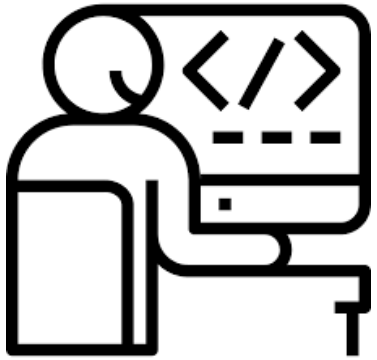


Conceptos de Algoritmos Datos y Programas

CADP – **TEMAS**

- Parámetros por valor
- Parámetros por referencia
- Ejercicios – PREGUNTAS FINALES

PARAMETRO POR VALOR

Un dato de entrada por valor es llamado parámetro IN y significa que el módulo recibe (sobre una variable local) un valor proveniente de otro módulo (o del programa principal).

Con él puede realizar operaciones y/o cálculos, pero no producirá ningún cambio ni tampoco tendrá incidencia fuera del módulo.

Con qué tipo de parámetro se relaciona?

Cómo se declaran?

Cómo se usan?

PARAMETRO POR VALOR

```
procedure uno (nombre1: tipo; nombre2: tipo);

  var
     …

  Begin

    Uso de los parámetros con nombre1 y nombre2

  End;
```

CADP – MODULARIZACION

COMUNICACION

PARAMETRO POR VALOR

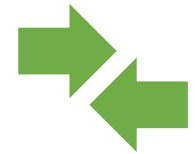
```
Program porValor;

procedure uno (num: integer);
Begin
    if (num = 7)  then
      num:= num + 1;
    write (num);
end;
var
   x: integer;
begin
  x:= 7;
  uno (x);
end.
```

Dentro del procedimiento uno, el parámetro num copia el valor enviado por x (variable del programa)

Cómo funciona?

CADP – MODULARIZACION

COMUNICACION



```
Program porValor;

procedure uno (num: integer);
Begin
    if (num = 7)  then
      num:= num + 1;
    write (num);
end;
var
   x: integer;
begin
  x:= 7;
  uno (x);
end.
```

Qué pasa si después de llamar al procedimiento uno en el programa imprimo num?

Procedimiento uno
Variables locales
Parámetros

num = 8

Imprime 8

Programa ppal
Variables globales
Variables de prog

x = 7

Imprime 7

MEMORIA

CADP – MODULARIZACION

COMUNICACION

```
Program porValor;

procedure uno (num: integer);
Begin
    if (num = 7)  then
      num:= num + 1;
    write (num);
end;
var
   num: integer;
begin
  num:= 7;
  uno (num);
end.
```

Qué pasa si después de llamar al procedimiento uno en el programa imprimo num?

Procedimiento uno
Variables locales
Parámetros

num = 8

Imprime 8

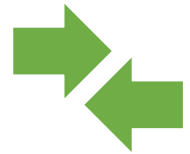
Programa ppal
Variables globales
Variables de prog

num = 7

Imprime 7

MEMORIA

CADP – MODULARIZACION COMUNICACION



```
Program porValor;

procedure uno (x: integer);
Begin
    if (x = 7)  then
      x:= x + 1;
    write (x);
end;
var
   x: integer;
begin
  x:= 7;
  uno (x);
end.
```

Qué valores imprimen?

```
Program porValor;

procedure uno (num: integer);
Var
 x:integer;
Begin
    if (num = 7) then
       num:= num + 1;
    x:= num;
    write (num); write (x);
end;
var
   x: integer;
begin
  x:= 7;
  uno (x);
end.
```

PARAMETRO POR REFERENCIA

La comunicación por referencia (OUT, INOUT) significa que el módulo recibe el nombre de una variable (referencia a una dirección) conocida en otros módulos del sistema.

Puede operar con ella y su valor original dentro del módulo, y las modificaciones que se produzcan se reflejan en los demás módulos que conocen la variable.

Con qué tipo de parámetro se relaciona?

Cómo se declaran?

Cómo se usan?

PARAMETRO POR REFERENCIA

```
procedure uno (var nombre1: tipo; var nombre2: tipo);

  var
     …

  Begin

    Uso de los parámetros con nombre1 y nombre2

  End;
```

CADP – MODULARIZACION

COMUNICACION

PARAMETRO POR REFERENCIA

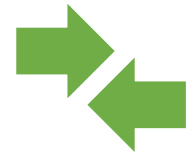
```
Program porReferencia;

procedure uno (var num: integer);
Begin
    if (num = …)  then
      num:= num + 1;
    write (num);
end;
var
   x: integer;
begin
  x:= 7;
  uno (x);
end.
```

Dentro del procedimiento uno, el parámetro num comparte la dirección de memoria con x (variable del programa)

Cómo funciona?

