ALGORITMIA Y PROGRAMACIÓN

ESTRUCTURAS ITERATIVAS

Universidad del Valle

Contenido

Estructuras de Repetición

- for
- Ciclos anidados
- Contadores y acumuladores
- Ejercicios

Estructuras de Repetición

- Ejemplo: Suponga que debe mostrar los números del 1 a 10.

Estructuras de Repetición

Ejemplo: Suponga que debe mostrar los números del 1 a 10.

Se puede hacer con la siguiente función:

```
Inicio mostrarNumeros()
   imprimir("El número es: 1")
   imprimir("El número es: 2")
   imprimir("El número es: 3")
   imprimir("El número es: 4")
   imprimir("El número es: 5")
   imprimir("El número es: 6")
   imprimir("El número es: 7")
   imprimir("El número es: 8")
   imprimir("El número es: 9")
   imprimir("El número es: 10")
Fin
```

Pseudocódigo

```
def  mostrarNumeros():

    print ("El número es: 1")
    print ("El número es: 2")
    print ("El número es: 3")
    print ("El número es: 4")
    print ("El número es: 5")
    print ("El número es: 6")
    print ("El número es: 7")
    print ("El número es: 8")
    print ("El número es: 9")
    print ("El número es: 10")
```

Python

¿ Y si se quieren mostrar los números del 1 al 5000?

Estructuras de Repetición

Son un grupo de instrucciones que permite la ejecución repetitiva de otro grupo de instrucciones. Hay una variable asociada al **ciclo** o **estructura de repetición** que controla el número de veces que se repetirán las instrucciones.

Existen 2 estructuras de repetición:

Para → for
mientras que → while

Estructuras de Repetición

Problema: Desarrollar un programa que solicite un número y muestre los números desde el 1 hasta el número solicitado.

1. Análisis del problema

Entradas: ??
Salidas: ??
Proceso:??

Estructuras de Repetición

Problema: Desarrollar un programa que solicite un número y muestre los números desde el 1 hasta el número solicitado.

1. Análisis del problema

Entradas: n
Salidas: ??

Proceso:??

Estructuras de Repetición

Problema: Desarrollar un programa que solicite un número y muestre los números desde el 1 hasta el número solicitado.

1. Análisis del problema

Entradas: n
Salidas:

imprimir (número 1)
imprimir (número 2)
.
.
imprimir (número n)

Proceso:??

Estructuras de Repetición

Problema: Desarrollar un programa que solicite un número y muestre los números desde el 1 hasta el número solicitado.

1. Análisis del problema

Entradas: n
Salidas:
    imprimir (número 1)
    imprimir (número 2)
    .
    .
    imprimir (número n)
Proceso:
    desde 1 hasta n

Estructuras de Repetición

Problema: Desarrollar un programa que solicite un número y muestre los números desde el 1 hasta el número solicitado.

1. Análisis del problema

Entradas: n
Salidas:
imprimir (número 1)
imprimir (número 2)
.
.
imprimir (número n)
Proceso:
desde 1 hasta n
Imprimir 1,2,3,.....n

Estructuras de Repetición

Problema: Desarrollar un programa que solicite un número y muestre los números desde el 1 hasta el número solicitado.

2. Diseñar el algoritmo y escribirlo en pseudocódigo

```
Inicio
   n: entero

   n= leer ("Digite un número")

   ???



Fin
```

Estructuras de Repetición

Problema: Desarrollar un programa que solicite un número y muestre los números desde el 1 hasta el número solicitado.

2. Diseñar el algoritmo y escribirlo en pseudocódigo

```
Inicio
   n: entero
   i: entero  (Variable de control)

   n= leer ("Digite un número")

   Para  i = 1  hasta n;   incrementar i en 1
   Haga
              Imprimir (i)
   Fin_para

Fin
```

Estructuras de Repetición

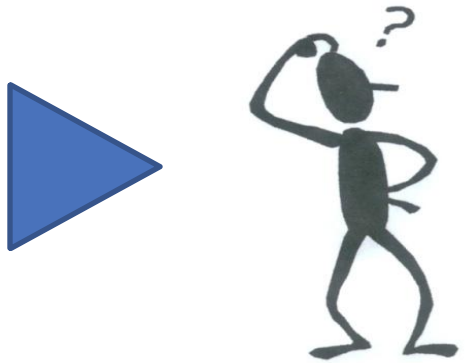
Problema: Desarrollar un programa que solicite un número y muestre los números desde el 1 hasta el número solicitado.

