Prácticas de la Asignatura

METODOLOGÍA Y TECNOLOGÍA DE LA PROGRAMACIÓN(curso 2006-2007).

Módulo II (2º Cuatrimestre)

Febrero de 2007.

1º Ingeniero Técnico en Informática de Gestión.
1º Ingeniero Técnico en Informática de Sistemas.

Documentación de las prácticas.

Cada práctica deberá documentarse de la siguiente forma:

- En el fichero fuente que contiene la función ***main*** se especificará:
  - Nombre y especialidad del alumno.
  - Enunciado de la práctica.
- Cada función, se presentará por separado con la siguiente información en la cabecera:
  - Nombre del módulo al que pertenece.
  - Nombre de la función.
  - Nombre del autor.
  - Fecha de la última modificación.
  - Parámetros de entrada (nombre, tipo y descripción).
  - Parámetros de salida (nombre, tipo y descripción).
  - Valor devuelto.
  - Funciones a las que llama (no incluir funciones estándar del lenguaje).
  - Comentarios adicionales
- Fecha de entrega.

Consideraciones sobre estilo de programación.

Las siguientes sugerencias no son obligatorias, pero influirán de forma positiva en la evaluación de la parte práctica de la asignatura.

- El código ha de ser legible y estar correctamente tabulado(identado), diferenciándose claramente los diferentes bloques.
- Se pondrán comentarios que expliquen las partes más complicadas de comprender en el código. Los comentarios no tienen utilidad si simplemente enuncian lo que hace el programa. Es muy positivo que el código se documente de forma autónoma.
- No se utilizarán variables globales a menos que sean estrictamente necesarias, en cualquier caso, deberá justificarse su uso.
- No se utilizarán saltos en el flujo de código, tales como instrucciones ***goto*** o salidas atípicas (***break***) de esquemas iterativos.
- Los nombres de las funciones darán una idea de su utilidad. En caso de que se use un verbo en el nombre de la función utilizad siempre el infinitivo (Ejemplo: ***buscarRegistro***). Si el nombre tiene más de una palabra, usad siempre minúsculas, excepto en el comienzo de la 2ª palabra y siguientes si las hubiese (Ejemplo: ***buscarNombrePersona***)
- Normalmente en las funciones no se utilizarán funciones de entrada/salida, los valores necesarios para su funcionamiento se recibirán como parámetros y el valor o valores a devolver se devolverán como parámetros pasados por referencia o mediante la sentencia ***return***. La única excepción serán las funciones implementadas específicamente para entrada/salida, además de la función ***main***.
- Una función ha de hacer una sola cosa, y por ello ha de poseer pocas líneas de código.

- Se utilizarán siempre los tipos de datos más apropiados para cada variable.
- Los nombres de las variables deben dar una idea de su función, y se han de escribir en minúscula (Ejemplo: ***tiempoinicial***).
- Los nombres de los tipo de dato creados por el usuario se han de escribir en mayúscula. (Ejemplo: ***DATOSPERSONA***).
- Todos los tipos de datos y prototipos de funciones serán incluidos en un fichero de cabecera global de cada programa.
- Las constantes necesarias se definirán siempre mediante la directiva ***#define***, nunca se utilizará directamente el valor numérico. Además se nombrarán con mayúscula (Ejemplo*: **#define PI 3.1416***)
- Se optimizará el código primero en función de la velocidad, y después en función del espacio, a no ser que se indique específicamente lo contrario.
- Se controlarán todos los errores que se puedan producir en la llamada a funciones estándar de biblioteca o a funciones creadas por el programador.
- Todas las estructuras de datos cuyo tamaño sea conocido en tiempo de ejecución, habrán de reservarse de forma dinámica.
- Cualquier estructura de datos creada de forma dinámica se liberará de forma inmediata cuando deje de ser útil.

Práctica 1: Ficheros

Enunciado

Para el control de las notas de una asignatura se tiene un fichero TEXTO denominado "alumnos.txt" que almacena la siguiente información de cada alumno:

- Nombre. (Cadena de 15).
- Apellido 1. (Cadena de 15).
- Apellido 2. (Cadena de 15).
- DNI. (Cadena de 10).

y un fichero BINARIO denominado "notas.bin" que almacena la siguiente información correspondiente a las notas:

- DNI. (Cadena de 10).
- Fecha. (Cadena de 15).
- Nota. (Tipo double)

Implementa un programa que realice las siguientes funciones para el mantenimiento de las notas de la asignatura:

- Añadir alumnos: Para añadir los datos de un alumno. Ha de comprobar que el alumno no existía previamente, usando el DNI como campo de búsqueda.
- Modificar los datos de un alumno dado su DNI. El DNI también podrá ser modificado.
- Listado de alumnos por pantalla.
- Poner nota a un alumno: Para ello se ha de indicar el DNI del alumno, la fecha (por ejemplo 12-07-07), y la nota. Un alumno podrá ser calificado las veces que sea necesario hasta que apruebe. Se ha de comprobar que si el alumno ya está aprobado, no podrá ser calificado de nuevo.
- Buscar un alumno dado su DNI, y mostrar todas sus calificaciones con sus fechas.
- Listado de alumnos aprobados, con su nota final.
- Eliminación de los datos de aquellos alumnos aprobados, tanto del fichero de alumnos, como del fichero de notas.

Duración de la práctica

Semanas del 5 al 9 de Marzo, del 12 al 16 de Marzo y del 19 al 23 de Marzo.