CADP – MODULARIZACION

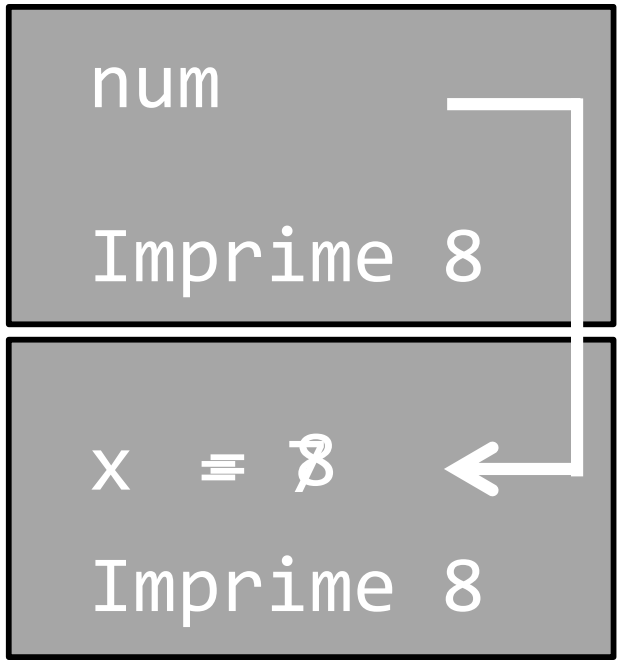
COMUNICACION



```
Program porReferencia;

procedure uno (var num: integer);
Begin
    if (num = 7)  then
    num:= num + 1;
    write (num);
end;
var
   x: integer;
begin
  x:= 7;
  uno (x);
end.
```

Procedimiento uno
Variables locales
Parámetros

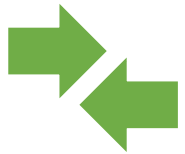
Programa ppal
Variables globales
Variables de prog



MEMORIA

CADP – MODULARIZACION

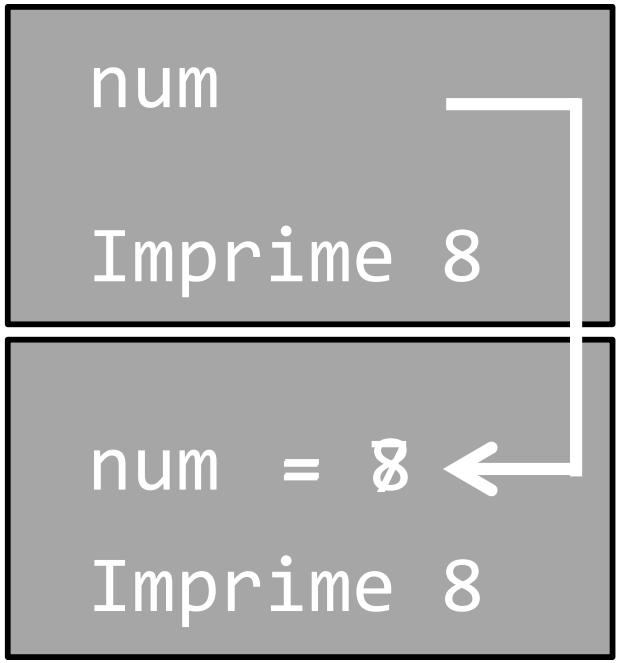
COMUNICACION



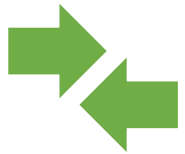
```
Program porReferencia;

procedure uno (var num: integer);
Begin
    if (num = 7)  then
    num:= num + 1;
    write (num);
end;
var
   num: integer;
begin
  num:= 7;
  uno (num);
end.
```

Procedimiento uno
Variables locales
Parámetros

Programa ppal
Variables globales
Variables de prog

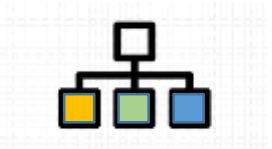


MEMORIA



- El número y tipo de los argumentos utilizados en la invocación a un módulo deben coincidir con el número y tipo de parámetros del encabezamiento del módulo.
- Un parámetro por valor debiera ser tratado como una variable de la cuál el módulo hace una copia y la utiliza localmente. Algunos lenguajes permiten la modificación local de un parámetro por valor, pero toda modificación realizada queda en el módulo en el cual el parámetro es utilizado.
- El número y tipo de los argumentos utilizados en la invocación a un módulo deben coincidir con el número y tipo de parámetros del encabezamiento del módulo.

