



Nombre y apellidos: _____

Notas previas:

- Escribe tu **nombre** y **apellidos** en esta hoja e inmediatamente en todas las suplementarias, incluso las de sucio. El no hacerlo puede suponer tu expulsión.
- Todos los alumnos implicados en una copia** de un ejercicio **tendrán una nota final de 0**. El alumno es responsable de velar por su examen. Es decir **tanto el que copia como el que se deja copiar (ya sea de manera activa o pasiva) recibirán el mismo castigo sin que exista atenuante alguno**.
- Puedes utilizar **lápiz**. No puedes tener un **móvil** encendido ni utilizar **calculadora**.
- “Recibe” es distinto de “lee del teclado”. “Devuelve” es distinto de “escribe” o “muestra en pantalla”.

- [1 punto]** Indica en un **recuadro** lo que escribirá el siguiente programa en lenguaje C, explicando el razonamiento y explicitando los cálculos:

```
#include <stdio.h>
void main (void)
{
    int a=0, b=1, c=2, d=-1;
    printf ("1. %d\n", a|c);
    printf ("2. %d\n", a||c);
    printf ("3. %d\n", b&c);
    printf ("4. %d\n", b&& c);
    printf ("5. %d\n", (a<b)+c);
    printf ("6. %d\n", !a+~d);
}
```

Notas recordatorias:

- El operador **|** realiza una operación lógica **Or bit a bit** de dos números enteros
- El operador **||** realiza una operación lógica **Or** de dos números enteros
- El operador **&** realiza una operación lógica **And bit a bit** de dos números enteros
- El operador **&&** realiza una operación lógica **And** de dos números enteros
- El operador **!** realiza una operación **Not** de un número entero
- El operador **~** **complementa a uno** un número, es decir, realiza una operación lógica **Not bit a bit** de un número entero
- Los números negativos utilizan representación con **complemento a dos**

- [1,5 puntos]** **Codifica** un programa C que, utilizando sólo **dos variables**, lea **tres** números enteros **distintos** y escriba en pantalla si están ordenados en sentido **creciente**, en sentido **decreciente** o están **desordenados**.

Sólo se permite utilizar dos variables llamadas **x1** y **x2** en todo el programa. Cualquier otra solución se considerará completamente incorrecta.

No se verificará si los números se introducen correctamente ni si son distintos por lo que, si no se cumplen estas condiciones, no importa que el programa no funcione.

- [2 puntos]** **Diseña el diagrama de flujo** y **codifica** una función C que, dado un número natural, calcule y devuelva la raíz cuadrada entera del número dado.

Definimos **raíz cuadrada entera** de un número n al mayor número x que cumpla $x^2 \leq n$.

Como algoritmo se propone, partiendo de 0, ir incrementando el valor de x hasta encontrar el primer valor que haga falsa la expresión. La solución será entonces ese valor menos uno.

Codifica además un **programa** que lea un número natural y obtenga la raíz cuadrada entera utilizando esta función, mostrando el resultado en pantalla.

4. [2,5 puntos] **Codifica** un programa en lenguaje C que vaya calculando los términos de la **sucesión de Fibonacci** a partir del 3º y los vaya preguntado al usuario numerándolos (ver ejemplo). El programa finalizará cuando el usuario introduzca un número distinto al término correspondiente en ese momento.

Según la *Wikipedia* la sucesión de Fibonacci es una sucesión infinita de números naturales

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...

donde el primer elemento es 0, el segundo es 1 y cada elemento restante es la suma de los dos anteriores. A cada elemento de esta sucesión se le llama número de Fibonacci.

Ejemplo de ejecución (**procura que los textos de tu programa coincidan literalmente**) con los datos introducidos por el usuario en **negrita**:

```
Los dos primeros terminos son: 0 y 1
Introduce el termino 3: 1
Introduce el termino 4: 2
Introduce el termino 5: 3
Introduce el termino 6: 5
Introduce el termino 7: 8
Introduce el termino 8: 12
El termino 8 es el 13 y no el 12
```

5. [1 punto] **Diseña el diagrama de flujo de la cabecera**, incluyendo *nombre*, parámetros de *entrada y salida* y *valor devuelto*, y el **prototipo** en el lenguaje C para las siguiente especificaciones:
- Función **LogB** que recibe un número X y una base B y devuelve el logaritmo en base B de X. Utilizará números reales.
 - Función **MayS** que recibe tres cadenas s0, s1 y s2 y devuelve un 0, un 1 ó un 2 según sea la mayor la primera, la segunda o la tercera. Si dos cadenas son iguales podrá devolver indistintamente el número correspondiente a cualquiera de las dos.
 - Función **OsoPanda** que recibe un número N y muestra en pantalla un dibujo con asteriscos de un oso panda de altura N (asteriscos).
 - Función **LeeV** que recibe un vector V de enteros y el número de elementos del vector N y lee una lista de números al vector V devolviendo en M la cantidad de números leídos.

Nota importante: no es necesario diseñar ni codificar las funciones.

6. [2 puntos] (Elige entre esta pregunta o la siguiente) **Disponemos** de las funciones:

DiaSis	Obtiene día, mes y año del reloj del sistema (de hoy)
DiaJul	Devuelve una fecha en formato numérico (juliano)
DiaGrg	Convierte de formato juliano a formato día-mes-año (gregoriano)
DiaSem	Devuelve el ordinal del día de la semana de una fecha juliana numerados del 0 al 6.

Sus prototipos, especificados en el fichero “**fechas.h**”, son los siguientes:

```
void DiaSis (int *dd, int *mm, int *aa);
long DiaJul (int dd, int mm, int aa);
void DiaGrg (long jul, int *dd, int *mm, int *aa);
int DiaSem (long jul);
```

Codifica un programa en lenguaje C que escriba la fecha para el primer y tercer lunes de cada mes a partir de la fecha del sistema (incluida). El programa leerá el número de fechas a mostrar, tal y como se muestra en el siguiente ejemplo (supóngase que se ejecuta el 5 de septiembre de 2008):

Introduce cuantas fechas quieres mostrar: 3 15/09/2008 06/10/2008 20/10/2008
--

7. [2 puntos] (Elige entre esta pregunta o la anterior) **Diseña el diagrama de flujo de la cabecera y codifica** una función en lenguaje C que reciba el *vector de reales* **v** y el número de elementos **n**, e invierta el orden de sus elementos.

Ejemplo:

Vector **v** antes de la llamada (**n** = 5):

0	1	2	3	4
3.51	9.12	6.30	0.11	7.88

Vector **v** después de la llamada:

0	1	2	3	4
7.88	0.11	6.30	9.12	3.51