Programación de Sistemas de Telecomunicación
Entrega de la Práctica 1

GSyC

6 de octubre de 2015

Instrucciones

- Antes de nada, crea una carpeta con tu nombre de usuario. Por ejemplo, si tu nombre de usuario es `fgomez`, deberás ejecutar desde un terminal los siguientes comandos:

  ```
  cd; mkdir fgomez
  ```

- Dentro de tu carpeta de usuario crea una carpeta con el nombre de cada ejercicio:

  ```
  cd; cd fgomez
  mkdir ejercicio1
  mkdir ejercicio2
  mkdir ejercicio3
  ```

- Todos los programas de ejemplo que menciona el enunciado están en la carpeta `/opt/pst`
  Abre esa carpeta con el navegador para acceder a esos materiales. Copia a tu carpeta los ficheros de `/opt/pst` según los vayas necesitando.
- Los materiales de la asignatura (transparencias y enunciados) también están disponibles en `/opt/pst`

Ejercicio 1

Modifica el paquete `Word_Lists` para que ofrezca en su especificación la siguiente función:

```
function Length_Average (List: Word_List_Type) return Natural;
-- Devuelve la media aritmética de las longitudes de las palabras
-- almacenadas en List.
```

La media aritmética de n números, $x_1, x_2, ..., x_n$ es $\frac{x_1+x_2+...+x_n}{n}$

El paquete modificado deberá funcionar con el programa principal `prueba_length_average.adb` que te proporcionamos, debiendo producir la siguiente salida al ejecutar dicho programa:

```
./prueba_length_average
 5
```

Norma de entrega: En la carpeta del `ejercicio1` deja los nuevos ficheros `word_lists.ads` y `word_lists.adb`.

Ejercicio 2

Escribe un programa principal en un fichero llamado `trim.adb` que reciba uno o dos argumentos en la línea de comandos:

- Si hay dos argumentos, el primero sólo puede ser `-h`, y el segundo será el nombre de un fichero.
- Si sólo hay un argumento, éste será el nombre de un fichero.

Si el programa se ejecutó con un sólo argumento, devolverá en la salida estándar las mismas líneas de texto del fichero que se ha pasado como argumento en la línea de comandos.

Pero si el programa recibió dos argumentos, el programa devolverá en la salida estándar las mismas líneas de texto del fichero que se ha pasado como segundo argumento, pero sin los espacios en blanco que existan al comienzo de cada línea.

Se proporciona el fichero `quijote.txt`. A continuación se muestra la salida que debería producir el programa cuando se ejecuta pasándole como argumento el nombre de este fichero:

```
./trim quijote.txt
   Este libro no tiene cosa digna que no corresponda a su original; en
 testimonio de lo haber correcto, di esta fee. En el Colegio de la Madre de
     Dios de los Teólogos de la Universidad de Alcalá, en primero de diciembre
  de 1604 años.


./trim -h quijote.txt
Este libro no tiene cosa digna que no corresponda a su original; en
testimonio de lo haber correcto, di esta fee. En el Colegio de la Madre de
Dios de los Teólogos de la Universidad de Alcalá, en primero de diciembre
de 1604 años.
```

Norma de entrega: En la carpeta del `ejercicio2` deja el fichero `trim.adb`.

Ejercicio 3

Modifica el paquete `Word_Lists` para que ofrezca en su especificación la siguiente función:

```
procedure Sorted_Add_Word(List: in out Word_List_Type; Word: String);
-- Hace lo mismo que Add_Word, pero cuando inserta la palabra
-- lo hace en el lugar que le corresponde alfabéticamente
```

Los operadores < y > se pueden utilizar para comparar tanto `String` como `Unbounded_String`.

El paquete `Word_Lists` modificado deberá funcionar con el programa principal `prueba_sorted_add_word.adb` que te proporcionamos, debiendo producir la siguiente salida al ejecutar dicho programa:

```
./prueba_sorted_add_word
|abecedario| - 1
|bonito| - 1
|cachorro| - 1
|pavo| - 1
|ramona| - 1
|saturno| - 2
|vivo| - 1
|xen| - 1
```

Norma de entrega: En la carpeta del `ejercicio3` deja los nuevos ficheros `word_lists.ads` y `word_lists.adb`.