Práctica 2: Recursividad, Reserva de memoria dinámica, Ordenación y Creación de Librerías

Enunciado

1) Codificar un programa que permita calcular la suma de los n primeros números naturales, donde n será introducido por el usuario. El cálculo se realizará de dos formas distintas, usando las funciones que a continuación se detallan :

sumaRecursiva : función recursiva que calcula la suma de los primeros n números naturales de la siguiente forma:

$S_1 = 1$
$S_2 = S_1 + 2$
$S_3 = S_2 + 3$
……
$S_i = S_{i-1} + i$
.....
$S_n = S_{n-1} + n$

sumaNoRecursiva : función que calcule el valor de la suma de forma directa, es decir, aplicando la fórmula de la suma de los términos de una progresión aritmética:

$S_n = ((1 + n) / 2) * n$

2) Realizar un programa que maneje los datos de una serie de equipos de futbol que componen una división y que se deberán almacenar en un vector de estructuras del tipo DATOSEQUIPO (definida mediante **typedef**), las cuales estarán compuestas por los siguientes campos:

- **nombre**: cadena de 20 caracteres
- **partidos ganados**: entero.
- **partidos empatados**: entero.
- **partidos perdidos**: entero.
- **puntos**: entero

El programa deberá permitir al usuario realizar las opciones que a continuación se detallan:

a) Reserva de memoria. Pedir por pantalla el número de equipos. Llamar a una función que reserve memoria para la introducción de los datos.

Nombre: reservarMemoria.
Parámetros:
Un puntero a la estructura de tipo DATOSEQUIPO, pasado por referencia, a partir del cual se podrá acceder a la memoria reservada.
El número de alumnos.
Valores devueltos: 1 si no hay errores, -1 en caso contrario.

b) Introducción de los datos de los equipos.

Nombre: leerTeclado.
Parámetros:
Un puntero a la estructura de tipo DATOSEQUIPO.
Número de equipos.
Valores devueltos: Ninguno.
Nota: Los puntos se calcularán a partir de los partidos ganados(3 puntos) y empatados (1 punto).

c) Clasificación de los equipos: Función que muestre por pantalla los datos de los equipos ordenados por el número de puntos (la ordenación se realizará según el método de inserción directa)
Nombre: clasificacionEquipos
Parámetros:
Un puntero a la estructura de tipo DATOSEQUIPO.
Número de alumnos.
Valores devueltos: ninguno

e) Grabar en fichero. Llamar a una función que almacene los datos en un fichero binario.
Nombre: almacenarDatos
Parámetros:
Un puntero a la estructura de tipo DATOSEQUIPO .
Número de equipos.
Nombre del fichero.
Valores devueltos: 1 si todo va bien, -2 si hay algún error.

f) Recuperar datos de un fichero binario. Pedir el nombre del fichero al usuario. Abrirlo para lectura. Calcular el número de registros del fichero mediante la utilización de la función ***ftell*** . Llamar una función que llame a la función Reserva para reservar memoria. A continuación leerá los registros del fichero y los introducirá en la estructura de datos para la que hemos reservado la memoria.
Nombre: leerFichero
Parámetros:
Un puntero a la estructura de tipo DATOSEQUIPO pasado por referencia (porque se va a reservar memoria dinámica sobre él).
Nombre del fichero.
Número de registros a leer.
Valores devueltos: 1 si no hay errores, -1 si hay errores en la reserva de memoria, -2 si hay errores en la lectura del fichero

i) Salir

NOTA: El programa se estructurará en los siguientes ficheros:

- principal.c , contendrá el programa principal,
- entradasalida.c , contendrá las funciones de entrada salida, ya sea en ficheros, teclado o pantalla.
- metodoordenacion.c , contendrá la función de ordenación.
- miscelanea.c contendrá la función de reservar memoria dinámica y cualquier otra función implementada que no aparezca en los ficheros anteriores.

- entradasalida.h, metodoordenacion.h, miscelanea.h correspondientes a sus respectivos archivos .c.

Se creará una librería productos.a que contenga todas las funciones anteriores. Para la creación de la librería se compilarán por separado los ficheros anteriores y se unirán con el comando “ar” .

Duración de la práctica

Semanas del 26 al 29 de Marzo, del 9 al 13 de Abril y del 16 al 20 de Abril.

Práctica 3: Manejo de listas simplemente enlazadas y Makefiles.

Implementad un programa en C que genere n (valor introducido por el usuario) números aleatorios enteros comprendidos entre 1 y un número máximo dado por el usuario. Cada vez que se genere un número, se comprobará si es primo o no. En caso de que sea primo se insertará en una pila denominada “primos”, y en caso de que no lo sea, se insertará en una cola denominada “noPrimos”. Finalmente, la pila y la cola se han de fusionar en una lista simple ordenada según el valor del número, en la que cada elemento, además de contener al número, ha de contener una cadena cuyo valor puede ser “primo” o “no primo”.
El programa ha de mostrar los elementos de la pila y de la cola por separado, indicar la proporción de primos en el total, y por último mostrar los elementos de la lista definitiva.
Para realizar el programa se han de implementar todas las funciones necesarias de manipulación de pilas, colas y listas.

Observaciones:

- El programa principal se almacenará en el archivo principal.c.
- Las funciones de manejo de listas se almacenarán en un fichero llamado lista.c, las de colas en un fichero llamado cola.c, y las de pilas en un fichero denominado pila.c. Estos ficheros tendrán su correspondiente fichero .h.
- El proceso de compilación y enlace será controlado por un fichero Makefile creado para tal efecto.
- Todas las funciones deben tener una cabecera informativa.

Duración de la práctica
Semanas del 7 al 11 de Mayo y del 14 al 18 de Mayo.