3. Codificar el algoritmo usando algún lenguaje de programación

pseudocódigo

```
Para  i = 1  hasta n;   incrementar i en 1
Haga
            Imprimir (i)
fin_para
```

Python



Estructuras de Repetición

Problema: Desarrollar un programa que solicite un número y muestre los números desde el 1 hasta el número solicitado.

3. Codificar el algoritmo usando algún lenguaje de programación

pseudocódigo

```
Para  i = 1  hasta n;   incrementar i en 1
Haga
                Imprimir (i)
fin_para
```

Python

Estructura De Repetición for

Estructuras de Repetición

Se usa **para** repetir una instrucción o un conjunto de instrucciones, **Desde** un inicio, **Mientras** una condición se cumpla y con un **incremento** o **decremento**.

Un ciclo **for** tiene tres partes:

Expresión de inicio,
Expresión de condición y
Expresión de incremento/decremento.

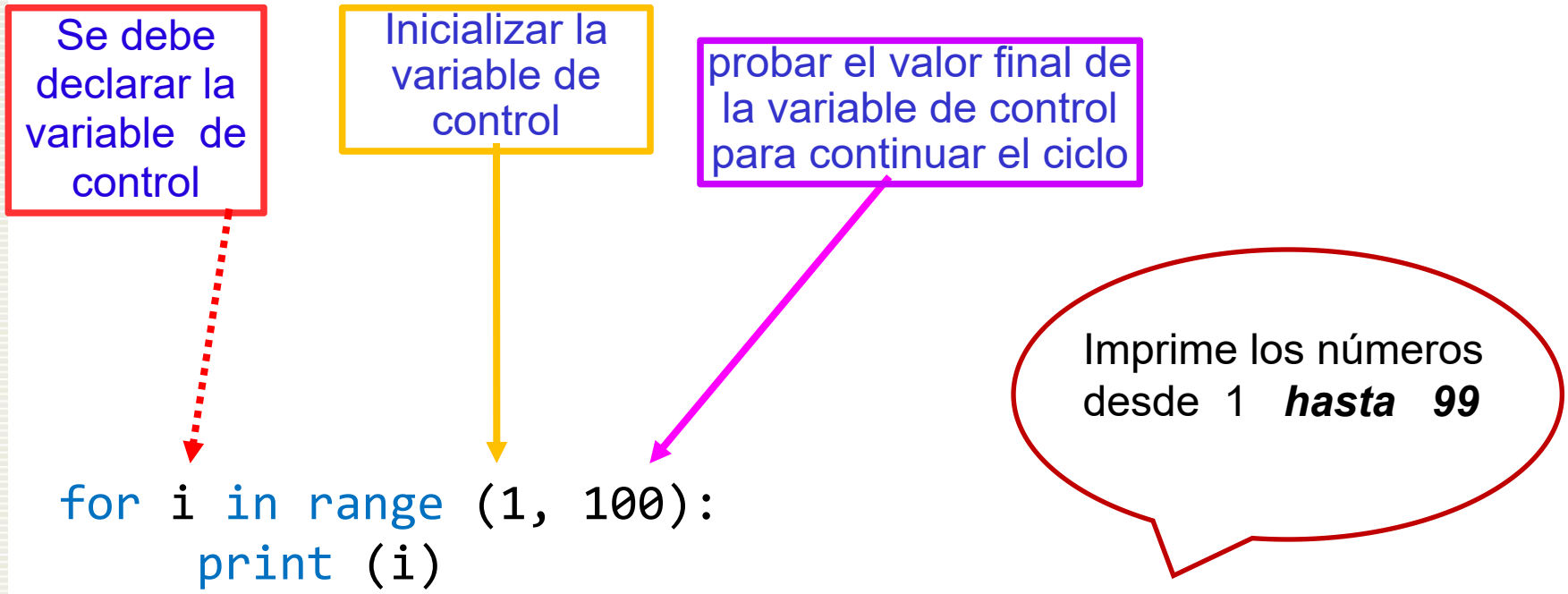
```
Para i = 1 hasta n; incrementar i en 1
Haga
                Imprimir (i)
Fin_para
```

```
for i in range (1,n+1):

   print (i)
```

