RESOLUCIÓN DE PROBLEMAS Y ALGORITMOS

CLASE 11

Archivos de texto
para entrada y salida.

Luciano H. Tamargo
http://cs.uns.edu.ar/~lt
Depto. de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur, Bahía Blanca
2016

ARCHIVOS DE TEXTO EN PASCAL (TEXT)

- En Pascal, existe un tipo predefinido "**TEXT**" que permite trabajar con **archivos de texto**.

```
PROGRAM ejemplo;
VAR documento: TEXT;
```

- A primera vista parece que fuera lo mismo que **FILE OF char**.
- Pero en esta clase veremos que **no es así**.
- El tipo **TEXT** tiene **características propias** que pueden ser vistas como facilidades para determinados problemas.

ARCHIVOS DE TEXTO EN PASCAL (TEXT)

- Recordemos que un **tipo de dato** define los valores y las operaciones que pueden usarse sobre ellos.
  - **Valores**: **TEXT** es un **tipo** estructurado, que permite almacenar una secuencia de caracteres ASCII.
  - **Operaciones**: todas las operaciones de **FILE** y además agrega: **readln**, **writeln**, y **EOLN** (end of line).

CARACTERÍSTICAS DEL TIPOTEXT

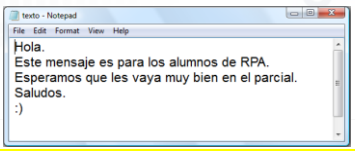
- Una característica del tipo **TEXT** es que permite el **manejo de archivos creados con otros editores de texto** (como el block de notas o notepad, el editor de lázarus, etc.).
- Observe que tiene que ser cualquier editor que genere un ***texto plano*** (*plain text*).
- Usando un archivo **TEXT** podemos ***procesar*** desde nuestro programa en Pascal cualquier archivo de texto en memoria secundaria (creado con otro programa en Pascal o no).

CARACTERÍSTICAS DEL TIPOTEXT

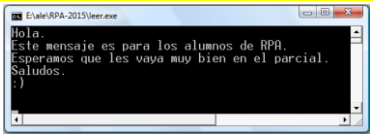
- También, desde un programa en Pascal podemos crear y escribir en un archivo **TEXT**, y luego ver el contenido del archivo con cualquier editor de texto que tengamos en nuestro dispositivo.
- **Observación interesante**: el "buffer" es un archivo **TEXT**.
- Todo lo que puede hacer con el buffer lo puede hacer en un **TEXT**.

PROBLEMA PROPUESTO

- Considere un archivo llamado "texto.txt" creado con un editor como este:



Problema: escriba un programa para abrir un archivo de texto ya existente llamado "texto.txt", y mostrar por consola su contenido.



MOSTRAR EL ARCHIVO "TEXTO.TXT"

```
PROGRAM leer; {Este programa permite leer todo
el contenido de un archivo de texto, (carácter
por carácter) y mostrarlo en pantalla}
VAR
 T: TEXT;
 elemento: char;
BEGIN
 assign(T, 'texto.txt');
 reset(T);  //abre archivo para leer
 while not eof(T) do
  BEGIN
   read(T,elemento);
   write(elemento);
  END;
 close(T);
END.
```

Aquí indico el nombre del archivo a mostrar por consola

```
E:\ale\RPA-2015\leer.exe
Hola.
Este mensaje es para los alumnos de RPA.
Esperamos que les vaya muy bien en el parcial.
Saludos.
:)
```


¿QUÉ ES ENTER?

- En las máquinas de escribir mecánicas al finalizar un renglón había que hacer dos movimientos: (1) retorno de carro (2) nueva línea

- En algunos sistemas operativos ENTER tiene asociados 2 caracteres:
  (1) ASCII 13: retorno de carro (CR: carriage return)
  (2) ASCII 10: nueva línea (LF: line feed)
- Los símbolos ASCII 13 y 10 son **caracteres de control** y al imprimirlos en pantalla **producen un efecto** en lugar de mostrar algo visible. Vea por ejemplo:

```
PROGRAM uno;
BEGIN
WRITE(CHR(65));
WRITE(CHR(66));
END .
```

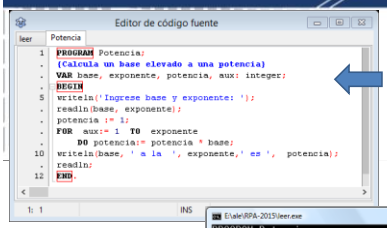
```
PROGRAM dos;
BEGIN
WRITE(CHR(65));
WRITE(CHR(10));
WRITE(CHR(66));
END .
```

```
PROGRAM tres;
BEGIN
WRITE(CHR(65));
WRITE(CHR(13));
WRITE(CHR(66));
END .
```

```
PROGRAM cuatro;
BEGIN
WRITE(CHR(65));
WRITE(CHR(13));
WRITE(CHR(10));
WRITE(CHR(66));
END .
```

¿Cómo estará implementado **writeln**?

