Guion de prácticas

MPALABRADOS (move)

Marzo 2020

UNIVERSIDAD DE GRANADA

ETSIIT Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

DECSAI

Metodología de la Programación

DGIM

Curso 2019/2020

Índice

1. Descripción **5**

2. Práctica a entregar **5**
2.1. Configuración de la práctica 7
2.2. Ejecución de prueba 8
2.3. Validación de la práctica 9
2.4. Entrega de la práctica 9

3. Código de la práctica **10**
3.1. Normalizar palabras con el alfabeto permitido 10

1. Descripción

En esta práctica se va a desarrollar la siguiente capa de la arquitectura, según el plan de trabajo fiijado en el guión de la Práctica 1. En este caso, se va a implementar la clase **move**, según la documentación sobre la misma contenida en el fichero **move.h**.

2. Práctica a entregar

Se deberá duplicar el proyecto de Netbeans de la práctica anterior y realizar los siguientes cambios (en todos ellos aparece la marca **@warning** avisando de las tareas de implementación que están pendientes).

- **move.h**
  Añadirlo al proyecto recién creado.
- **move.cpp**
  Completar la implementación y añadirlo al proyecto.
- Carpeta **tests**
  Eliminar los ficheros de test anteriores y susituirlos por los que están en Prado.
- **main.cpp**
  Sustituir al anterior y completar el código para realizar el siguiente programa.

  1. El main() recibe como parámetro obligatorio `"-l <ID>"` y como parámetros opcionales `"-i <file>"` y `"-r <random>"`, en cualquier orden entre los tres. Si se especifica `"-i"` se leen los datos desde ese fichero, si no, se leen desde el teclado. Si se especifica `"-r"` se define el aleatorio con el número indicado, si no, no se define aleatorio.
  2. Crear una instancia de la clase **Language** con el anterior ID y mostrar el conjunto de caracteres permitido para ese lenguaje.
  3. Crear una instancia de la clase **Bag**, inicializar el generador de números aleatorios con el número aleatorio anterior, si es que se ha indicado, y definir su contenido en base al lenguaje que se ha declarado anteriormente.
  4. Crear una instancia de la clase **Player** y llenarla por completo con caracteres de la bolsa. Este objeto player deberá estar siempre ordenado de la A a la Z.
  5. Repetir la siguiente secuencia hasta que se lea un movimiento con la palabra `"@"`

     a) Usar el método **read(...)** para leer un movimiento (desde teclado o desde el fichero de entrada, según el parámetro `"-i"`). Los valores para isHorizontal, row y column se leen pero se van a ignorar en el resto del programa, pues sólo se

usará letters. En las anteriores prácticas se han usado palabras con letras controladas, pues, al fin y al cabo, todas provenían de bag, las cuales provienen del diccionario, y todas siguen el mismo patrón del juego scrabble: las letras son mayúsculas, no contienen tildes ni diéresis, aunque algunos caracteres internacionales están soportados, como la Ñ. Aunque no se ha usado hasta ahora, al incluir **language.h** en cualquier proyecto, también se incluye la definición de esta constante:

```
static const std::string ALPHABET=toISO(" ABCDEFGHIJKLMNOPQRSTUVWXYZÑ");
```

Esto quiere decir que cualquier palabra que se consulte en el diccionario debe estar representada con caracteres de ese `ALPHABET` Esta es la primera vez que exponemos el programa a leer datos externos no controlados, por lo que habría que transformar cada palabra de juego que se lee, a una palabra expresada en ese alfabeto. Y eso es lo que hace la función **normalizeWord(...)**

b) Si la palabra leída es válida para el valor de **Player** (al menos ha de tener dos caracteres) entonces se anota la palabra, se calcula la puntuación de la palabra según el diccionario y se anota, se eliminan las letras de player, se sacan nuevas letras de bag para rellenar player, y se sigue jugando. Ya nos estamos acercando al ciclo de juego de la práctica final.

c) Si la palabra leída no es compatible con el valor de player, se desecha y se lee el siguiente movimiento.

6. Terminar con la llamada a **HallOfFame** para visualizar los resultados.
7. Si en cualquier momento se presenta un error en los argumentos, en la apertura de ficheros o en la lectura de datos del fichero, se debe usar la función **errorBreak(...)** para notificar el error y parar el programa.

