Dpto. Física y Arquitectura de Computadores

Fundamentos de Programación

GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Examen Parcial 13-DIC-2012

Alumno (apellidos y nombre)	D.N.I.	Firma

PREGUNTAS (0.5 PUNTOS CADA UNA) ----------

1) Dado el siguiente fragmento de programa, responda razonadamente a las siguientes preguntas:

```
int a=0, b=5;
int *c=NULL, *d=NULL;
c=&a;
d=c;
*d= *c + *(&b);
```

a. ¿las variables **c** y **d** se almacenan en la misma dirección de memoria?

No c se almacena en la dir. de memoria de a y d en la dir. de memoria

b. ¿las variables **c** y **d** referencian a la misma dirección de memoria?

c. ¿La sentencia ***c = 4;** modifica el contenido de la variable **a**, de la variable **b**, de las dos o de ninguna?

d. ¿Qué valor tomará **a** al finalizar la ejecución? ¿y **b**?

2) ¿Cuál es el resultado del siguiente programa? – redondea opción correcta

```
#include <stdio.h>

main(){
  int a[5]={10,20,30,40,50};
  int *p =a;
 printf("%d", *(p+2));
}
```

a. Imprime basura (valor no predecible)
b. Imprime 30
c. Produce un error en tiempo de compilación
d. Ninguno de los anteriores

3) ¿Cuál es el resultado del siguiente programa? – redondea opción correcta

```
#include <stdio.h>

main(){
  int a[5]={10,20,30,40,50};
  printf("%d", *a++);
}
```

a. Imprime 20
b. Imprime 10
c. Produce un error en tiempo de compilación
d. Produce un error en tiempo de ejecución

4) ¿Cuál es el resultado del siguiente programa? – redondea opción correcta

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

main(){
  int *p=NULL;
 asigmem( p, 3);
 p[0]=10; p[1]=20; p[2]=30;
  printf("%d", p[1]);
}
void  asigmem(int *p, int t){
   p=(int*) malloc (t * sizeof(int));
}
```

a. Imprime 20
b. Imprime basura (valor indeterminado)
c. Produce un error en tiempo de compilación
d. Produce un error en tiempo de ejecución

5) ¿Cuál es el resultado del siguiente programa? – redondea opción correcta

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

main(){
  char m[10]="abc";
  FILE *pf=NULL;
  int i=0;
  pf = fopen("datos", "w");
  for (i=0; m[i]; i++)
     fwrite(&m[i], sizeof(char), 1, pf);
 fclose(pf);
}
```

a. Un fichero *datos* con ***a***
b. Un fichero *datos* con basura porque no se puede emplear **fwrite** para escribir datos de tipo **char,** uno a uno, en un fichero.
c. Un fichero *datos* con ***abc***
d. Un fichero *datos* con basura porque el primer parámetro de **fwrite** está mal especificado.

PROGRAMAS ----------

6) (3 PUNTOS) Crear un programa que permita guardar estructuras de tipo "FichaLibro" en un fichero binario.

```
typedef struct EstructuraFichaLibro{
  char titulo[256];
  char autor[256];
  char ISBN[256];
  char Estante[256];
  int prestado;
 char cliente[256];
} FichaLibro;
```

Para ello crearemos las siguientes funciones:

a) Función **ObtenerDatos** que solicitará los datos de la estructura **FichaLibro** al usuario por teclado.

b) Función **AnyadirAFichero** que guardará la estructura en el fichero "**biblioteca.bin**".

7) (4 PUNTOS) Crear un programa que lea del fichero creado anteriormente ("biblioteca.bin"), de manera que almacene los datos en un array de estructuras creado de forma dinámica.

Para ello crearemos las siguientes funciones:

a) **CalculaElementos** . Esta función se encargará de calcular cuántas estructuras hay almacenadas en el fichero, en función del tamaño del fichero.
b) **ReservaMemoria** será la función encargada de reservar memoria dinámica para el array de estructuras de tipo "**FichaLibro**".
c) **LeerDatos** será la función encargada de leer los datos de las estructuras del fichero y almacenarlas en el array de estructuras creado anteriormente.
d) **OrdenarArray** será la función encargada de ordenar el array de estructuras en orden alfabético por el campo autor de las estructuras. Se puede utilizar cualquiera de los métodos de ordenación vistos en clase.

Dpto. Física y Arquitectura de Computadores

Fundamentos de Programación

GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Examen Final ENERO-2013

Alumno (apellidos y nombre)	D.N.I.	Firma

Descárguese de la página Web de la asignatura el fichero comprimido (extensión .ZIP) que corresponda con las partes a las que se presenta.

- **Solo Parte 4 (Listas/Pilas/Colas/Arboles)** -> LaunchMenuP4.zip
- **Solo Partes 3 (Ficheros) y 4 (Listas/Pilas/Colas/Arboles)** -> LaunchMenuP3-4.zip
- **Todas las partes ->** LaunchMenuALL.zip

Los ficheros comprimidos contienen un proyecto de Visual Studio escrito en lenguaje C. Este proyecto corresponde a un programa que gestiona menús de una forma similar a como se hizo la práctica de gestión de menús (se adjunta enunciado de la práctica para aclaraciones). La diferencia con el enunciado de la práctica es que en este programa se utilizan listas lineales simplemente enlazadas para almacenar tanto los menus, como los meuitems.

Dependiendo de las partes a las que se deba presentar, complete el programa para que funcione correctamente. Las partes a completar están indicadas en el código fuente mediante el comentario:

//COMPLETAR COMPLETAR COMPLETAR

A continuación se enumeran las funciones a completar dependiendo de las partes a realizar:

- **Solo Parte 4**
  1. `int LoadMenuConfig(tllse *lmenus, tllse *lmenuitems)`
  2. `t_menu *GetMenu(int menu_id,tllse *lmenus)`
  3. `t_menuitem *GetMenuitem(int menu_id, int orden, tllse *lmenuitems)`
  4. `int GetNumMenuitems(int menu_id, tllse *lmenuitems)`
- **Partes 3 y 4**
  1. `int LoadMenuConfig(tllse *lmenus, tllse *lmenuitems)`
  2. `t_menu *GetMenu(int menu_id,tllse *lmenus)`
  3. `t_menuitem *GetMenuitem(int menu_id, int orden, tllse *lmenuitems)`
  4. `int GetNumMenuitems(int menu_id, tllse *lmenuitems)`
  5. `char *ReadLabelContent(FILE *fd)`
- **Todo**
  1. `int LoadMenuConfig(tllse *lmenus, tllse *lmenuitems)`
  2. `t_menu *GetMenu(int menu_id,tllse *lmenus)`
  3. `t_menuitem *GetMenuitem(int menu_id, int orden, tllse *lmenuitems)`
  4. `int GetNumMenuitems(int menu_id, tllse *lmenuitems)`
  5. `char *ReadLabelContent(FILE *fd)`

FPALC Tg 7