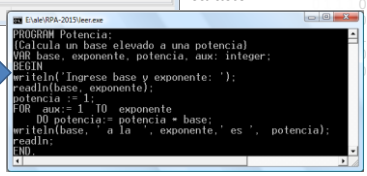
OTRO ARCHIVO DE TEXTO



Observe que el código fuente de un programa en Pascal es un archivo de texto.
Por lo tanto, al igual que "texto.txt" es posible mostrarlo por consola, carácter a carácter.

Esta es la salida del programa leer asignando el manejador T a el archivo potencia.pas.

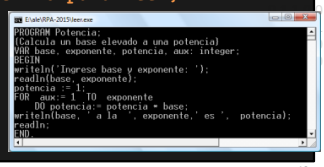
Es el primer paso para hacer un compilador.



MOSTRAR EL ARCHIVO DE TEXTO "POTENCIA.PAS"

```
PROGRAM leer; {Este programa permite leer todo
el contenido de un archivo de texto, (carácter
por carácter) y mostrarlo en pantalla }
VAR
 T: TEXT;
 elemento: char;
BEGIN
 assign(T, 'potencia.pas');
 reset(T);  {abre archivo para leer}
 while not eof(T) do
 BEGIN
  read(T,elemento);
  write(elemento);
 END;
 close(T);
END.
```

Ahora el manejador T está asociado al código fuente de potencia.pas



OPERACIONES SOBRE ARCHIVOS DE TEXTO EN PASCAL

- Además de todas las operaciones vistas sobre archivos secuenciales (**FILE**) se agregan:
- **Función predefinida:**
  - **eoln(F)** (end of line): retorna **TRUE** si se llegó al final de una línea y **FALSE** en caso contrario.
- **Procedimientos predefinidos:**
  - **writeln(T)**: escribe un fin de línea (enter) en T.
  - **readln(T)**: avanza en el texto hasta llegar a un fin de línea (enter) y se prepara para leer el carácter siguiente.

Observaciones:

- **readln(T,e)** es equivalente a **read(T,e); readln(T)**
- **writeln(T,e)** es equivalente a **write(T,e); writeln(T)**
- El "buffer" es un archivo TEXT. Todo lo que puede hacer con read y write en el buffer lo puede hacer en un TEXT.

END OF LINE (FIN DE LÍNEA) EOL

- **End-of-line (EOL)** fin de línea: **es un caracter especial, o secuencia de caracteres**, que indica el final de una línea de texto y el paso a la siguiente.
- Se le llama así porque el carácter a la derecha de **EOL** aparecerá en la línea de abajo.
- Los sistemas operativos representan **EOL** con los caracteres ASCII: **LF** (Salto de línea) o/y **CR** (Retorno de carro) pero puede ser de diferente manera. **Por ejemplo**:
  - **LF**: Multics, Unix, GNU/Linux, AIX, Xenix, MacOSX, FreeBSD, BeOS, Amiga, RISC OS.
  - **CR+LF**: MS-DOS, OS/2, Microsoft Windows, Symbian.
  - **CR**: Apple II family, Mac OS.
- En Pascal, las primitivas **predefinidas eoln, readln, writeln**, nos permiten "abstraernos" de esto, ya que conocen como está implementado **EOL** en cada sistema operativo para el cual se compilará nuestro programa.

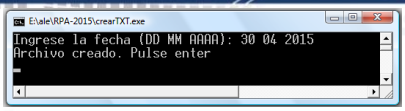
CARACTERÍSTICAS DE ARCHIVOS DE TEXTO (TEXT)

- Una característica del tipo **TEXT** es que utilizando los procedimientos predefinidos **write** o **writeln** **puedo escribir elementos de cualquier tipo simple** y son transformados a texto automáticamente.
- La **transformación a texto es automática** y funciona igual que la transformación a texto al usar **write** o **writeln** para mostrar en la consola.
- Incluyendo al formateo de números reales o enteros utilizando el símbolo ":" .
- Tenga en cuenta que el "buffer" es un archivo **TEXT** . Todo lo que puede hacer con el buffer lo puede hacer en un **TEXT** .
- Esto los diferencia a **TEXT** de FILE OF **char**;