Estructuras de Repetición

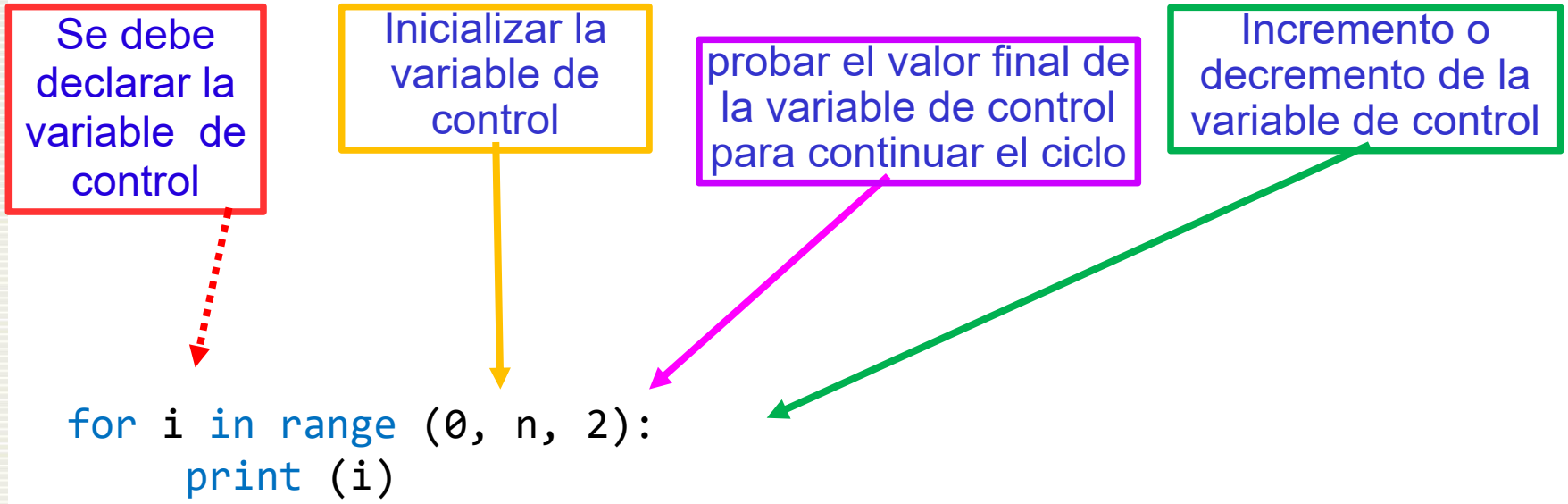
Estructura de la instrucción for en Pytho



Imprimer los valores hasta que sean menores que 100.
Por defecto el incremento es de **1 en 1**

Estructuras de Repetición

Estructura de la instrucción for en Python



En este caso el incremento será **de 2 en 2.**
Si el valor inicial es omitido, se supone que es 0. Si el incremento es omitido, se supone que es 1.
El valor inicial siempre es parte del rango. El valor final nunca es parte del rango.

Estructura for

Un ciclo for siempre debe especificar tres partes:

- Expresión de inicio
- Expresión de condición (o prueba)
- Expresión de incremento/decremento

```
for i in range (0,n):
    print (i)
```

```
for i in range (0,n,2):
    print (i)
```

Estructura for

Variable de control: Es la variable que se utiliza para contar la cantidad de iteraciones realizadas y **es usada en la condición** que determina el límite de repeticiones a realizar.

