TECNOLOGÍAS EN EDUCACIÓN MATEMÁTICA

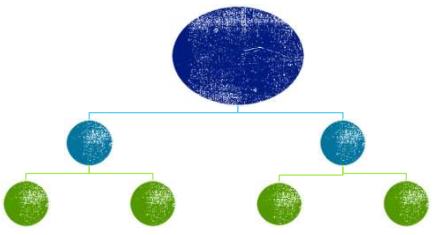
MODULO 16

Dpto. de Ciencias e Ingeniería de la Computación
UNIVERSIDAD NACIONAL DEL SUR
Año 2019

Descomposición de Problemas en Sub-problemas

Cuando la complejidad de los problemas aumenta, la tarea de hallar una solución se torna más difícil.

Una metodología para reducir la complejidad consiste en plantear la solución del problema a partir de la solución de una serie de sub-problemas más sencillos que forman parte del problema original.



Descomposición de Problemas en Sub-problemas

Problema: Dado un número se desea conocer el producto y la suma de sus dígitos.

```
Algoritmo SumaYprod
DE: N (entero)
DS: sum, prod (enteros)
sum ← 0
prod ← 1
mientras (N>0) hacer
        sum ← sum + N mod 10
        prod ← prod * N mod 10
        N ← N div 10
```

Escribirlo como primitiva.

Pascal - Procedimientos

```
procedure SumaYprod (N: integer; var sum, prod: integer);
begin
sum := 0
prod := 1
while (N>0) do
 begin
        sum := sum + N mod 10;
        prod := prod * N mod 10;
        N := N div 10;
 end;
end;
```

Datos de salida con la palabra reservada VAR adelante.

Pascal - Procedimientos

Problema: Dados dos números, se desea saber si coincide la suma o el producto de sus dígitos.

```
Algoritmo Verificar
DE: N1, N2 (enteros)
DS: CoincideAlgo (lógico)
Daux: sumN1, prodN1, sumN2, prodN2

SumaYprod(N1, sumN1, prodN1)
SumaYprod(N2, sumN2, prodN2)
Si (sumN1=sumN2) O (prodN1=prodN2)
        entonces CoincideAlgo ← Verdadero
        sino CoincideAlgo ← Falso
```

Pascal - Procedimientos

```
program verificar;
  var
        N1, N2, sumN1, sumN2, prodN1, prodN2: integer;
        coincideAlgo: boolean;
  procedure SumaYprod (N: integer; var sum, prod: integer); …
begin
writeln('Ingrese dos números: ');
readln(N1, N2);
SumaYprod(N1, sumN1, prodN1);
SumaYprod(N2, sumN2, prodN2);
if ((sumN1=sumN2) OR (prodN1=prodN2))
    then        coincideAlgo:=true
    else        coincideAlgo:=false;
if coincideAlgo then writeln ('HAY COINCIDENCIA!')
        else writeln('NO HAY COINCIDENCIA.');
end.
```

