

RESOLUCIÓN DE PROBLEMAS Y ALGORITMOS

CLASE 9

Resolución de Problemas con secuencias de datos

011100
10011
10110
01110
01110
10011
10110
001
11
0

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

Departamento de Ciencias e Ingeniería de la Computación

Panel de Graduados DCIC 2016

Fecha: Lunes 19 de Septiembre
Hora: 18:00 hs
Lugar: Aula 4 del edificio azul de Pabellón

Imperdible
Coffee Break!!!

Nuevas panelistas, nuevas experiencias y nuevas historias

Diego Salpico
Se recibió como Licenciado en Ciencias de la Computación. Pasado de un año de la computación en Francia, regresó al lugar a la fundación de una empresa dedicada al desarrollo de aplicaciones de software y en el área de dispositivos móviles.

Patricia Marini
Se recibió de Ingeniero en Sistemas de Computación en la Universidad Nacional del Sur. Actualmente trabaja en el área de desarrollo de software en una empresa de tecnología.

Enzo Lopez
Se recibió de Ingeniero en Sistemas de Computación en el año 2010. Actualmente trabaja en el área de desarrollo de software en una empresa de tecnología.

Ellos hacen poco estuivoren en tu lugar y hoy triunfan en el mundo de la computación. Veni, escuchalos hablar de su experiencia y preguntales lo que quieras.

Inscríbete completando el formulario que se encuentra en el siguiente enlace:
<http://tinyurl.com/paneldegraduadosdcic2016>

Universidad Nacional del Sur

CONCEPTOS

Tipo de Dato: define el conjunto de valores posibles que puede tomar una variable, las operaciones que pueden aplicarse, y cual es la representación interna.

- Los tipos de datos en Pascal se pueden dividir en:
 - Simple**
Ejemplos que vimos en RPA:
INTEGER, REAL, CHAR, BOOLEAN
 - Estructurados**
Ejemplos que veremos en RPA:
FILE OF ... (archivos de datos) y **TEXT** (archivo de texto)

ARCHIVOS DE DATOS

- Las palabras reservadas **FILE** y **OF** se utilizan en Pascal como un **constructor de un nuevo tipo de dato** (estructurado) creado por el programador **sobre la base de otros tipos ya existentes**.
- Esto permite al programador trabajar con **archivos de datos**.

ARCHIVOS DE DATOS

- Como se puede ver abajo, el programador puede crear un nuevo tipo de dato archivo y luego declarar variables de ese tipo.
- También puede directamente declarar variables de tipo archivo.

El identificador reservado **TYPE** indica que lo que sigue son declaraciones de nuevos tipos de datos creados por el programador.

```
PROGRAM ejemplo;
TYPE
nuevo_tipo = FILE OF integer;
archivo_letras = FILE OF char;
VAR
numeros, valores : nuevo_tipo;
letras : archivo_letras;
temperaturas, lluvias : FILE OF real;
decisiones : FILE OF boolean;
```

OBSERVACIONES

- En Pascal, la palabra reservada **TYPE** permite al programador crear nuevos tipo de datos, sobre la base de otros tipos ya existentes.
- Vea el diagrama sintáctico que está en "bloque".

Ejemplo:

```
PROGRAM ejemplo;
TYPE nuevo_tipo = FILE OF integer;
VAR valores : nuevo_tipo;
```

Observación importante sobre los archivos declarados con FILE OF: estos archivos **no son archivos de texto**, son archivos de datos, y por lo tanto no podrá ver o editar su contenido con editores de texto como por ejemplo el "notepad". Para disponer de archivos de texto se debe usar el tipo predefinido **TEXT** (que veremos pronto).

PRIMITIVAS DE PASCAL PARA ARCHIVOS SECUENCIALES

- **assign(F,N):** vincula F con N (nombre del archivo en SO).
- **rewrite(F):** crea un archivo nuevo con el nombre N que está vinculado al **manejador de archivo F** (si ya existe otro archivo con ese nombre N se sobre-escribe y se pierde el viejo archivo).
- **write(F,e):** en un archivo F creado con rewrite, **escribe** el valor de "e" a continuación del último elemento de F.
- **close(F):** **cierra** el archivo vinculado al *manejador F*.
- **reset(F):** **abre** un archivo existente de nombre N para leer, y queda preparado para leer el primer elemento.
- **read(F,e):** **lee** un elemento del archivo F, copia el valor leído en "e" y queda preparado para leer el siguiente elemento (si existe) o queda en el fin del archivo.
- **eof(F)** (end of file – fin de archivo): **retorna TRUE** si se llegó al final de un archivo o **FALSE** en caso contrario.

Resolución de Problemas y Algoritmos - 2016

7

EJEMPLO PROPUESTO PARA HACER EN EL PIZARRÓN

Problema: crear un archivo de números reales llamado "numeros.reales" y permitir al usuario ingresar y almacenar en él tantos elementos en el archivo como quiera.

Algoritmo Cargar un archivo

Crear el archivo.

Repetir

- Leer un elemento,
- Almacenar en el archivo
- Preguntar si quiere cargar otro número

Hasta que no quiera cargar más

Cerrar el archivo.

fin.

Resolución de Problemas y Algoritmos - 2016

8

PARTE DE LA IMPLEMENTACIÓN QUE HICIMOS EN EL PIZARRÓN

```
PROGRAM crear_archivo_reales;
TYPE
archivo_reales = FILE OF real; //tipo nuevo creado por el
                           constructor
VAR
archi: archivo_reales; //manejador del archivo de reales
opcion: char; //letra que contendrá la respuesta del usuario
BEGIN
assign(archi, 'numeros.reales'); //vincula el nombre con el
manejador
rewrite(archi); //crea el archivo vacío en una carpeta del disco
duro
REPEAT
...//el resto del programa fue desarrollado en el pizarrón
UNTIL (opcion = 'N') or (opcion = 'n');
close(archi); //el archivo queda cerrado y listo para ser usado por
otros
END.
```

Resolución de Problemas y Algoritmos - 2016

9

EJEMPLO

Problema: escriba un programa que busque **cuantas veces está el elemento E** (ingresado por el usuario) en el archivo "numeros.reales" el cual ya fue creado.

Algoritmo contar cuantas veces

Abrir el archivo para leer

Solicitar número a buscar

Asignar el valor inicial 0 a cantidad de veces

Repetir mientras no sea fin de archivo (EOF)

Leer un elemento del archivo

Si el elemento es el buscado entonces
incrementar cantidad de veces en uno

Cerrar el archivo.

fin.

Resolución de Problemas y Algoritmos - 2016

10

PARTE DE LA IMPLEMENTACIÓN QUE HICIMOS EN EL PIZARRÓN

```
PROGRAM leer_archivo_reales;
TYPE
archivo_nums = FILE OF real; //tipo nuevo creado por el constructor
VAR
numeros: archivo_nums; //manejador del archivo de reales
buscado: real; //contendrá el nro buscado ingresado por el usuario
leido: real; //contendrá valores que va leyendo de a uno del archivo
cantidad: integer; //contendrá la cantidad de apariciones encontradas
BEGIN
assign(numeros, 'numeros.reales'); //vincula el nombre con el
manejador
reset(numeros); //abre el archivo para leer
... //aquí se implementó el ingreso de datos e inicialización
WHILE NOT eof(numeros) DO //mientras no sea el fin del archivo numeros
BEGIN
... //el resto del programa fue desarrollado en el pizarrón
END; //end del while
close(numeros); //el archivo queda cerrado y listo para ser usado por otros
END
```

Resolución de Problemas y Algoritmos - 2016

11