Ejemplo:

```
for i in range (0,n):
    print (i)
```

i es la variable de control

Estructura for

Qué imprimen estas funciones?

```
def ejemplo1():
    n=100
    for i in  range (0, (n+1), 1):
        print (i)
```

```
def ejemplo2():
    n=0
    for i in  range (100, (n-1), -1):
        print (i)
```

Estructura for

- Ejemplo:

```
def listarNumeros():
        for i in  range (0, 6):
          print (“Número”,i)
```

Valor de i	Número Iteración	Salida
0	1	“Número 0”
1	2	“Número 1”
2	3	“Número 2”
3	4	“Número 3”
4	5	“Número 4”
5	6	“Número 5”
Se interrumpe		

El **número de iteraciones** indica la cantidad de veces que se repite la(s) instrucción(es) que estén en el cuerpo del ciclo.

En el ejemplo, la instrucción:
print (“Número”,i)
se ejecuta 6 veces.

Estructura for

- Ejemplo:

```
def cicloPrueba():
    for i in range (18, 10, -2):
        print (i)
```

- Cuántas veces se ejecutará la instrucción: ***print (i)***
- Qué valores imprime?

Estructura for

- Ejemplo:

```
def cicloPrueba():
    for i  in range (18, 10, -2):
        print (i)
```

Valor de i	Número Iteración	Salida
18	1	18
16	2	16
14	3	14
12	4	12
10	**Se interrumpe**	

La instrucción:
print (i)
se ejecuta 4 veces

Estructura for

- Ejemplo:

```
def muestraAlgo():
    for i  in  range (0, 20):
        if (i%2 == 0):
            print ("El valor de i  es ", i)
```

- Cuantas veces se ejecutará la instrucción:

```
print ("El valor de i  es ", i)
```

Estructura for

- Ejemplo:

```
def muestraAlgo2( ):
    for j in range (0,101):
            if ((j % 3) == 0):
                print (“El valor de J es“, j)
            else:
                print (j,“NO es ..." )
```

- Que debo completar en el mensaje **“NO es ...“ ?**

```
print (j,“NO es ..." )
```

Estructura for

Una vez el ciclo se interrumpe se ejecuta la instrucción ubicada después de él. Por ejemplo:

```
a=0

for k in range (0,5,2):
      a+=1

print (“El valor de a es: ” , a)
```

Estructura for

Una vez el ciclo se interrumpe se ejecuta la instrucción ubicada después de él. Por ejemplo:

```
a=0

for k in range (0,5,2) :
      a+=1

print ("El valor de a es: " + a)
```

- La variable ***a*** inicia con el valor de **cero** (0).
- Se ingresa al ciclo y se ejecuta tres (3) veces la instrucción **a+=1**
- Luego se muestra en pantalla: El valor de **a** es: **3**

Estructura for

- **Ejercicio 1:**

Desarrollar un programa Python que pregunte al usuario el número de estudiantes de un curso, luego pregunte el nombre de cada uno de ellos. Finalmente, se debe mostrar un listado con todos los estudiantes.

```
>>>
Digite la cantidad de estudiantes: 3
Digite el nombre del estudiante 1: Maria
Digite el nombre del estudiante 2: Pablo
Digite el nombre del estudiante 3: Carlos
Los nombres de los 3 estudiantes son:
Maria
Pablo
Carlos

>>>
```

Estructuras de Repetición

Problema: Desarrollar un programa Python que pregunte al usuario el número de estudiantes de un curso, luego pregunte el nombre de cada uno de ellos. Finalmente, se debe mostrar un listado con todos los estudiantes.

