correcta

Como la variable *fechaOk* se ha inicializado a 0 en la declaración las dos alternativas son correctas

La respuesta correcta es: Las dos alternativas son correctas

### Pregunta 16

Correcta

Puntúa 1,00  
sobre 1,00

Si tenemos definida la estructura Fecha:

```
typedef struct{
```

```
    int dia;  
    int mes;  
    int anyo;
```

```
}Fecha;
```

Y tenemos una función de prototipo:

```
void cambiaFecha(Fecha *f);
```

¿Cómo podemos acceder a la información del día, mes y año pasada como argumento dentro de la función?

Seleccione una:

- ☐ Accediendo a los campos de la forma: f.dia, f.mes, f.anyo.
- ☒ Accediendo a los campos de la forma: f->dia, f->mes, f->anyo. ✓
- ☐ Utilizando las variables dia, mes y anyo.

EXPLICACIÓN:

Puesto que la función recibe un puntero a la estructura, la manera correcta entre las opciones de acceder a sus campos es con ->.

La respuesta correcta es: Accediendo a los campos de la forma: f->dia, f->mes, f->anyo.

### Pregunta 17

Correcta

Puntúa 1,00  
sobre 1,00

Si definimos la función `areaTriangulo()` como sigue:

```
double areaTriangulo (int base, int altura) {  
    double area;  
    area=base*altura/2.0;  
    return area;  
}
```

¿Cómo debemos llamarla dentro de main para que se escriba el valor devuelto en la pantalla?

```
#include <stdio.h>
```

```
int main () {  
    int ba, al;  
    double ar;  
    printf("Introduce la base del triangulo: ");  
    scanf("%d", &ba);  
    printf("Introduce la altura del triangulo: ");  
    scanf("%d", &al);  
  
    // Llamada a la función  
  
    printf("El area del triangulo de base %d y altura %d es %.2lf", ba, al, ar);  
    return 0;  
}
```

Seleccione una:

- ☒ `ar=areaTriangulo(ba, al);` ✓
- ☐ `ar=areaTriangulo();`
- ☐ `areaTriangulo();`

#### EXPLICACIÓN:

La función necesita dos argumentos de tipo int para ejecutarse y devuelve un valor de tipo double. Por lo tanto, la llamada correcta es:

```
ar = areaTriangulo (ba, al);
```

pues la variable ar de main es de tipo double y las variables ba y al son ambas de tipo int.  
La respuesta correcta es: `ar=areaTriangulo(ba, al);`

### Pregunta 18

Incorrecta

Puntuaje -0,50  
sobre 1,00

¿Qué vale  $p < q$  dadas las siguientes declaraciones?

```
int a[] = {5,15,34,54,14,2,52,72};  
int *p = &a[1];  
int *q = &a[5];
```

Seleccione una:

- ☐ 0
- ☒ ninguna de las otras ✖
- ☐ 1

#### EXPLICACIÓN:

Al sustituir los valores de  $p$  y  $q$ , vemos que la condición lógica  $p < q$  es equivalente a  $\&a[1] < \&a[5]$ .

La condición es cierta porque los elementos de los arrays ocupan posiciones de memoria consecutivas y el elemento 1 está antes que el 5.

Como la condición es cierta, el valor devuelto por el operador  $<$  es 1.

La respuesta correcta es: 1

### Pregunta 19

Correcta

Puntúa 1,00  
sobre 1,00

¿Qué muestra por pantalla el siguiente programa?

```
#include <stdio.h>
```

```
void funcion1(char letrax, char *letray)
{
    letrax++;
    (*letray)++;
}
```

```
int main()
{
    char car='b';
    funcion1(car, &car);
    printf("%c %c", car, car);
    return 0;
}
```

Seleccione una:

- ☐ d d
- ☒ c c ✓
- ☐ c d

#### EXPLICACIÓN:

La variable car se pasa por valor y por referencia a la función funcion1.

Sólo la variable que recibe el valor por referencia (letray) hace afecta al valor de car una vez se ha salido de la función.

Por lo tanto car se aumenta en uno y pasa a valer 'c'.

La respuesta correcta es: c c

## Pregunta 20

Correcta

Puntúa 1,00  
sobre 1,00

¿Qué instrucción nos permite declarar la siguiente matriz?

12 3 6

1 4 78

3 8 112

Seleccione una:

- ☒ `int matriz[3][3]={ {12, 3, 6}, {1, 4, 78}, {3, 8, 112}};` ✓
- ☐ `int matriz[3][3]={ {12, 3, 6}{1, 4, 78}{3, 8, 112}};`
- ☐ `int matriz[3][3]={12, 3, 6}, {1, 4, 78}, {3, 8, 112};`

### SOLUCIÓN:

Sólo la respuesta correcta cumple la sintaxis necesaria para declarar e inicializar la matriz especificada. En la otras faltan, o bien llaves, o bien comas.

La respuesta correcta es: `int matriz[3][3]={ {12, 3, 6}, {1, 4, 78}, {3, 8, 112}};`

Volver a: General ➡