Relación de prácticas de la asignatura
METODOLOGÍA DE LA PROGRAMACIÓN
Segundo Cuatrimestre
Curso 2014-2015

1º Grado en Informática

Práctica 3: Recursividad, Ficheros y Argumentos en Línea de órdenes

Objetivos

- Practicar conceptos básicos sobre ficheros de texto, ficheros binarios en C y recursividad.
- Usar argumentos en línea de órdenes para que los programas puedan recibir argumentos en el momento de ser ejecutado.

Recomendaciones

- Pasa a tus funciones el nombre del fichero como parámetro y ábrelo y ciérralo dentro de la función.
- Organiza los ejercicios 8 y 9 en al menos tres ficheros .c (*main.c, ficheros.c, util.c*) y dos ficheros .h (*ficheros.h y util.h*)

Temporización recomendada

3 sesiones de prácticas

- 1ª sesión: Ejercicios de recursividad
- 2ª sesión: Ejercicios de ficheros de texto
- 3ª sesión: Ejercicios de ficheros binarios

¿Qué hay que entregar?

- El análisis y diseño del ejercicio 5 y de la función contar número de registros del ejercicio 8 (texto).
- El análisis consistirá en el estudio del problema que plantea el ejercicio.
  - Qué datos de entrada necesita y de qué tipo son
  - Cómo van a llegar esos datos
  - Qué resultado se va a obtener y de qué tipo es
  - Cómo se obtiene el resultado a partir de los datos de entrada
  - Cómo se va a presentar al usuario el resultado final
  - Ejemplo de que la solución propuesta funciona, utilizando los nombres dados a los datos

- El diseño incluirá un algoritmo en pseudocódigo o diagrama de flujo que resuelva el problema y que servirá como base para la posterior codificación.
  - Recordad que el diseño es independiente del lenguaje de programación utilizado

¿Cuándo hay que entregar el análisis y el diseño?

Grupo	P3, P5, P6 y P9	P4, P8	P1, P2 y P7
Fecha	14/04/2015	15/04/2015	17/04/2015

Recursividad

1. La siguiente función recursiva se encarga de multiplicar los números comprendidos entre [inicio,limite):

```
int funcionRecursiva (int inicio, int limite) {
int retorno;
if (inicio > limite)
  retorno = -1;
else
  if (inicio == limite)
    retorno = 1;
  else
    retorno = inicio * funcionRecursiva(inicio+1, limite);
return retorno;
}
```

Se pide:

1. Identificar el caso de parada de la función *funcionRecursiva.*
2. Realizar un esquema sobre las variaciones en la pila de control.
3. ¿Qué resultado daría la llamada *funcionRecursiva(14,10)*?
4. ¿Qué resultado daría la llamada *funcionRecursiva(4,7)*?

2. Implementa una función recursiva que calcule cuántos dígitos pares tiene un número.
   - Por ejemplo: el número 347 tiene un dígito par.

3. Dada una cadena *c,* diseña una función recursiva que cuente la cantidad de veces que aparece un carácter *x* en *c*.
   - *Ej: para c = "elementos de programación" y x = 'e', el resultado es 4*.

4. Implementa una función recursiva que calcule el producto de los elementos de un vector de enteros.

5. Codifica una función recursiva que permita calcular el valor de π usando la serie de Leibniz:
   π = (4 - 4/3 + 4/5 - 4/7 + 4/9 -4/11 + 4/13 ....)
   - El programa pedirá al usuario que introduzca el número "*n*" de términos a usar en la aproximación.

Ficheros de texto

6. Codifica un programa en C que, utilizando funciones, abra un fichero en modo **texto**, cuyo nombre se pedirá al usuario, y genere un nuevo fichero llamado "mayusculas-nombreFicheroDeEntrada.txt", que tenga el mismo contenido que el fichero original pero en mayúsculas.

7. Codifica un programa en C que, utilizando funciones, modifique un fichero de **texto** para que cualquier secuencia de espacios en blanco se sustituya por un único espacio en blanco. El nombre del fichero se pedirá al usuario.

8. Construye un programa que gestione mediante ficheros de **texto** el *stock* de libros de una librería. Para cada libro se almacenarán tres líneas en un fichero de texto (*stock*):
   - en la primera línea el *título*,
   - en la segunda línea el *autor,*
   - y en la tercera línea *el precio* y las *unidades* disponibles del libro.

   El programa contará con un menú que permitirá realizar las siguientes operaciones:
   - Comprobar la existencia de un determinado libro buscando por su título.
   - Introducir un nuevo libro en el *stock*.
   - Contar el número de libros en el *stock*.
   - Listar los libros almacenados en el *stock* almacenándolos previamente en un vector dinámico.
   - Vender un libro buscándolo por su título.
   - Borrar aquellos registros con 0 unidades disponibles.
   - Salir

Ficheros binarios

9. Construye un programa que gestione, mediante ficheros **binarios**, el *stock* de libros de una librería. Para cada libro se almacenará la siguiente estructura:

```
struct libro {
  char titulo[100];
  char autor[50];
  float precio;
  float unidades;
}
```

   El programa contará con un menú que permitirá realizar las siguientes operaciones:
   - Comprobar la existencia de un determinado libro buscando por su título.
   - Introducir un nuevo libro en el *stock*.
   - Contar el número de libros en el *stock*.
   - Listar los libros almacenados en el *stock* almacenándolos previamente en un vector dinámico
   - Vender un libro buscándolo por su título sin cargar el fichero en memoria
   - Borrar aquellos registros con 0 unidades disponibles.
   - Salir

Argumentos en línea de órdenes

10. Modifica el ejercicio 6 para que el programa reciba en la línea de órdenes el fichero a convertir a mayúsculas.