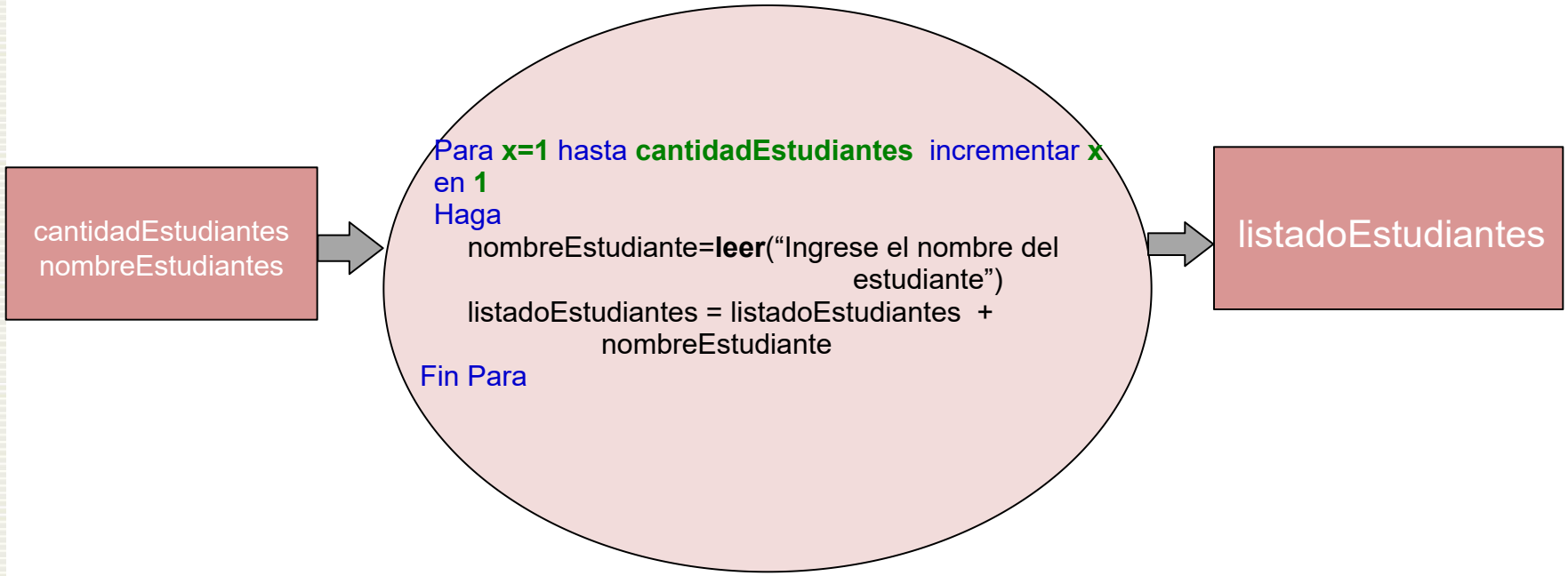
1. Análisis del problema

Entradas: cantidadEstudiantes, nombreEstudiante,
Salidas: listadoEstudiantes

Proceso:??

Estructuras de Repetición

Problema: Desarrollar un programa Python que pregunte al usuario el número de estudiantes de un curso, luego pregunte el nombre de cada uno de ellos. Finalmente, se debe mostrar un listado con todos los estudiantes.



Estructuras de Repetición

Problema: Desarrollar un programa Python que pregunte al usuario el número de estudiantes de un curso, luego pregunte el nombre de cada uno de ellos. Finalmente, se debe mostrar un listado con todos los estudiantes.

1. Análisis del problema

Entradas: cantidadEstudiantes, nombreEstudiante,
Salidas: listadoEstudiantes

Proceso:

```
Para x=1 hasta cantidadEstudiantes  incrementar x en 1
Haga
    nombreEstudiante=leer("Ingrese el nombre del estudiante")
    listadoEstudiantes  = listadoEstudiantes  + nombreEstudiante
 Fin Para
```

