



## Resolución de Problemas y Algoritmos

**Clase 21:**  
Resolución de problemas utilizando recursión



**Dr. Alejandro J. García**  
http://cs.uns.edu.ar/~ajg



Departamento de Ciencias e Ingeniería de la Computación  
Universidad Nacional del Sur  
Bahía Blanca - Argentina

### Problema propuesto: cantidad de elementos

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:** 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.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 2

### Problema propuesto: cantidad de elementos

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 (tiene 2 elementos)  
archivo vacío (tiene 0 elementos)

**Planteo:** 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.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 3

```

program prueba1;
type Telemeto = integer; Tarchi = file of Telemeto;
var A: Tarchi; cantidad: integer;

Procedure contar (var F: Tarchi; var cant:integer);
{cuenta los elementos de un archivo}
var ele: telemeto; 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 primer elemento}
cant:= aux +1;
end;
end;
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?

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 4

### 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.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 5

### 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: telemeto; aux:integer;
begin
reset(F); ← MAL
if EOF(F) then ... ← 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).

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 6

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:  
 "Resolución de Problemas y Algoritmos. Notas de Clase". Alejandro J. García. Universidad Nacional del Sur. (c) 02/06/2016

**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);
var ele: telemento; {cuenta los elementos de un archivo}
begin
  if EOF(F) then cant:=0 {caso base}
  else begin read(F,ele); {caso recursivo}
            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"

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 8

**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; cantidad_elem: integer;

Procedure contar (var F: Tarchi; var cantidad:integer);
Procedure contar_rec (var F: Tarchi; var cant:integer);
var ele: telemento; {cuenta los elementos de un archivo}
begin
  if eof(F) then cant:=0 {caso base}
  else begin read(F,ele); {caso recursivo}
            contar_rec(F,cant);
            cant:=cant+1;
          end;
end;

begin {abra 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, cantidad_elem);
writeln('cantidad de elementos: ',cantidad_elem); end.
    
```

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 10

**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:** 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 vacío 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 Telemento = integer; Tarchi = file of Telemento;
var A: Tarchi; cantidad: integer; E: Telemento;

Procedure contar (E: Telemento; var F: Tarchi; var cant:integer);
{cuenta las apariciones de E en un archivo F}
var aux: telemento;
begin
  if EOF(F) then cant:=0 {caso base}
  else begin read(F,aux); {caso recursivo}
            contar(E, F, cant);
            if aux = E then cant:= cant +1; end;
          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 apariciones: ',cantidad);
end.
    
```

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 12

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:  
 "Resolución de Problemas y Algoritmos. Notas de Clase". Alejandro J. García. Universidad Nacional del Sur. (c) 02/06/2016

**Problema propuesto: cantidad de elementos**

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

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

**Planteo:** 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.