FACULTADE DE INFORMÁTICA
Programación II

Ejecución Script

En cada práctica, el programa principal implementará un bucle que procese una a una las distintas peticiones que haya que realizar. Para simplificar tanto el desarrollo como la prueba de la práctica, en vez de solicitar por pantalla los datos correspondientes a cada petición, el programa deberá **funcionar en modo *batch*** (i.e. sin interactividad con el usuario) utilizando para ello como entrada un fichero en el que se habrá almacenado previamente la secuencia de peticiones a ejecutar. Para cada iteración del bucle, el programa leerá del fichero una nueva petición y, seguidamente, dicha petición será procesada.

Para poder probar distintos ficheros, el nombre (y ruta, si es precisa) del fichero de entrada NO estará "*hardcoded*" en el código del programa, sino que habrá de pasarse como parámetro al ejecutarlo. Supongamos que, por ejemplo, el fichero de entrada es */tmp/input.txt*, al ejecutar el programa habrá que indicar (asumiendo que el ejecutable es `main`):

```
./main /tmp/input.txt
```

En caso de que no se especifique ningún fichero de entrada, el programa empleará por defecto el fichero indicado en el fichero `read.pas` proporcionado con cada práctica. En ese caso la llamada sería:

```
./main
```

Las funciones `ParamCount` y `ParamStr`, descritas en la documentación del lenguaje FreePascal, serán de utilidad para la implementación de esta nueva funcionalidad.

Al realizar la lectura de la entrada de este modo, el uso de un script permite probar distintos ficheros. Así, junto con la documentación de la práctica se proporciona un fichero comprimido denominado `script.zip` que contiene un script (`script.sh`) que permite probar tanto el `TestUnit.pas` como el `main.pas`. En este fichero comprimido se incluyen 2 carpetas (`script_test_unit` y `script_minimos`) que contienen todos los ficheros de entrada necesarios y la salida esperada (ficheros terminados en ref). Para la correcta ejecución de este script se deben ubicar en un directorio, bien de los equipos de los laboratorios de docencia o bien de los servidores de referencia, todos los ficheros `.pas` con el código de la práctica (`TestUnit.pas`, `main.pas`, `StaticList.pas` y `DynamicList.pas`) y

descomprimir el fichero `script.zip`. Posteriormente, desde consola y tras ubicarse en dicho directorio, hay que darle a `script.sh` permisos de ejecución:

```
chmod u+x script.sh
```

y, finalmente, ejecutar dicho *script.* Para ello existen dos alternativas:

1) `./script.sh` equivalente a `./script.sh -p main`
2) `./script -p TestUnit`

En el primer caso se ejecuta el programa principal con los distintos ficheros de entrada que se encuentran en el directorio `script_minimos` y, en este mismo directorio, se crearán dos ficheros (por ejemplo, `new_DynamicList.txt` y `new_StaticList.txt`) que muestran la salida tras ejecutarse el programa principal con cada uno de los ficheros de prueba (`new.txt, vote.txt, illegalize1.txt e illegalize2.txt`) con cada implementación del TAD (estática y dinámica). En la segunda alternativa, se ejecutará el `TestUnit.pas`, de nuevo con ambas listas, y se comprobará que la salida es la esperada (igual a `ref.txt`).

Obsérvese que el propio script se encarga ya de modificar los ficheros TestUnit.pas y main.pas para poner el nombre de la unit correspondiente, compilar y, a continuación, ejecutar. Si se produce algún error de compilación para alguna de las units, el fichero resultado correspondiente sólo contendrá los nombres de los distintos ficheros que se intentan probar.

En ambas opciones, script dispone de la opción `-v` que, además de indicar la corrección de la salida, muestra por pantalla las líneas en las que la salida del programa difiere de la salida de referencia:

```
./script.sh -v
```