Estructura for

- **Ejercicio 1:**

2. Algoritmo en pseudocódigo.

```
Inicio
    cantidadEstudiantes: entero
    listadoEstudiantes:   texto
    nombreEstudiante:   texto

    cantidadEstudiantes=  leer(Ingrese la cantidad de estudiantes)
      Para x=1 hasta cantidadEstudiantes  incrementar x en 1
        Haga
          nombreEstudiante=leer("Ingrese el nombre del estudiante")
          listadoEstudiantes  = listadoEstudiantes  + nombreEstudiante
       Fin Para
       imprimir(listadoEstudiantes )
Fin
```

Estructura for

- **Ejercicio 1**:

3. Función en Python

```
#lee una lista de nombres de estudiantes y luego los muestra

cantidadEstudiantes = int(input("Ingrese la cantidad de estudiantes: "))
listadoEstudiantes = ""

for i in range(1,cantidadEstudiantes+1):
        nombreEstudiante = input("Ingrese el nombre del estudiante "
                                 +str(i)+ " ")
        listadoEstudiantes = listadoEstudiantes + nombreEstudiante + "\n"

print("Los estudiantes son: \n" + listadoEstudiantes)
```

Estructura for

- Ejercicio 2 :

Suponga que se desea crear un programa en Python que permita imprimir en pantalla el factorial de un número el cual es digitado por un usuario.

El factorial de un número n es la multiplicación de todos los números de la serie 1 a n, entonces, el factorial de 5 es 1 x 2 x 3 x 4 x 5 = 120, el factorial de 3 es 1 x 2 x 3 = 6.

```
Para i=1 hasta n incrementar i en 1
Haga


      instrucción  3
      instrucción  4
Fin Para
```

Estructura for

2. Algoritmo en pseudocódigo

Inicio

factorial,n: entero
respuesta: texto
leer(n)

```
para  i=1  hasta n incrementar i en 1
haga
  factorial=factorial*i
fin para
respuesta=“el factorial de “ + n + ”es:” factorial
imprima (respuesta)
```

fin

Estructura for

3. Función en Python

```
def factorial(n):
    factorial = 1
    for i in range(1,n+1):
        factorial = factorial * i
    respuesta="El factorial de " ,n , "es: " , factorial
    print(respuesta)
```

Estructura for

3. Prueba de escritorio

n	factorial	i	i<n+1	salida
3	1	1	1<4 verdadero	
	1x1=1	2	2<4 verdadero	
	1x2=2	3	3<4 verdadero	
	2x3=6	4	4<4 falso	el factorial de 3 es 6

Estructura for – Ciclos Anidados

- Ejercicio 3:

Suponga que se desea crear un programa en Python que permita imprimir en pantalla el factorial de cada número existente en la serie 1 a n, siendo n un número menor a 20 el cual es digitado por un usuario.

```
Para i=1 hasta n incrementar i en 1
Haga
        Para j=1 hasta i  incrementar  j en  1
        Haga
        instrucción 1
        instrucción 2
        …
         Fin Para
    instrucción  4
Fin Para
```

NOTA: Usamos la condición n<=20 con fines académicos y evitar resultados con números exageradamente grandes.

Estructura for – Ciclos Anidados

Un **ciclo anidado** es un ciclo, de cualquier tipo, que contiene otro ciclo de cualquier tipo.

En el ejercicio propuesto se observa que se debe crear un ciclo para manejar la serie de 1 a n, y un ciclo anidado para calcular el factorial de n_i.

Por ejemplo: si el usuario digita el número 5 el sistema debe recorrer la serie 1 a 5, y para cada número devolver su factorial, así:

Factorial de 1 = 1, Factorial de 2 = 2, Factorial de 3 = 6 , Factorial de 4 = 24 y el Factorial de 5 = 120.