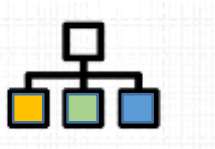
CADP – COMUNICACION ENTRE MODULOS



Implemente un programa que lea un valor entero e invoque a un módulo al cual le envía el valor leído y le devuelve si el valor recibido es múltiplo de 6.

- Cuántos y de qué tipo son los parámetros que recibe el módulo?
- Cuántos y de qué tipo son los parámetros que devuelve el módulo?
- Qué tipo de módulo utilizo?

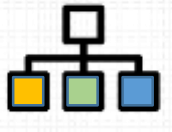
CADP – COMUNICACION ENTRE MODULOS



Implemente un programa que lea un valor entero e invoque a un módulo al cual le envía el valor leído y le devuelve si el valor recibido es múltiplo de 6.

```
procedure multiplo (num:integer; var esmultiplo:boolean);

begin
 if (num MOD 6 = 0) then
  esmultiplo:= true
 else
  esmultiplo:= false;
end;

procedure multiplo (num:integer; var esmultiplo:boolean);

begin
  esmultiplo:= (num MOD 6 = 0);
end;
```

CADP – COMUNICACION ENTRE MODULOS



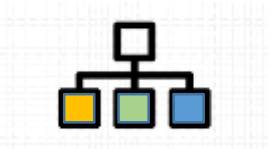
Implemente un programa que lea un valor entero e invoque a un módulo al cual le envía el valor leído y le devuelve si el valor recibido es múltiplo de 6.

```
function multiplo (num:integer):boolean;
var
 res:boolean;
begin
 if (num MOD 6 = 0) then
  res:= true
 else
  res:= false;
 multiplo:= res;
end;
```

```
function multiplo (num:integer):boolean;
  begin
   if (num MOD 6 = 0) then
    multiplo:= true
   else
    multiplo:= false;
  end;
```

```
function multiplo (num:integer):boolean;

begin
  multiplo:= (num MOD 6 = 0);
end;
```

CADP – COMUNICACION ENTRE MODULOS

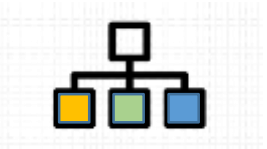


Implemente un programa que lea un valor entero e invoque a un módulo al cual le envía el valor leído y le devuelve si el valor recibido es múltiplo de 6.

```
program uno;
 procedure multiplo (num:integer;
                     var esmultiplo:boolean);

 begin
   esmultiplo:= (num MOD 6 = 0);
 end;

var
 valor:integer;
 res:boolean;
begin
  read(valor);
  multiplo (valor,res);
  write (res);
end.
```

```
program uno;
 function multiplo (num:integer):boolean;

 begin
   multiplo:= (num MOD 6 = 0);
 end;

var
 valor:integer;
begin
  read(valor);
  write (multiplo (valor));
end.
```

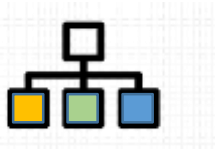
CADP – COMUNICACION ENTRE MODULOS



Implemente un programa que lea un valor entero e invoque a un módulo al cual le envía el valor leído y le devuelve cuántos dígitos pares y cuántos impares componen el numero recibido

- Cuántos y de qué tipo son los parámetros que recibe el módulo?
- Cuántos y de qué tipo son los parámetros que devuelve el módulo?
- Qué tipo de módulo utilizo?

CADP – COMUNICACION ENTRE MODULOS



Implemente un programa que lea un valor entero e invoque a un módulo al cual le envía el valor leído y le devuelve cuántos dígitos pares y cuántos impares componen el numero recibido

```
procedure contador (num:integer; var par:integer; var impar:integer);
var
 resto:integer;
begin
 par:= 0;
 impar:=0;
 while (num <> 0) do
   begin
     resto:= num MOD 10;
     if (resto MOD 2 = 0) then par:= par + 1
     else impar:= impar + 1;
     num:= num DIV 10;
   end;
 end;
```

num = 2681

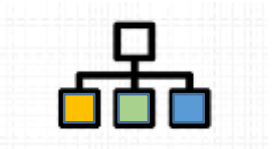
resto = 1 Impar = 1

resto = 8 Impar = 1
Par = 1

resto = 6 Impar = 1
Par = 2

resto = 2 Impar = 1
Par = 3

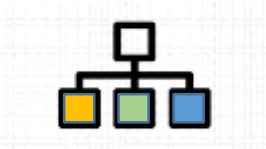
CADP – COMUNICACION ENTRE MODULOS



Implemente un programa que lea un valor entero e invoque a un módulo al cual le envía el valor leído y le devuelve cuántos dígitos pares y cuántos impares componen el numero recibido

```
program uno;
 procedure contador (num:integer; var par:integer; var impar:integer);
   var
   begin
      ....
   end;

var
 valor,pares,impares:integer;

begin
  read(valor);
  contador (valor,pares,impares);
  write (pares,impares);
end.
```

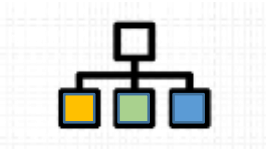
CADP – COMUNICACION ENTRE MODULOS



```
program uno;

 procedure uno (num:integer; var res:integer);

 begin
    ….
 end;


begin
  uno (8,4);

end.
```

Es correcta la invocación?