Pascal - Procedimientos

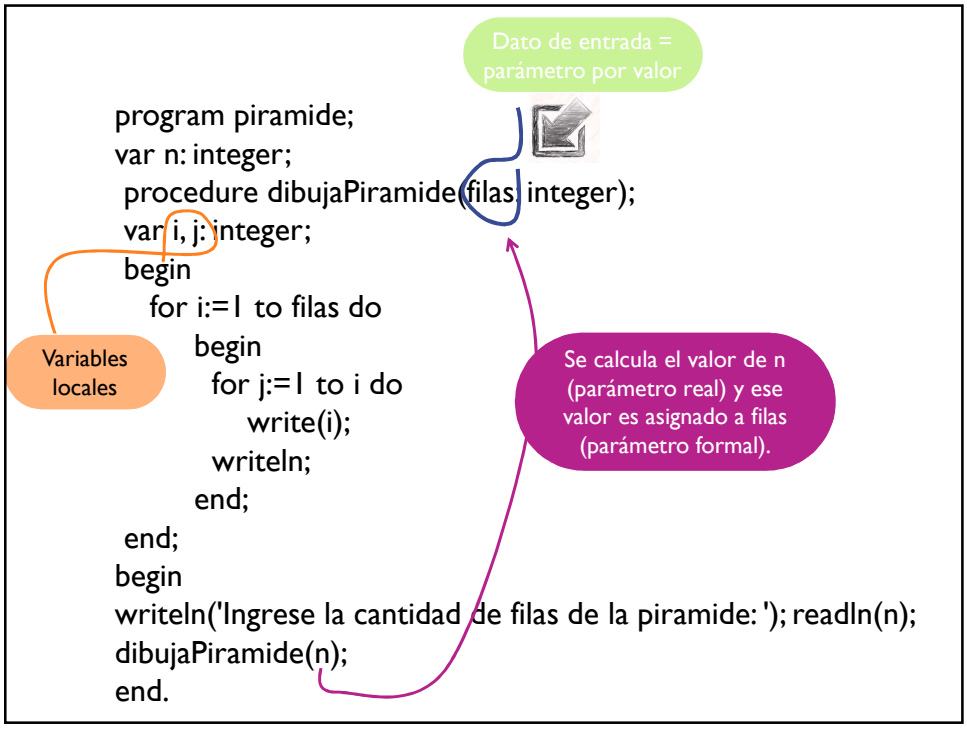
Problema: Dibujar una pirámide como la que se muestra a continuación, donde la cantidad de filas depende de un entero filas

```
1
22
333
4444
```

Pascal - Procedimientos

```
procedure dibujaPiramide(filas: integer);
 var i, j: integer;
 begin
   for i:=1 to filas do
       begin
         for j:=1 to i do
             write(i);
         writeln;
       end;
 end;
```

```
1
22
333
4444
```



Pascal - Procedimientos

Problema: Dado un número entero, devolver en anterior y el siguiente

```
procedure sigant(n: integer; var na, ns: integer);
begin
   na := n – 1;
   ns := n + 1;
end;
```

Pascal - Procedimientos

```
procedure sigant(n: integer; var na, ns:
integer);
begin
   na := n – 1;
   ns := n + 1;
end;
```

El procedimiento *sigant* tiene tres parámetros.

- El parámetro n está pasado por valor, cuando empieza la ejecución del procedimiento, se reserva una locación de memoria para n, que se inicializa con el valor del parámetro real y se destruye cuando termina el procedimiento.
- Los parámetros na y ns están pasados por referencia (también llamados por variable), son referencias a las locaciones de memoria que corresponden a los parámetros reales. También se reservan locaciones de memoria para estas referencias.

Pascal - Procedimientos

```
procedure sigant (n: integer; var na,
ns: integer);
begin
   na := n – 1;
   ns := n + 1;
end;
```

```
sigant(a, b, c);
```

La primera invocación de sigant asigna el valor del parámetro real a al parámetro formal n.

El parámetro real b queda ligado al parámetro formal na, cualquier modificación en na, afecta al valor almacenado en b.

El parámetro real c queda ligado al parámetro formal ns, cualquier modificación en ns, afecta al valor almacenado en c.

Cuando termina la ejecución se destruyen las locaciones de memoria ligadas a los parámetros.

Pascal - Procedimientos

```
procedure sigant (n: integer; var na,
ns: integer);
begin
   na := n – 1;
   ns := n + 1;
end;
```

```
sigant(a, c, b);
```

La segunda invocación de sigant asigna el valor del parámetro real a al parámetro formal n.

El parámetro real c queda ligado al parámetro formal na, cualquier modificación en na, afecta al valor almacenado en c.

El parámetro real b queda ligado al parámetro formal ns, cualquier modificación en ns, afecta al valor almacenado en b.

Cuando termina la ejecución se destruyen las locaciones de memoria ligadas a los parámetros.

