RESOLUCIÓN DE PROBLEMAS Y ALGORITMOS

CLASE 20

Resolución de problemas utilizando **recursión**

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

PROBLEMA PROPUESTO: CANTIDAD DE ELEMENTOS

Problema propuesto: Escriba un planteo recursivo y luego una función que respete ese planteo para contar la cantidad de dígitos de un número.

- Ejemplos:
  - 1234 (tiene 4 dígitos)
  - 12 y -34 (tienen 2 dígitos)
  - 2 y 0 (tienen 1 dígito)

Planteo recursivo: **cantidad de dígitos de N**
Caso base: si N tiene un dígito entonces la cantidad es 1
Caso general: si N tiene más de un dígito entonces la cantidad es 1 + **cantidad de dígitos de N sin uno de sus dígitos**.

PROBLEMA PROPUESTO: CANTIDAD DE ELEMENTOS

Problema propuesto: Escriba un planteo recursivo y luego un **procedimiento** que respete ese planteo para contar la cantidad de elementos de un archivo.

- Ejemplos:
  - 1,2,3,4 (tiene 4 elementos)
  - 12, -34 (tienen 2 elementos)
  - Archivo vacío (tienen 0 elementos)

Planteo recursivo: **cantidad de elementos de un archivo A**
Caso base: si el archivo está vacío
entonces la cantidad es 0 (cero)
Caso general: si el archivo no está vacío,
entonces la cantidad es 1 + la **cantidad de elementos del archivo A sin su primer elemento**.


```
PROGRAM prueba1;
TYPE Telemento= integer; Tarchi = file of Telemento;
VAR A: Tarchi; cantidad: integer;
 PROCEDURE contar(VAR F: Tarchi; VAR cant:integer);
 {cuenta los elementos de un archivo}
 VAR ele: telemento; aux:integer;
 BEGIN
  IF EOF(F) THEN
   cant:=0 {caso base}
  ELSE
   BEGIN {caso general}
    read(F,ele); {leo el primero elemento}
    contar(F, aux);{llamo con F sin su 1er elemento}
    cant:= aux + 1;
   END;
 END;
BEGIN
assign (A, 'el-archivo');
reset(A); contar(A, cantidad); close(A);
writeln('cantidad de elementos: ',cantidad);
END.
```

¿Qué pasaría si hago reset y close dentro del procedimiento recursivo?

IMPLEMENTACIÓN EN PASCAL

- Como fue dicho antes no hay una única forma de escribir un procedimiento que respete el planteo.
- Hay que tener cuidado donde realiza "assign", "reset" y "close" del archivo.
- **Pregunta teórica:**
  ¿necesita hacer una primitiva recursiva diferente para cada tipo diferente de archivo?
  Escriba su respuesta y consulte sus dudas.

- **Tarea:** (para practicar) Realice una función recursiva que respete el planteo anterior y cuente la cantidad de elementos de un archivo.

OBSERVACIONES

- En el programa anterior (**prueba1**) **assign**, **reset** y **close** del archivo se realizan en el bloque principal. Vea por ejemplo lo que pasa en estos dos casos que está mal implementado:

```
PROCEDURE contar(var F:Tarchi; …
{cuenta los elementos de un archivo}
VAR ele: telemento; aux:integer;
BEGIN
reset(F);
IF EOF(F) THEN …
```

MAL

```
…
IF EOF(F) THEN
 cant:=0
ELSE
 BEGIN {caso general}
  reset(F);
  read(F,ele);
  contar(F, aux);
  cant:= aux +1;
 END;
```

MAL

- En cualquiera de los dos ejemplos anteriores cada vez que se llama recursivamente se ejecuta nuevamente reset(F), con lo cual se vuelve a comenzar a leer del primer elemento y se produce una ejecución infinita, ya que nunca se reduce el archivo en un elemento (no respeta el planteo).