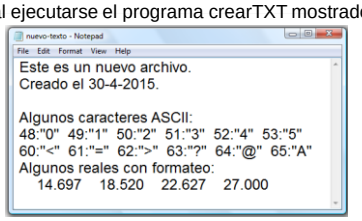

```
PROGRAM crearTXT;{Ej.: diferencias entre FILE OF char y TEXT}
VAR
T: TEXT;//archivo de texto
A: FILE OF char;//archivo de datos
BEGIN
assign(A,'texto.txt');//cuidado: la extensión txt no lo hace archivo TEXT
rewrite(A);//crea un archivo de caracteres (no de texto)
write(A, 'H');//escribe correctamente la letra H en el archivo
write(A, 126);//ERROR de compilación (tipos incompatibles, esperaba char)
assign(T, 'otro-texto.txt');
rewrite(T);//crea un archivo de texto
write(T, 'H');//escribe correctamente la letra H en el archivo
write(T, 126);//transforma el número 126 al texto "126" y lo escribe en T
write(T, 3.518);//transforma el real a texto y lo escribe en T
write(T, 3.518:0:2);//transforma el real al texto "3.52" y lo escribe en T
writeln(T);//deja renglón en blanco
writeln(T,'Hasta luego amigos');//Escribe en T todo el texto entre comillas
close(T); close(A);
END.
```

```
PROGRAM crearTXT; {Ejemplo: crear y escribir en archivo de texto}
VAR T: TEXT;
    dia, mes, anio, i: integer;
BEGIN
write('Ingrese la fecha (DD MM AAAA):');
readln(dia, mes, anio);
assign(T, 'nuevo-texto.txt');
rewrite(T);   // crea el archivo para escribir en él
writeln(T, 'Este es un nuevo archivo.'); // escribe texto
writeln(T,'Creado el',dia,'-',mes,'-',anio,'.'); // escribe texto y valores
writeln(T); // deja renglón en blanco
writeln(T, 'Algunos caractes ASCII');
FOR i:=48 TO 53 DO write(T, i, ':', '"',CHR(i),'"  ');
writeln(T); // baja de línea
FOR i:=60 TO 65 DO write(T, i, ':', '"',CHR(i),'"  ');
writeln(T);
writeln(T, 'Algunos reales con formateo: ');
FOR i:= 6 TO 9 DO write(T, SQRT(i*i*i):10:3);
close(T);
writeln('Archivo creado. Pulse enter');
readln;
END.
```

ARCHIVO GENERADO CON CREAR TXT



- Esta es la vista desde Notepad del archivo "nuevo-texto.txt" creado al ejecutarse el programa crearTXT mostrado antes.



PROBLEMA

Problema: escriba un programa para abrir un archivo de texto ya existente llamado "texto.txt", y contar cuantas líneas tiene.

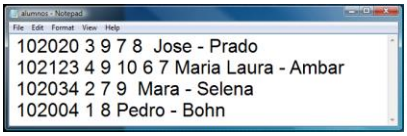
```
PROGRAM lineas; {Este programa cuenta las líneas de un archivo
de texto aprovechando el prodemiento predefinido readln que avanza
hasta el final de una línea del archivo (i.e., un enter) }
VAR T: TEXT; cant:integer;
BEGIN
   assign(T, 'texto.txt');
   reset(T); cant := 0;
   while not eof(T) do
     BEGIN
       readln(T); cant := cant + 1;
     END;
   writeln('Cantidad de líneas: ', cant);
   close(T);
END.
```

CARACTERÍSTICAS DE ARCHIVOS DE TEXTO (TEXT)

- Otra característica de **TEXT**, es que si el archivo a procesar tiene un formato uniforme y este formato se conoce de antemano, entonces utilizando los procedimientos **read** y **readln** puedo leer con variables de diferentes tipos simples y se realizará automáticamente la conversión adecuada.
- La conversión de texto a otro tipo de dato simple es la misma que ocurre al usar **read** o **readln** para lectura de lo ingresado por teclado (y que queda en el buffer).
- **Recuerde**: el "buffer" es un archivo de tipo **TEXT**.
  Todo lo que puede hacer con el buffer lo puede hacer con un **TEXT**.

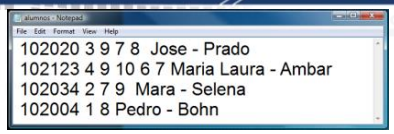
APLICACIÓN DE PROCESAMIENTO DE TEXTO

- Considere un archivo de tipo **TEXT** donde cada línea tiene un formato fijo con los siguientes elementos:
  - un número de LU, la cantidad (N) de notas de un alumno, seguido de una secuencia de N enteros que representan sus notas, y finalmente un texto con nombre y apellido (separados por un guión).
- Ejemplo:

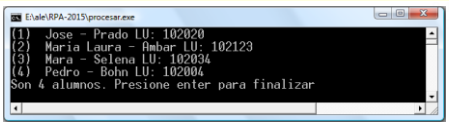


- Se asume que el archivo no tiene errores de carga de datos.