2.1. Configuración de la práctica

La misma que en la práctica anterior pero añadiendo una carpeta **data** en la que almacenar los ficheros de datos ***.data**.

```
.
|-- build
|-- data
|   `-- *.data
|-- dist

|-- doc
|   `-- documentation.doxy
|-- include
|   |-- bag.h
|   |-- move.h
|   `-- player.h
|-- languages
|   |-- EN.*
|   |-- ES.*
|   `-- FR.*
|-- Makefile
|-- nbproject
|-- scripts
|   `-- *.sh
|-- src
|   |-- bag.cpp
|   |-- main.cpp
|   |-- move.cpp
|   `-- player.cpp
|-- tests
|   `-- *.test
`-- zip
```

2.2. Ejecución de prueba

Se resaltan en rojo las entradas desde teclado.

```
dist/Debug/GNU-Linux/mp1920practica3 -l es -r 16
Opening tree file ./languages/ES.tree
Trying to read 48428 words
OK 48428 words read
Opening ./languages/ES.scrabble
OK 25 Scrabble's letter read
LANGUAGE: es
ALLOWED LETTERS: LTNRUISOAEGDBMPCFVYHQJÑXZ
SEED: 16
BAG (95) : CCATMAVARODGDAPNEEAQZCINOUPOIEOGDBARBEOTOATAUED
LASAEOHNORXAEEUOMSISJYILLIENTFERISNDÑUSELUHARESC
PLAYER: AACCMTV
h 0 0 prueba

READ: H 0 0 PRUEBA INVALID!
PLAYER: AACCMTV
h 0 0 macat

READ: H 0 0 MACAT NOT REGISTERED!

PLAYER: ACDGORV
h 0 0 gorda

READ: H 0 0 GORDA FOUND! 7 points

PLAYER: ACDENPV
h 0 0 pena

READ: H 0 0 PENA FOUND! 6 points

PLAYER: ACDEQVZ
h 0 0 @

%%%OUTPUT
LANGUAGE: ES ID: 16
BAG (74): CINOUPOIEOGDBARBEOTOATAUEDLASAEOHNORXAEEUO
MSISJYILLIENTFERISNDÑUSELUHARESC
PLAYER (7): ACDEQVZ
2 words and 13 points
GORDA - PENA -
```

2.3. Validación de la práctica

Se debe ejecutar la script **doTests.sh** y comprobar que los resultados que aparecen por pantalla coinciden con lo indicado en cada caso de validación.

```
Generating fresh binaries
"make" -f nbproject/Makefile-Debug.mk QMAKE= SUBPROJECTS= .build-conf
make[1]: se entra en el directorio '/home/lcv/Dropbox/Teaching/MP/NetBeans/MP1920Practica3'
"make"  -f nbproject/Makefile-Debug.mk dist/Debug/GNU-Linux/mp1920practica3
make[2]: se entra en el directorio '/home/lcv/Dropbox/Teaching/MP/NetBeans/MP1920Practica3'
make[2]: 'dist/Debug/GNU-Linux/mp1920practica3' está actualizado.
make[2]: se sale del directorio '/home/lcv/Dropbox/Teaching/MP/NetBeans/MP1920Practica3'
make[1]: se sale del directorio '/home/lcv/Dropbox/Teaching/MP/NetBeans/MP1920Practica3'

[  OK  ] Test #1 [tests//CALL_ERROR.test] ( mp1920practica3  -r 2020 -i data/EN_2020_177.data)
[  OK  ] Test #2 [tests//DATA_ERROR.test] ( mp1920practica3  -l ES -r 100 -i data/ES_100_ERROR.data)
[  OK  ] Test #3 [tests//EN_2020_177_F.test] ( mp1920practica3  -r 2020 -l EN -i data/EN_2020_177.data)
[  OK  ] Test #4 [tests//EN_3_175_F.test] ( mp1920practica3  -l EN -r 3 -i data/EN_3_175.data)
[  OK  ] Test #5 [tests//EN_3_175_K.test] ( mp1920practica3  -l EN -r 3 < data/EN_3_175.data)
[  OK  ] Test #6 [tests//ES_2020_EMPTY.test] ( mp1920practica3  -r 2020 -l ES < data/FR_6_166.data)
[  OK  ] Test #7 [tests//ES_2020_K.test] ( mp1920practica3  -r 2020 -l ES < data/ES_2020_155.data)
[  OK  ] Test #8 [tests//OPEN_ERROR.test] ( mp1920practica3  -l ES -r 100 -i data/OPEN_ERROR.data)
```

2.4. Entrega de la práctica

Se deberá ejecutar la script **doZipProject.sh** y subir a Prado, en las fechas que se indican en la temporización de la asignatura, el zip resultante, que está almacenado en la carpeta **.zip**/ del proyecto de Netbeans y siempre se llama **MPPractica.zip**.

3. Código de la práctica

3.1. Normalizar palabras con el alfabeto permitido

Siempre que se lee una cadena desde el exterior del juego, se debe normalizar, lo que implica un proceso de dos pasos. Primero usar sólo el alfabeto permitido. Segundo pasar la codificación a ISO8859. Ambos pasos los hace la siguiente función. En caso de que se use un carácter que no esté en el alfabeto, se traducirá por el carácter _

```
/**
 * @brief Translate any word into valid characteres only
 * @param word
 * @return ISO Word
 */
std::string normalizeWord(const std::string & word);
```

Original	Normalizada
CAlló	CALLO
sueño	SUEÑO
cigüeña	CIGUEÑA
CO$A	CO_A