Estructura for – Ciclos Anidados

2. Algoritmo en pseudocódigo.

```
Inicio
    factorial, n: entero
    respuesta: texto
    leer(n)
    SI  n <= 20 Entonces
        Desde i=1 hasta n incrementar i en 1
        Haga
                factorial=1
                Desde j=1 hasta i incrementar j en 1
                Haga
                        factorial = factorial*j
                Fin Para
                respuesta = respuesta + "El factorial de " + j + "es: " factorial;
        Fin Desde
    imprima(respuesta)
    SINO
        imprima("Ha digitado un numero superior a 20")
Fin
```

Estructura for – Ciclos Anidados

2. Algoritmo en pseudocódigo.

```
Inicio
    factorial, n: entero
    respuesta: texto
    leer(n)
    SI  n <= 20 Entonces
        Desde i=1 hasta n incrementar i en 1
        Haga
                factorial=1
                Desde j=1 hasta i incrementar j en 1
                Haga
                        factorial = factorial*j
                Fin Para
                respuesta = respuesta + "El factorial de " + j + "es: " factorial;
        Fin Desde
    imprima(respuesta)
    SINO
        imprima("Ha digitado un numero superior a 20")
Fin
```

El ciclo principal, basado en la variable de control **i**, contiene un ciclo anidado basado en la variable de control **j.**

Estructura for – Ciclos Anidados

3. Función en Python

```
def listaFactorial(n):
        if(n<=20):
                for i in range(1,n+1):
                        factorial = 1
                        for j in range(1,i+1):
                                factorial *= j
                        respuesta="El factorial de", i ,"es:",factorial
                        print(respuesta)
        else:
                respuesta="Ha digitado un número superior a 20"
                print(respuesta)
```

Estructura for – Ciclos Anidados

- Ejercicio 4:

  Desarrollar un programa Python que permita generar y visualizar la siguiente figura:

```
*
* *
* * *
* * * *
```

- Tenga en cuenta que el usuario ingresa al programa el número de líneas que debe tener la figura (En el ejemplo hay 4 filas).

Estructura for – Ciclos Anidados

- Ejercicio 4:

```
crearFigura(numeroFilas : entero)
 Inicio
      triangulo = " " : texto
      Para i=1 hasta numeroFilas incrementar i en 1
      Haga
              Para j=1 hasta i incrementar j en 1
               Haga
                 triangulo = triangulo  + "*"
               Fin Para
       triangulo = triangulo  + "\n"
      Fin Para
      imprimir(triangulo)
 Fin
```

- Prueba de escritorio crearFigura:

numeroFilas	triangulo	i	i<=numeroFilas	j	j<=i	salida
3	“ ”	1	1<=3v	1	j<=1 v	*
	*			2	2<=1 f	
		2	2<=3 v	1	1<=2 v	
	* *|			2	2<=2 v	
	* * *			3	3<=2 f	
		3	3<= 3v	1	1<=3 v	
	* * * *			2	2<=3 v	
	* * * * *			3	3<=3 v	
	* * * * * *			4	4<=3 f	* * * * * *
		4	4<=3 falso			

Estructura for – Ciclos Anidados

- Ejercicio 4:

```
def crearFigura(numeroFilas):
        triangulo = ""
        for i in range(1,numeroFilas+1):
                for j in range(1,i+1):
                        triangulo += "*"
                triangulo += "\n"
        print(triangulo)
```

Estructura for - Contadores

Los contadores son variables utilizadas para realizar, como su nombre lo indica, conteos de la cantidad de veces que se cumple una situación específica.

Como su objetivo principal es contar, deben ser de tipo entero y normalmente se inicializan en cero.

Los contadores y los acumuladores pueden ser usados en cualquier tipo de ciclos.

Estructura for - Contadores

- Ejemplo:

  Suponga que se desea crear un programa en Python que permita imprimir en pantalla la cantidad de números múltiplos de 3 que se encuentran en la serie 1 a n, siendo n un número digitado por un usuario.