OBSERVACIONES

- En el programa anterior (**prueba1**), en el procedimiento recursivo "**contar**" la variable local "**aux**" es utilizada para almacenar la cantidad de elementos del "*archivo sin su primer elemento*".
- Realice la traza y verá que en cada llamada recursiva "**aux**" recibe la cantidad calculada por la invocación recursiva y luego el parámetro por referencia "**cant**" retorna "**aux**" + 1 a quién lo llamó.
- En el programa siguiente (**prueba2**) hay otra versión correcta del procedimiento recursivo que también respeta el planteo **pero no usa** "**aux**". Realice una traza para ver la diferencia en ejecución.


```
PROGRAM prueba2;
TYPE Telemento= integer; Tarchi = file of Telemento;
VAR A: Tarchi; cantidad: integer;
 PROCEDURE contar(VAR F: Tarchi; VAR cant:integer);
 {cuenta los elementos de un archivo}
 VAR ele: telemento;
 BEGIN
  IF EOF(F) THEN
   cant:=0 {caso base}
  ELSE
   BEGIN {caso general}
    read(F,ele); {leo el primero elemento}
    contar(F,cant);
    cant:= cant + 1;
   END;
 END;
BEGIN
assign (A, 'el-archivo');
reset(A); contar(A, cantidad); close(A);
writeln('cantidad de elementos: ',cantidad);
END.
```

Observe que cambia en la traza si no uso la variable local "aux"

OBSERVACIONES

- En el programa siguiente (prueba3) hay otra versión correcta del procedimiento recursivo que también respeta el planteo.
- En este caso contar abre y cierra el archivo.
- Para hacer esto tiene su propio procedimiento interno que hace la tarea recursiva.
- Realice una traza para ver la diferencia en ejecución.


```
PROGRAM prueba3;
TYPE Telemento= integer; Tarchi = file of Telemento;
VAR A: Tarchi; cant elem: integer;
PROCEDURE contar(VAR F:Tarchi;VAR cantidad:integer);
 PROCEDURE contarRec(VAR F:Tarchi;VAR cant:integer);
 VAR ele: telemento;
 BEGIN
  IF EOF(F) THEN
   cant:=0 {caso base}
  ELSE
   BEGIN {caso general}
    read(F,ele); {leo el primero elemento}
    contarRec(F,cant);
    cant:= cant + 1;
   END;
 END;
BEGIN {abre el archivo, llama al recursivo y cierra el archivo}
reset(F); contar_rec(F, cantidad); close(F);
END;
BEGIN
assign (A, 'el-archivo'); contar(A, cant_elem);
writeln('cantidad de elementos: ',cant_elem);
END.
```

PROBLEMA PROPUESTO: CANTIDAD DE ELEMENTOS

Problema propuesto: Escriba un planteo recursivo y luego un procedimiento que respete ese planteo para contar la cantidad de apariciones de un elemento de un archivo.

- **Ejemplo**: el 3 está 2 veces en F: 4 3 4 3 2

Planteo recursivo: **Cantidad de apariciones de E en F**
Caso base: Si F está vacío,
la cantidad de apariciones de E en F es 0.
Caso general: Si F no está vació entonces
la cantidad de apariciones de E en F, es **la cantidad de apariciones de E en F sin su primer elemento**,
más uno si el primer elemento de F es E.


```
PROGRAM prueba1;
TYPE Tele= integer; Tarchi = file of Tele;
VAR A: Tarchi; cantidad: integer; E:Tele
 PROCEDURE contar(E:Tele;VAR F: Tarchi; VAR cant:integer);
 {cuenta las apariciones de E en un archivo F}
 VAR aux: tele;
 BEGIN
  IF EOF(F) THEN
   cant:=0 {caso base}
  ELSE
   BEGIN {caso general}
    read(F,aux);
    contar(E,F,cant);
    IF aux = E THEN cant:= cant + 1;
   END;
 END;
PROCEDURE leer_elemento(VAR E: Telemento); {..completar..}
BEGIN
assign (A, 'el-archivo'); leer_elemento(E);
reset(A); contar(E,A,cantidad); close(A);
writeln('cantidad de elementos: ',cantidad);
END.
```

PROBLEMA PROPUESTO: CANTIDAD DE ELEMENTOS

Problema propuesto: Escriba un planteo recursivo y luego una función que respete ese planteo para sumar los elementos de un archivo.

Ejemplo: 1, 2, 3, 4 (suma 10)
12, -34 (suma -22)
archivo vacío (suma 0)

Planteo recursivo: suma de elementos de un archivo A
Caso base: si el archivo está vacío
entonces la suma es 0 (cero)
Caso general: si el archivo no está vacío,
entonces la suma es el primer elemento + la **suma de elementos del archivo A sin su primer elemento**.