From Prog1 - InCo

Laboratorio: PrimeraTarea2009

Primera Tarea. Programación 1

Para obtener una versión apropiada para impresión, siga el vínculo **imprimir**.

En esta página... (ocultar)

1. Introducción

Este documento presenta el problema que deberá resolverse para la aprobación de la primera tarea del laboratorio del curso 2009.

Se presenta información acerca de: normas, recursos, plazos, una presentación general del problema, las especificaciones del mismo, ejemplos de ejecución y la forma de entrega.

---

2. Información general

El estudiante que no respete alguna de las consideraciones que siguen, corre el riesgo de que su trabajo sea invalidado, con la consiguiente pérdida del curso.

Compilador

Todos los programas deben ser compatibles con el compilador del curso (**Free Pascal**). No se aceptarán como válidas aquellas tareas que pudieran funcionar con algún otro compilador Pascal, pero no funcionen con Free Pascal, versión 2.2.2 para windows.

Grupos

Esta primera tarea se podrá realizar en grupos de 1 o 2 estudiantes.
Para todas las tareas rige el **Reglamento del Instituto de Computación ante Instancias de No Individualidad en los Laboratorios**. A continuación se adjunta un fragmento del mismo; ante cualquier duda se recomienda leer el documento completo (**http://www.fing.edu.uy/inco/cursos/prog1/pm/field.php/Laboratorio/NoIndividualidad**)

> Los laboratorios deben ser realizados únicamente por los integrantes del grupo establecido La realización de los laboratorios es estrictamente individual, sea a nivel unipersonal en el primer caso, o del grupo establecido en el segundo. Se entiende que compartir total o parcialmente cualquier actividad del laboratorio atenta contra la integridad del estudiante universitario y de su formación, y por lo tanto constituye una falta grave. Específicamente no es posible compartir por ninguna vía entre integrantes de grupos distintos las tareas de codificación, digitación, compilación, depuración y documentación de los programas u objetos (o entregas) del laboratorio. Además de que no se pueden compartir actividades del laboratorio, no se pueden compartir los productos de las mismas. Cada grupo es responsable de su trabajo de laboratorio y de que el mismo sea individual, independientemente de las causas que pudiesen originar la no individualidad. A modo de ejemplo y sin ser exhaustivos: utilización de código realizado en cursos anteriores (por otros estudiantes) u otros cursos, perder el código, olvidarse del código en lugares accesibles a otros estudiantes, prestar el código o dejar que el mismo sea copiado por otros estudiantes, dejar la terminal con el usuario abierto al retirarse, enviarse código por mail, utilizar código suministrado por terceros, etc. Asimismo se prohíbe el envío de código al grupo de noticias del curso, dado que el mismo será considerado como una forma de compartir código y será sancionado de la manera más severa posible.

Forma de entrega

Las entregas se realizarán por la *web*. Para ello se deberá acceder a un sitio destinado a tal fin y seguir los pasos que se explicarán en la página correspondiente. Esta página estará disponible cuando comience el plazo de entrega.

Fecha de Entrega

Se debe entregar desde el **5/10** hasta el **9/10** a la medianoche (hora 12).

---

3. Contador de Palabras

En este laboratorio se debe implementar un programa que permita calcular la cantidad de palabras de un texto que cumplen con ciertas condiciones. Las condiciones son ingresadas antes de ingresar el texto.

3.1 Descripción

Se desea escribir un programa que lea un texto y que despliegue la cantidad de palabras que contiene el texto que cumplen con una condición.

La condición es especificada por el usuario y puede estar compuesta por una combinación de las siguientes condiciones:

- La palabra termina con una letra dada.
- La palabra comienza con una letra dada.
- La palabra tiene una cantidad de caracteres dada.
- La palabra contiene una letra dada.

El usuario puede especificar si las palabras a considerar deben cumplir con *alguna* o con *todas* las condiciones especificadas.

Más adelante se describen los detalles de las condiciones y el texto.

3.2 Datos de entrada

Los datos son leídos desde la entrada estándar. Están consituidos por dos partes:

1. Especificación de las condiciones que deben cumplir las palabras que deben ser consideradas en el conteo.
2. El texto a ser analizado.

3.3 Especificación de las condiciones

Las condiciones se especifican codificadas como una secuencia de **líneas consecutivas**, cada línea contiene una condición simple. El usuario puede ingresar o no todas las condiciones, pero al menos especificará una de ellas. En caso de ingresar más de una condición del mismo tipo, sólo se tomará en cuenta la última ingresada.

Para terminar la entrada de condiciones, se ingresará una marca especial que lo indique.

A continuación se especifica la codificación de cada una de las condiciones:

Terminación de palabra

Para indicar la condición de que las palabras a contar terminen con una letra dada se incluye una línea de la forma

```
Fx
```

donde *x* es la letra con la cuál la palabra debe terminar.

