PROGRAMACIÓN I

AÑO 2025

Corrección y Eficiencia

TEMAS de la CLASE

- Concepto de Eficiencia

  Ejemplo

Eficiencia de un Algoritmo

Medición de la Memoria utilizada en un programa

- Se puede calcular únicamente la cantidad de memoria estática que utiliza el programa.
- Se analizan las constantes como así también las variables declaradas y el tipo correspondiente.

¿Cuánta memoria se utiliza?

```
{declaración de tipos}
Type
cadena10 = string [10];
PtrString = ^cadena10;
Datos = record
            Nombre: cadena10;
            Apellido: cadena10;
            Edad: integer;
            Altura: real
        End;
Personas = array [1..100] of datos;
PtrDatos = ^datos;

var
  frase : PtrString;
  s : cadena10;
  puntero : PtrDatos;
  p : personas;
```

Eficiencia – Análisis Teórico – Regla 1

Supongamos que se desea conocer el valor en grados centígrados de una temperatura medida en grados Fahrenheit. Se tiene la siguiente función que recibe un valor expresado en F y lo devuelve en C

```
Function convertirFaC (tem:real): real;
begin
  convertirFaC := (tem-32) * 5 /9;
end.
```

¿Cuántas operaciones elementales realiza la función?

Recordar que una operación elemental una unidad de tiempo

Eficiencia – Análisis Teórico – Regla 2 (FOR)

Se tiene el siguiente programa que lee una cantidad N y luego lee esa vcantidad N de temperaturas registradas en una ciudad de la Pcia de Bs. As y calcula la temperatura promedio. ¿Cuántas operaciones elementales realiza el programa?

```
Program temperaturas;

Var valor, total: real;
    n, i: integer;
Begin
 total:= 0; read (n);
 for i:= 1 to n do begin
   read (valor);
   total:= total + valor;
 end;
 prom:= total div n;
   write(´Temperatura   Promedio:´,
prom);
end;
```

Se debe calcular la cantidad de operaciones elementales que se ejecutan dentro del FOR y multiplicarla por la cantidad de veces que se ejecuta la instrucción FOR.

Además, la instrucción FOR realiza:

-asignación inicial i:=1 (1)

-testeo de i <=n (n+1)

-incrementos de i:= i+1 (n*2) entonces 1+ n+1 + 2n

En general : 3*n+2, siendo n la cantidad de repeticiones

Total-> (2*n) + (3*n+2) + 3 = 5n + 5 op. elem.

Eficiencia – Análisis Teórico – Regla 2 (FOR)

Se tiene el siguiente programa que lee 30 temperaturas registradas durante el mes de abril de 2025 en una ciudad de la Pcia de Bs. As. y calcula la temperatura promedio. ¿Cuántas operaciones elementales realiza el programa?

```
Program temperaturas;

Var valor, total: real;
Begin
 total:= 0;
 for i:= 1 to 30 do begin
   read (valor);
   total:= total + valor;
 end;
 prom:= total div 30;
 write(´Temperatura Promedio:´, prom);
end;
```

Se debe calcular la cantidad de operaciones elementales que se ejecutan dentro del FOR y multiplicarla por la cantidad de veces que se ejecuta la instrucción FOR.

Además, la instrucción FOR realiza:

-asignación inicial i:=1 (1)

-testeo de i <=30 (31)

-incrementos de i:= i+1 (30*2) entonces 1+ 31 + 60 = 92 op

En general : 3*n+2, siendo n la cantidad de repeticiones

Total-> (2*30) + (3*30+2) + 3 = 155 op. elem.

Eficiencia – Análisis Teórico

- Cálculo Teórico del tiempo de ejecución:

```
Type temperaturas = array [1..30] of real;

Function contar ( tem:temperaturas): integer;
 Var i: 1..30; can10 : integer;
 begin
   can10 := 0;                {1}
   {recorrido total del vector}
   For  i  := 1  to  30 do {2}
     If (tem[i] = 10 and I < 5) then        {3}
                        can10 := can10 + 1 {4}
                        else
                        can10:= i * I + 4; {5}
   contar := can10;           {6}
 end.
```

- La línea {1} -> 1 unidad de tiempo
- La línea {2} -> 3n + 2 = 92 unidades de tiempo
- La línea {3} evalúa dos condiciones -> 3 unidad de tiempo
- La línea {4} -> **2** unidades de tiempo y la línea {5} consume 3 □ 3 unidades.
- Por la tanto, dentro del FOR se cuentan 3 unidades □ **6 * 30**
- La línea {5} -> 1 unidad

Total de operaciones = 2 + 180 + 92 (como máximo!!!) ¿Por qué?