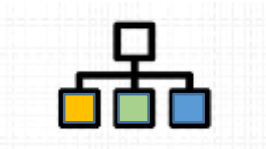
CADP – COMUNICACION ENTRE MODULOS



```
program uno;

 procedure uno (num:integer; var res:integer);

 begin
    ….
 end;

var
 x,pos:integer;
begin
  uno (8,pos);
  uno (x,pos);
end.
```

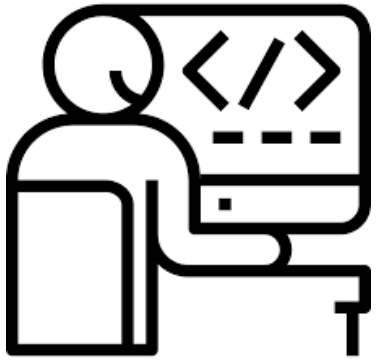
Es correcta la invocación?

CADP – COMUNICACION ENTRE MODULOS



```
program uno;
const
 x = 49;

 procedure uno (num:integer; var res:integer);

 begin
    ….
 end;

 begin
  uno (8,x);
 end.
```

Es correcta la invocación?

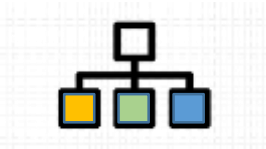


Conceptos de Algoritmos Datos y Programas

CADP – **TEMAS**

- Comunicación entre módulos

CADP – COMUNICACION ENTRE MODULOS



- VARIABLES GLOBALES
- PARAMETROS

CADP – COMUNICACION ENTRE MODULOS

VARIABLES GLOBALES

```
Program ejemplo1;
 Var
  x:integer;

 Procedure uno;
  Begin
    x:= x+1;
    write (x);
  End;
 Procedure dos;
  Begin
    x:= x MOD 10;
    write (x);
  End;
 var
  x: integer;

Begin
  x:=9;
  uno;
  write (x);
End.
```

Demasiados identificadores

No se especifica la comunicación entre los módulos

Conflictos de nombres de identificadores utilizados por diferentes programadores.

Posibilidad de perder integridad de los datos, al modificar involuntariamente en un módulo datos de alguna variable que luego deberá utilizar otro módulo.

CADP – COMUNICACION ENTRE MODULOS

PARAMETROS

La solución a estos problemas ocasionados por el uso de variables globales es una combinación de ocultamiento de datos (Data Hiding) y uso de parámetros.

El ocultamiento de datos significa que los datos exclusivos de un módulo NO deben ser "visibles" o utilizables por los demás módulos.

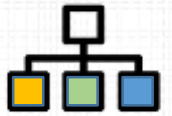
El uso de parámetros significa que los datos compartidos se deben especificar como parámetros que se trasmiten entre módulos.

CADP – COMUNICACION ENTRE MODULOS

PARAMETROS – Cómo vamos a trabajar?

1. Se analiza para cada módulo entonces: ¿cuáles son los datos propios? y ¿cuáles son los datos compartidos?
2. Los datos propios se declararan locales al módulo.
3. Los datos compartidos se declararán como parámetros.
   - Parámetros por valor
   - Parámetros por referencia

CADP – COMUNICACION ENTRE MODULOS



PARAMETROS

VALOR

- El módulo recibe un valor, puede realizar operaciones y/o cálculos, pero no producirá ningún cambio ni tampoco tendrá incidencia fuera del módulo.

REFERENCIA

- El módulo recibe una dirección, puede realizar operaciones y/o cálculos, que producirán cambios y tendrán incidencia fuera del módulo.

CADP – COMUNICACION ENTRE MODULOS

PARAMETROS

```
Program ejemplo2;

 Procedure uno (PARAMETRO1; PARAMETRO2);
  Begin
    …
  End;
 Procedure dos (PARAMETRO);
  Begin
    …
  End;
 var
  x,y,z: integer;

Begin
  ….
  uno(x,y);
  dos(z);
  …
End.
```

Cada módulo indica que necesita recibir

Cada módulo indica que devuelve

No existe el problema donde un se pueda modificar el valor sin darse cuenta.