Comienzo de palabra

Para indicar la condición de que las palabras a contar comiencen con una letra dada se incluye una línea de la forma:

```
Cx
```

donde *x* es la letra con la cuál la palabra debe comenzar.

Letra contenida en palabra

Para indicar la condición de que las palabras a contar contengan una letra dada se incluye una línea de la forma:

```
Ex
```

donde x es la letra que la palabra debe contener. La letra *x* puede estar en cualquier parte de la palabra,

incluso al principio o al final. También puede aparecer más de una vez.

Largo de palabra

Para indicar la condición de que las palabras a contar tengan un largo determinado se incluye una línea de la forma:

```
Ln
```

donde *n* es un número positivo que indica la cantidad de caracteres que debe contener la palabra.

Todas o Algunas de las condiciones

Para indicar que las palabras a contar deben cumplir con todas las condiciones especificadas se incluye una línea como la siguiente:

```
A
```

En caso de no aparecer una línea como la anterior se cuentan las palabras que cumplen con al menos una de las condiciones que se especificaron.

Fin de parte de condiciones

Luego de las condiciones, para indicar el fin de esta parte de la entrada, se ingresa una línea como la siguiente:

```
Q
```

3.4 El texto a ser analizado

Seguidamente de la marca de fin de condiciones (`Q`) se ingresa el texto que debe ser analizado en función de las condiciones especificadas. El mismo consta de una cantidad no pre-establecida de palabras separadas por al menos un espacio en blanco. Para marcar el fin del texto se ingresa al menos un espacio y un punto. Se hace notar que todo el texto viene dado en una sola línea (no se ingresan saltos de línea).

Asuma que las entradas vienen sin errores. Por lo que *no es necesario verificar* la buena formación ni de las condiciones ni del texto. Ante cualquier duda lea atentamente las sesiones de ejecución y utilice el "programa modelo".

3.5 La Salida

Se debe desplegar en la salida estándar el siguiente mensaje:

```
La cantidad de palabras que cumplen con la condicion es: n
```

donde n es el valor que se computó al analizar el texto.

4. Se pide

Escribir un programa Pascal que realice lo que se describe más arriba.

Se debe utilizar el lenguaje Pascal tal como fue dado en el curso (ver **FreePascal Y Pascal Estándar**)

> El programa entregado debe seguir el comportamiento que se presenta en esta letra. **El formato de la entrada y la salida debe ser EXACTAMENTE IGUAL. De lo contrario la corrección automática de la tarea fallará.**

Por ejemplo, se considera que es una solución incorrecta si imprime mensajes del tipo:

- `Bienvenido al progrograma para contar palabaras`",
- `Por favor ingrese las condiciones: ` ,
- "`Chau.`"
- o cualquier variante que no respete lo que se establece en la letra.

Tampoco debe ponerse un `read` o `readln` al final para pausar el programa.

Se puede utilizar todo lo visto en las clases teóricas y prácticas. Para realizar la tarea alcanza con lo visto en teórico hasta la clase 4 inclusive. Esto incluye los siguientes conceptos:

- Tipos de datos:
  - `Integer`
  - `Boolean`
  - `Char`
- Instrucciones:
  - Asignación
  - Entrada y Salida(`read`, `write`, `WriteLn`, `ReadLn`)
  - Secuencia
  - Selección (`if`, `case`)
  - Repetición (`for`, `while`, `repeat`)

En esta parte **no** se deben utilizar subprogramas (procedimientos y funciones)

Para la corrección de las tareas, se compilarán con la versión 2.2.2 para windows. La compilación y la ejecución se realizarán en línea de comandos. El comando de compilación se invocará de la siguiente manera:

```
fpc -Co -Cr programa.pas
```

Si trabaja con el IDE, asegúrese de configurarlo para que compile de la misma manera que el comando anterior (habilitación de *range checking* e *Integer Overflow Checking*)

> No está permitido utilizar facilidades de Free Pascal que no forman parte del estándar y no se dan en el curso. Así por ejemplo, no se pueden utilizar ninguna de las palabras siguientes: `uses`, `crlscr`, `gotoxy`, `crt`, `readkey`, `longint`, `string`, `break`, etcétera.

En esta tarea como en todos los problemas de este curso, se valorará además de la lógica correcta, la utilización de un buen estilo de programación de acuerdo a los criterios impartidos en el curso. De esta manera, se hará énfasis en buenas prácticas de programación que lleven a un código legible, bien documentado y mantenible, tales como:

- indentación adecuada
- utilización correcta y apropiada de las estructuras de control
- código claro y legible
- algoritmos razonablemente eficientes
- utilización de comentarios que documenten y complementen el código
- utilización de constantes simbólicas
- nombres mnemotécnicos para variables, constantes, etcétera.

