Programación 2

Examen de teoría (junio 2014)

9 de junio de 2014

Instrucciones

- **Duración: 3 horas**
- El fichero del primer ejercicio debe llamarse `monoinfinito.cc`. Para el segundo problema es necesario entregar cuatro ficheros, llamados `Biblioteca.cc, Biblioteca.h, Libro.cc, Libro.h`. Pon tu DNI y tu nombre en un comentario al principio de los ficheros fuente.
- La entrega se realizará como en las prácticas, a través del servidor del DLSI (`http://pracdlsi.dlsi.ua.es`), en el enlace `Programación 2`. Puedes realizar varias entregas, aunque sólo se corregirá la última.

Problemas

1. **(5.5 puntos)** Teorema del mono infinito: con suficiente tiempo, un chimpancé pulsando las teclas de una máquina de escribir al azar podría escribir una obra de Shakespeare (para una obra de John Grisham se estima una semana).

   Se debe desarrollar un programa que simule el comportamiento del mono infinito. El programa generará secuencias de letras minúsculas al azar para formar palabras de una longitud determinada, comprobando en un diccionario si dichas palabras existen o no. Para ello debe recibir como entrada dos números[1], que indican el rango de longitud de las palabras que se deben generar. Por ejemplo, si ejecutamos el programa así:

   ```
   ./monoinfinito 3 5
   ```

   se generarán palabras aleatorias de 3, 4 y 5 letras. Para cada tamaño se deben generar 10.000 palabras (en el ejemplo anterior, tendríamos 10.000 de tamaño 3, 10.000 de tamaño 4 y 10.000 de tamaño 5). Debe comprobarse que el primer número[2] pasado como parámetro (inicio del rango) es menor o igual que el segundo (fin del rango), y que ambos son mayores que 1 y menores o iguales a 10.

   Para generar las secuencias de letras aleatorias, puedes utilizar la función `rand()` de la librería `cstdlib`:

   ```
   char c = rand()%26+'a'; // Genera una letra minúscula al azar entre 'a' y 'z'
   ```

   Para cada una de las palabras generadas, deberá comprobarse su existencia en un diccionario, que estará almacenado en un fichero de texto llamado `diccionario.txt` y contendrá una palabra por línea. Puedes descargar un diccionario de prueba desde `http://www.dlsi.ua.es/~pertusa/exam/diccionario.txt`. Si no se puede abrir el fichero, el programa no debe generar las palabras aleatorias.

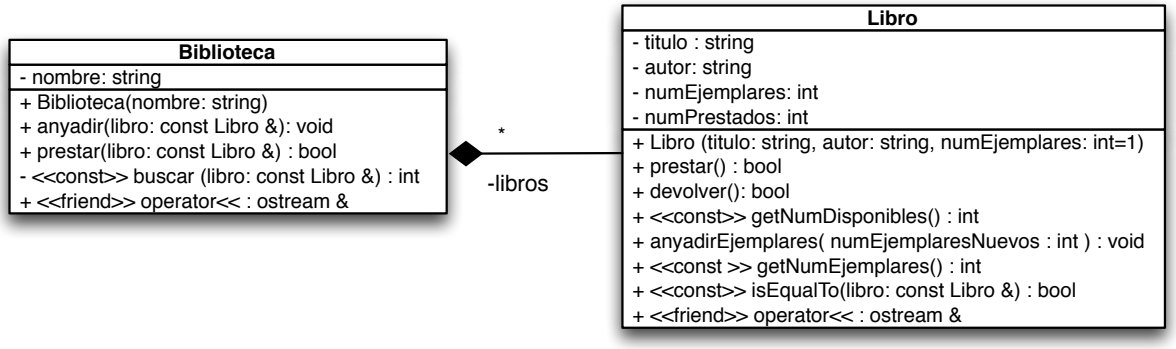
   Por eficiencia, se debe recorrer una sola vez el fichero y almacenar en un vector de strings sólo las palabras del diccionario que tengan un número de letras entre el mínimo y máximo tamaño. Finalmente, el programa mostrará por pantalla las palabras aleatorias encontradas en el diccionario. Por ejemplo[3]:

   ```
   - Longitud 3:
   por
   una
   - Longitud 4:
   bebe
   casa
   pera
   - Longitud 5:
   cardo
   salsa
   ```

[1] En caso contrario debe producirse un error de sintaxis. Se asume que si hay dos argumentos, estos siempre serán números.
[2] Puedes utilizar la función `atoi` de la librería `cstdlib` para transformar una cadena de caracteres a entero.
[3] En tu ejecución no te saldrá lo mismo, ya que las palabras son aleatorias. Puede haber palabras aleatorias repetidas.

2. **(4.5 puntos)**

Queremos hacer un programa que nos permita gestionar los préstamos de una biblioteca. A continuación se muestra el diagrama de clases:



Un libro tiene un autor, un título, el número de ejemplares del mismo, y la cantidad de ellos que están prestados. El método `prestar` incrementa el valor de `numPrestados`, y `devolver` lo decrementa. En ambos casos hay que comprobar que la operación sea posible en función del número de ejemplares y de los que están prestados, y si no es así debe mostrarse un error y devolver `false`. El método `anyadirEjemplares` añade `numEjemplaresNuevos` ejemplares del libro. El método `isEqualTo` devuelve `true` si dos libros tienen el mismo autor y título. El operador salida imprime el número de libros disponibles seguido del título y el autor entre paréntesis, tal como se puede ver en la salida de ejemplo.

En la clase `Biblioteca`, el método `buscar` devuelve la posición de un libro en el vector o `-1` si no está. El método `anyadir` añade un libro a la biblioteca, y si ya estaba incrementa el número de ejemplares de ese libro. El método `prestar` se usa para prestar un libro, y devuelve `true` si se puede. El operador salida imprime el nombre de la biblioteca y a continuación todos los libros que pueden prestarse.

Dado el siguiente fichero `main.cc`:

```
#include "Biblioteca.h"

int main()
{
  Biblioteca b("General");
  Libro l1("La guerra del fin del mundo","Mario Vargas Llosa");
  Libro l2("La tapadera","John Grisham", 4);
  Libro l3("A game of thrones","George R.R. Martin", 2);

  b.anyadir(l1);
  b.anyadir(l2);
  b.anyadir(l3);

  b.prestar(l1);
  b.prestar(l2);

  cout << b << endl;
}
```

...el programa debería imprimir lo siguiente:

```
General
--------
Libros disponibles:
3: La tapadera ( John Grisham )
2: A game of thrones ( George R.R. Martin )
```

Ayuda: Puedes descargar un `makefile` para compilar en `http://www.dlsi.ua.es/~pertusa/exam/makefile`