Estructura for - Contadores

- Ejemplo:

  Suponga que se desea crear un programa en Python que permita imprimir en pantalla la cantidad de números múltiplos de 3 que se encuentran en la serie 1 a n, siendo n un número digitado por un usuario.

```
Inicio
    contador:  entero;
    contador=0;
    leer(n)
    Para i=1 hasta n incrementar i en 1
    Haga
        if ((i%3)==0){
            contador=contador+1;
        }
    Fin Para
    imprimir(contador);
Fin
```

Estructura for - Contadores

- Ejemplo:

  Suponga que se desea crear un programa en Python que permita imprimir en pantalla la cantidad de números múltiplos de 3 que se encuentran en la serie 1 a n, siendo n un número digitado por un usuario.

```
Inicio
    contador:  entero;
    contador=0;
    leer(n)
    Para i=1 hasta n incrementar i en 1
    Haga
        if (esMultiploDeTres(i)){
            contador=contador+1;
        }
    Fin Para
    imprimir(contador)
 Fin
```

Estructura for - Contadores

- Ejemplo:

```
def esMultiploDeTres(a):
    if(a%3 == 0):
      return True
    else:
      return False


def multiploDeTres(n):
    contador=0

    for i in range(1,n+1):
      if(esMultiploDeTres(i)):
        contador = contador + 1

    print("La cantidad de números multiplos de 3 desde 1 hasta " + str(n) + " es "+ str(contador))
```

La variable ***contador*** se incrementa en **1** cada vez que el método ***esMultiploDeTres*** retorna **true** (cierto) para el número **i**

Estructura for - Acumuladores

Los acumuladores son variables utilizadas para ir almacenando (acumulando) el resultado de una operación.

Pueden ser de tipo numérico (entero o real) en las cuales acumula el resultado de operaciones matemáticas, o de tipo cadena en las cuales se concatenan frases o palabras.

En el ejemplo del factorial se pueden observar las variables ***factorial*** y ***respuesta*** las cuales actúan como acumuladores numéricos y de cadena respectivamente.

Estructura For - Acumuladores

```
def listaFactorial(n):
        if(n<=20):
                for i in range(1,n+1):
                        factorial = 1
                        for j in range(1,i+1):
                                factorial *= j
                        respuesta="El factorial de", i ,"es:",factorial
                        print(respuesta)
        else:
                respuesta="Ha digitado un número superior a 20"
                print(respuesta)
```

La variable ***factorial*** va acumulando el valor de la multiplicación y la variable ***respuesta*** va acumulando las cadenas usadas como respuesta en cada iteración

Ejercicios Estructura for

1. Diseñe un algoritmo que permita detectar los números pares existentes en una serie de 1 a n, siendo n un número digitado por un usuario.
2. Diseñe un algoritmo que permita obtener la suma de todos los números enteros existentes en una serie de 1 a n y la cantidad de números pares encontrados, siendo n un número digitado por un usuario. Use un ciclo **for** en su diseño.
3. Suponga que se desea saber la nota promedio del curso de algoritmia, diseñe un algoritmo que solicite la cantidad de estudiantes del curso y el promedio de cada estudiante.

NOTA: **Para cada ejercicio realice su respectiva implementación en Python**

Ejercicios Estructura for

4. Suponga que el calculo de la pensión de una persona se realiza de la siguiente manera: por cada año de servicio se paga $80 si el empleado ingresó en o después de 1995 y $100 si ingresó antes, dicho valor (80 o 100) se multiplica por el número de cada año más la edad que tenía en el año (ej. (100*1994+32)+(100*1995+33)+…), el descuento de seguridad social en salud es del 12%. El programa debe recibir el año de ingreso y la edad del empleado en el año de ingreso, devolver el sueldo o mesada bruta, la mesada neta y el valor del descuento por salud.

   Ejemplo: Para una persona que ingresó en el 2009 y que tenía 44 años en dicho año, su mesada o sueldo bruto para el 2011 es $482.535, el descuento por salud es $57.904 y por lo tanto su sueldo o mesada neta es $424.630.

NOTA: **Realice la respectiva implementación en Python.**

Ejercicios Estructura for

5. Crear una aplicación que permita:
   - Generar los números enteros pares entre p y q
   - Generar los números enteros impares entre a y b
   - Generar los primeros z múltiplos de 3
   - Generar la suma de m primeros múltiplos de 7 más los n primeros múltiplos de 9

NOTA: **Realice la respectiva implementación en Python.**