Pascal - Procedimientos

ACTIVACIÓN DE UN PROCEDIMIENTO

- Un procedimiento se invoca desde una instrucción.
- Una vez invocado el control (la ejecución) pasa al procedimiento.
- El bloque de código que incluye la invocación, se suspende.
- Cuando comienza la ejecución se reserva espacio en memoria para las variables locales y para los parámetros por valor, que se inicializan con los valores de los parámetros reales. También para las referencias almacenadas en los parámetros por variable.

TECNOLOGÍAS EN EDUCACIÓN MATEMÁTICA

FIN MODULO 16

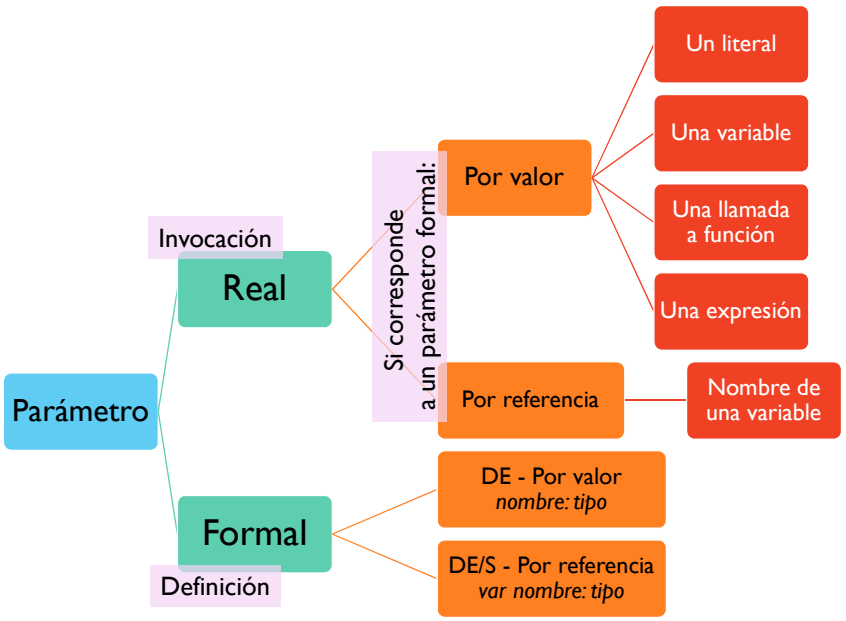
Dpto. de Ciencias e Ingeniería de la Computación
UNIVERSIDAD NACIONAL DEL SUR
Año 2019

Pascal – Procedimientos y Funciones

MANEJO DE MEMORIA:

- Un parámetro por valor se inicializa en el momento de la invocación del procedimiento con el valor del parámetro real.
- Al terminar la ejecución del procedimiento la variable asociada al parámetro formal se destruye y su valor se pierde.
- Un parámetro por referencia introduce un nuevo nombre para referirse a la celda de memoria que corresponde al parámetro real.

Pascal – Procedimientos y Funciones




```
Program parametros;
var pa1, pa2: integer;
procedure p(pf1: integer; var pf2: integer);
var local: integer;
begin
  local:=pf1+pf2;
  writeln(pf1, pf2, local);
  pf1:= pf1+1; pf2:=local+1;
  writeln(pf1, pf2, local);
end;
begin
   pa1:=1;  pa2:=3;
   writeln(pa1, pa2);
   p(pa1, pa2);
   writeln(pa1, pa2);
end.
```

Variables globales

Parámetros formales

Variables locales

Parámetros reales (actuales o efectivos)

Pascal – Procedimientos y Funciones

Resumen

Los datos de entrada:

- En el programa principal se leen con READ.
- En las primitivas se pasan como parámetro por valor (entre los paréntesis del encabezado).

Los datos de salida:

- En el programa principal se muestran con WRITE.
- En las primitivas:
  - Cuando es UNO SOLO se implementa una función y se devuelve en el nombre de la función.
  - Cuando hay cero o más de un dato de salida, se implementa un procedimiento y los datos de salida se devuelven en parámetros por variable o por referencia (entre los paréntesis del encabezado con la palabra reservada VAR adelante).