---

5. Ejemplos de sesión

```
Cp
L14
Ee
A
Q
este es un texto de prueba para mostrar el funcionamiento termina con punto .
La cantidad de palabras que cumplen con la condicion es: 0
```

```
Cp
L14
Ee
Q
este es un texto de prueba para mostrar el funcionamiento termina con punto .
La cantidad de palabras que cumplen con la condicion es: 10
```

```
Cc
L14
Ee
Q
este es un texto de prueba para mostrar el funcionamiento termina con punto .
La cantidad de palabras que cumplen con la condicion es: 9
```

```
Ct
L7
Ee
Q
este es un texto de prueba para mostrar el funcionamiento termina con punto .
La cantidad de palabras que cumplen con la condicion es: 9
```

```
Ct
L7
Ee
A
Q
este es un texto de prueba para mostrar el funcionamiento termina con punto .
La cantidad de palabras que cumplen con la condicion es: 1
```

6. Programas auxiliares

A continuación se presentan dos programas para ser utilizados como modelos. Ambos resuelven problemas incluídos en este primer obligatorio..

6.1 CuentaPalabras

Se trata de un programa que lee un texto de la entrada estándar y despliega en la salida estándar la cantidad de palabras leídas y el largo de la palabra más larga.

```
program CuentaPalabras(input,output);
const { declaracion de constantes}
   centinela = '.';   { caracter de fin de texto }
   espacio   = ' ';   { espacio                  }
var   { declaracion de variables }
   caracter : char;         { variable para leer caracter }
   contador : integer;      { contador de las palabras    }
   maximo_palabra,          { largo maximo de palabra     }
   largo_palabra : integer; { largo de palabra            }

begin { programa principal }
   { inicializacion }
   contador:= 0;
   maximo_palabra:= 0;

   { saltear espacios }
   repeat
      Read(caracter)
   until caracter <> espacio;

   { Ciclo principal
     en cada iteracion se trata completamente una palabra
   }
   while caracter <> centinela do begin
      { cuando ingresa aqui tiene cargado en caracter
        el primer caracter de la palabra
      }
      { leer palabra }
      largo_palabra:= 0;
      repeat
         largo_palabra:= largo_palabra + 1;
         Read(caracter)
      until caracter = espacio;
      { actualizar contador y maximo }
      contador:= contador + 1;
      if largo_palabra > maximo_palabra then
         maximo_palabra:= largo_palabra;
      { avanzar al primer caracter de la siguiente palabra }
      repeat
          Read(caracter)
      until caracter <> espacio;
   end; {while}
   { mostrar resultados }
   WriteLn;
   WriteLn('La cantidad de palabras es: ', contador);
```

```
    WriteLn('El maximo largo de palabra es: ', maximo_palabra);
    WriteLn
  end. {CuentaPalabras}
```

6.2 LeerCondiciones

Este programa lee de la entrada estándar las condiciones para contabilizar palabras en el texto.

```
program LeerCondiciones;
  const { declaracion de constantes }
    ESPACIO  = ' ';   { espacio en blanco              }
    FINAL    = 'F';   { codigo de final de palabra     }
    COMIENZO = 'C';   { codigo de comienzo de palabra  }
    LARGO    = 'L';   { codigo de largo de palabra     }
    CONTIENE = 'E';   { codigo de pertenencia a palabra }
    TERMINAR = 'Q';   { codigo de terminacion del test  }
  var { declaracion de variables }
    { variables que indican si el test se debe realizar
      las siguientes variables quedan en true si el
      correspondiente test aparecio en la entrada }
    test_final,              { test de ultima letra     }
    test_comienzo,           { test de primera letra    }
    test_largo,              { test de largo de palabra }
    test_contiene : boolean; { test de letra en palabra }
    { variables asociadas con los test }
    letra_final,                { ultima letra a buscar   }
    letra_comienzo   : char;    { primera letra a buscar  }
    largo_palabra    : integer; { largo requerido         }
    letra_contenida  : char;    { letra a buscar          }
    caracter : char;    { variable para leer el texto }
  begin {test}
    {inicializacion de variables }
    test_final:= false;
    test_comienzo:= false;
    test_largo:= false;
    test_contiene:= false;
    { lectura de los tests }
    repeat
      read(caracter);
      case caracter of
        FINAL    : begin
                     test_final:= true;
                     ReadLn(letra_final)
                   end;
        COMIENZO : begin
                     test_comienzo:= true;
                     ReadLn(letra_comienzo)
                   end;
        LARGO      : begin
                     test_largo:= true;
                     ReadLn(largo_palabra)
                   end;
        CONTIENE : begin
                     test_contiene:= true;
                     ReadLn(letra_contenida)
                   end
      end; { case }
    until caracter = terminar;
    { desplegar resultado del reconocimiento }
```

```
    WriteLn;
    WriteLn('Se deben verificar las siguientes condiciones:');
    if test_final then
       WriteLn('Termina con la letra: ', letra_final);
    if test_comienzo then
       WriteLn('Comienza con la letra: ', letra_comienzo);
    if test_largo then
       WriteLn('Contiene ', largo_palabra, ' caracteres');
    if test_contiene then
       WriteLn('Contiene la letra: ', letra_contenida);
end.
```