Cantidad de unidades de tiempo = 274 (como máximo!!!)

Eficiencia – Regla 2 (FOR ANIDADOS)

Aplicando la Regla del FOR, analicemos ahora el tiempo de ejecución del siguiente programa:

```
Program FA;
Var
  valor,i,j,suma :integer;
Begin
 suma:=0; {1}
 for j:= 1 to 300 do   {2}
  for i:= 1 to 200 do  {3}
    suma:= suma + I;   {4}
End.
```

Cantidad de operaciones (unidades de tiempo)

- La línea {1} -> 1
- La línea {2} -> 3n + 2 = 3*300 +2 = 902
- La línea {3} -> 3n + 2 = 3*200 +2 = 602
- La línea {4} -> **2**

(((2 * 200 + 602) * 300) + 902) + 1= 301.503 ut

{4} * 200 + {3}

{4} * 200 + {3} * 300

{4} * 200 + {3} * 300 + {2}

{4} * 200 + {3} * 300 + {2} + {1}

Eficiencia – Regla 3 (WHILE/REPEAT…UNTIL)

Supongamos que el siguiente programa lee montos de facturas (hasta leer un monto -1) e informa el monto total facturado y la cantidad de facturas. ¿Cuál es el tiempo de ejecución de la solución propuesta?

```
Program W;
Var
monto, total: real;
cant :integer;
Begin
 total:=0;  {1}
 cant:= 0; {2}
 read (monto);
 while (monto<>-1) do begin {3}
   cant:= cant+1; {4}
   total:=total + monto; {5}
   read (monto);
 end;
 writeln (`Total: ´, total)
 writeln (`Cantidad: ´, oant)
End.
```

Se debe calcular la cantidad de operaciones elementales que se ejecutan dentro del WHILE y multiplicarla por la cantidad de veces que se ejecuta el WHILE. Como no se conoce esa cantidad se considera el PEOR CASO. Por ejemplo, se supone una cantidad de notas n…

Cantidad de operaciones (unidades de tiempo)

- La línea {1} -> 1
- La línea {2} -> 1
- La línea {3} -> n + 1
- La línea {4} -> **2**
- La línea {5} -> **2**

$$((4 * n) + n+1) + 2 = (5n + 3)\ ut$$

¿Qué significa que las unidades de tiempo queden expresadas en términos de n?

Eficiencia – Regla 4 (IF THEN/ELSE)

Calcular la cantidad de operaciones elementales del siguiente programa

En el caso de una sentencia IF en su forma completa (then/else), debe calcularse la cantidad de operaciones que se realizan en cada parte y se debe elegir aquella que consuma más tiempo o sea la mayor cantidad de operaciones (el PEOR CASO).

```
Program uno;
Var
  valor, i, j, suma :integer;
 Begin
  read (valor);
  if (valor >8) then begin
                      suma:=0;
                      for i:= 1 to 3000 do
                          suma:= suma + I;
                    end
                    else begin
                      suma:=0;
                      for j:= 1 to 300 do
                         for i:= 1 to 200 do
                            suma:= suma + I;
                    end;
    end;
End.
```

((2*3000)+9002) + 1= 15.003 ut

(((2 * 200 + 602) * 300) + 902) + 1= 301.503 ut

Total de operaciones = 1 + 301.503= 301.504 (como máximo!!!) ¿Por qué?

Eficiencia

Ejercicio: Calcule el tiempo de ejecución del módulo agregarFinal y la memoria estática y dinámica en estos dos procedimientos e indique cuál considera más eficiente y por qué.

```
Type Lista= ^Nodo;
       Nodo= Record
                Datos: real;
                sig: Lista;
              End;
       Punteros= Record
                   pri, ult: Lista;
                 end;

Procedure agregarFinal (Var pun: punteros; nu:real);
var nuevo:lista;
Begin
  new (nuevo);
  nuevo^.datos:=nu;
  nuevo^.sig:=nil;
  if (pun.pri = nil)
  then pun.pri:= nuevo
  else pun.ult^.sig:=nuevo;
 pun.ult:=nuevo;
End;
```

```
Type  Lista= ^Nodo;
       Nodo= Record
                Datos: real;
                sig: Lista;
              End;

Procedure agregarFinal (Var pri: Lista; num:real);
var nuevo, pos:lista;
Begin
  new (nuevo);
  nuevo^.sig:= nil;
  nuevo^.datos:=num;
  if (pri = nil) then   pri:=nuevo
  else  begin
          pos:= pri;
          while (pos^.sig <> nil) do  pos:= pos^.sig;
          pos^.sig:= nuevo;
        end;
End;
```