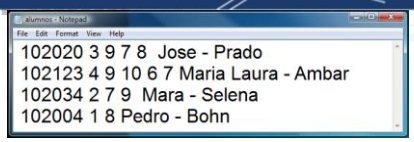
PROBLEMA PROPUESTO



Problema propuesto: escriba un programa que a partir de los datos en alumnos.txt, muestre un listado numerado como se muestra en el ejemplo a continuación. Para cada alumno se debe mostrar: nombre, apellido, "LU:" y número de libreta. Indicar al final la cantidad total.



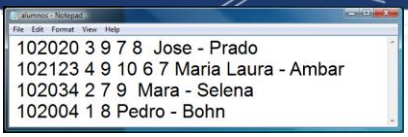
PROBLEMA PROPUESTO



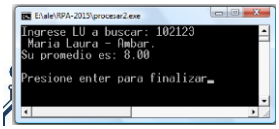
Algoritmo procesar alumnos.txt:
Inicializar contador en 0
Mientras no llegue al final del archivo:
leer un número de libreta
leer cantidad de notas y saltear todas las notas siguientes
incrementar contador en 1 y mostrar contador en pantalla
mostrar todo el texto hasta el final de la línea (nom. y apellido)
mostrar "LU:" y el número de libreta leído

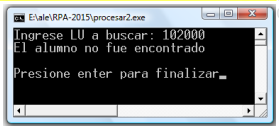

```
PROGRAM procesar; {procesa el texto de alumnos.txt}
VAR
T: TEXT; lu, cant, nota,i,cont: integer; elemento: char;
BEGIN
assign(T, 'alumnos.txt'); reset(T); cont:=0;
while not  eof(T) do
 BEGIN
  cont:=cont+1;
  read(T, lu);   // guardo lu para mostrar luego
  read(T, cant); // leo cantidad de notas a saltear
  for i:=1 to cant do
    read(T,nota); //salteo todas las notas
  write('(',cont,') ');
  while not eoln(T) do //  muestro nombre y apellido
    BEGIN
      read(T,elemento);  write(elemento);
    END;
    writeln(' LU: ', lu); // muestro LU leida antes
 END; // fin del while not oef(T)
write('Son',cont,'alumnos. Presione enter para terminar');
close(T);  readln;
END.
```

APLICACIÓN PROPUESTA



Problema propuesto: Escriba un programa que solicite un número de libreta (LU), y si lo encuentra en "alumnos.txt" muestre nombre, apellido y el promedio de las notas del alumno. Por ejemplo, si LU es 102034, mostrará en pantalla Mara – Selena, promedio: 8.





ALGORITMO

Problema propuesto: Escriba un programa que solicite un número de libreta (LU), y si lo encuentra en "alumnos.txt" muestre nombre, apellido y el promedio de las notas del alumno. Por ejemplo, si LU es 102034, mostrará en pantalla Mara – Selena, promedio: 8.

Algoritmo **promedio alumno**:
solicitar LU
Recorre el archivo de texto desde el comienzo.
Mientras no llegue al final del archivo y no encontró la LU
hacer: leer un número de libreta
Si el número leído es la LU buscada
entonces: calcular suma de notas
mostrar nombre y apellido
Mostrar promedio
de lo contrario: saltar al final de la línea del archivo

```
PROGRAM procesar2; //busca y muestra promedio alumno
VAR
T: text;
buscado, lu, cant, nota, i, suma: integer;
c: char;
encontre: boolean;
begin
assign(T, 'alumnos.txt');
reset(T);
write('Ingrese LU a buscar: ');
readln(buscado);
encontre := false;
while not eof(T) and not encontre do
…   // VER DIAPOSITIVA SIGUIENTE!!!
end; // while not eof
close(T);
if not encontre then
  writeln('El alumno no fue encontrado');
writeln;
write('Presione enter para finalizar');
readln;
end.
```

```
while not eof(T) and not encontre do
  begin
    read(T, lu);
    IF lu = buscado then
      begin // si no es el alu buscado, calculo y muestro los datos
        read(T, cant);
        suma:=0; encontre:=true;
        for i:=1 to cant do
          begin
            read(T,nota);
            suma:=suma+nota; // calcula suma
          end;
        while not eoln(T) do
          begin
            read(T,c); write(c); // mustra nombre-apellido
          end;
        writeln('.');
        writeln('Su promedio es:', suma/cant:5:2);
      end  // del if
    else
      readln(T); // si no es el alu buscado saltea la línea
  end; // while not eof
```