Facultad de Informática – Universidad Complutense

Fundamentos de la programación – Grupos A, B y DG

Curso 2012–2013

Examen final – 6 de junio de 2013

Tiempo disponible: 3 horas

Se quiere calcular lo que cada cliente de una empresa gasta en sus llamadas de telefóno. Se ha de mantener una lista de hasta 300 clientes.

Para cada cliente se mantiene la siguiente información (tipo `tCliente`):

- NIF (cadena con 8 dígitos y una letra mayúscula final; p. e., 00112233A)
- Gasto telefónico total
- Lista de las llamadas realizadas (hasta 100 llamadas) (**array dinámico**)
  Esta lista se mantendrá **ordenada por fecha de llamada**

De cada llamada se guarda la siguiente información (tipo `tLlamada`):

- Fecha de la llamada (cadena con el formato AA/MM/DD; en un mismo siglo)
- Duración de la llamada (en segundos)
- Coste de la llamada (0,15 € de establecimiento + 0,08 € por minuto o fracción)

La relación de llamadas de los clientes está en un archivo de texto `llamadas.txt` que nuestro programa deberá leer. Contiene una serie de líneas, cada una con estos datos separados por un espacio: NIF del cliente, fecha de la llamada y segundos que duró la llamada. Termina con una línea con `X` como NIF. El archivo será siempre correcto (no hay que comprobar posibles errores, más allá de que exista el archivo).

Ejemplo de archivo `llamadas.txt`:

```
00112233A 13/05/02 232
95637245G 13/05/02 112
00112233A 13/05/01 128
00112233A 13/05/02 657
12345678K 13/05/01 94
...
X
```

Desarrolla un programa en C++ que incluya los siguientes subprogramas:

- ✓ `cargaLlamadas()`: carga la información del archivo `llamadas.txt`; para cada llamada localiza el cliente con el NIF en la lista (si no existe, lo crea) y añade una llamada en su lista de llamadas con la fecha y duración; calcula el coste de la llamada y actualiza el gasto total del cliente. Devuelve la lista con los datos leídos, o vacía si el archivo no existe. Usa los dos siguientes...

- ✓ `buscaCliente()`: dada la lista de clientes y un NIF, devuelve la posición (índice) del elemento de la lista de clientes con ese NIF; si no hay ningún cliente con ese NIF, devuelve -1.
- ✓ `insertaLlamada()`: dada una lista de llamadas y un registro de llamada, añade la llamada en la lista. Recuerda: la lista debe seguir ordenada por fecha.

  En ningún momento se debe hacer uso de algoritmo de ordenación alguno.
- ✓ `muestraLlamada()`: acepta un registro de llamada y muestra su información con el siguiente formato:

```
13/05/02  232 seg. 0.47 Eur
```

- ✓ `muestraCliente()`: muestra la información de un cliente con este formato:

```
00112233A
    13/05/01  128 seg. 0.39 Eur
    13/05/02  232 seg. 0.47 Eur
    13/05/02   94 seg. 0.31 Eur
     ...
Coste total de las llamadas: 4.55 Eur
```

- ✓ `muestraClientes()`: muestra la lista de clientes.
- ✓ `descuento()`: aplica a las llamadas de más de 3 minutos un 10% de descuento, actualizando su coste y el gasto total de los clientes.

El programa principal comenzará cargando la información del archivo en la lista de clientes. A continuación mostrará la lista de clientes. A continuación aplicará el descuento a las llamadas, para terminar mostrando de nuevo la lista de clientes.

Se valorará la estructuración del código, así como el uso adecuado de los esquemas de recorrido y búsqueda, de la comunicación entre subprogramas y de la memoria.

Puntuación:

Estructuras de datos: 1,5 puntos
`cargaLlamadas()`: 1,75 puntos
`buscaCliente()`: 1 punto
`insertaLlamada()`: 1 punto
`muestraLlamada()`: 0,25 puntos
`muestraCliente()`: 0,5 punto
`muestraClientes()`: 0,25 punto
`descuento()`: 1,25 punto
Programa principal: 1 punto
Programa modular: 1 punto (*entrega un ZIP con todos los archivos de código*)
Estilo y comentarios: 0,5 puntos

Deberás entregar el código del programa a través del Campus Virtual. Asegúrate de entregar una versión sin errores de compilación.