

Inspira Crea Transforma

Programación de Computadores

ST0240

Semestre 2017-1

48 horas semestral

Fundamentos de Programación

ST0286

Semestre 2017-1

48 horas semestral

Agenda

2 clases por Semana

15m Contenido Previo (Tareas, Lecturas, Ejercicios propuestos o Anterior)

30m Contenido de Clase

30m Ejercicios Prácticos

15m Cierre: Trabajo Individual (Por fuera de Clase)

1 clase por Semana

30 m Contenido Previo (Tareas, Lecturas, Ejercicios propuestos o Anterior)

50 m Contenido de Clase

20 m Break

50 m Ejercicios Prácticos

30 m Cierre: Trabajo Individual (Por fuera de Clase)

Agenda

30 m Contenido Previo (Tareas, Lecturas, Ejercicios propuestos o Anterior)

50 m Condicionales, Ciclos

20 m Break

60 m Ejercicios Prácticos y Taller

10 m Cierre: Trabajo Individual (Por fuera de Clase)

Monitores

Nombre	Correo	Horario
Marcos Sierra	@eafit.edu.co	Lunes 1 p.m. - 5 p.m. Martes 3 p.m. - 6 p.m. Lugar:
PENDIENTE	@eafit.edu.co	pendiente

- Pendiente para la Tercera Semana*
- Monitor en Clase
 - Preguntas cortas con cita previa por correo electrónico
 - Horarios de Apoyo

Entrada y Salida Básica

	Python 2.7	Python 3	Ejemplos
Entrada	input()	raw_input()	nombre=input("Ingrese su nombre:") edad=int(input("Digite su edad:"))
Salida	print()	print()	print(nombre) print("Su edad es:",edad,"años")

Indentación

```
70. # Function checks if ball has hit paddle
71. def checkHitBall(ball, paddle1, paddle2, ballDirX):
72.     if ballDirX == -1 and paddle1.right == ball.left and
paddle1.top < ball.top and paddle1.bottom > ball.bottom:
73.         return -1 # return new direction (right)
74.     elif ballDirX == 1 and paddle2.left == ball.right and
paddle2.top < ball.top and paddle2.bottom > ball.bottom:
75.         return -1 # return new direction (right)
76.     else:
77.         return 1 # return new direction (left)
78.
```

En python, para reconocer el ámbito en el que nos encontramos, se utiliza el indentado, espacios o **tabulaciones** que crean jerarquías y así podemos identificar a qué parte del código pertenece una expresión

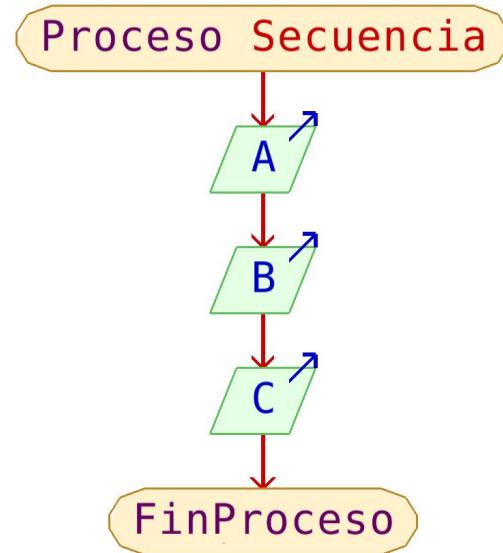
Estructuras básicas de programación

Cualquier programa, sin importar cuán complicado sea, puede construirse usando una o más de solo *tres estructuras.*

- Secuencia
- Selección o Decisión
- Ciclo

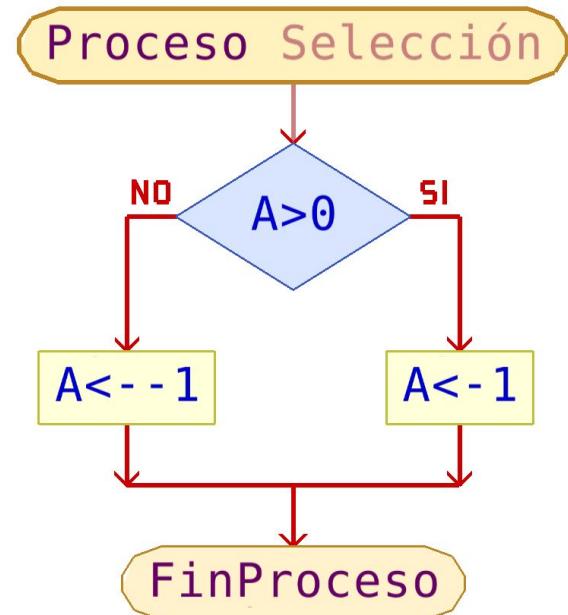
Secuencia

- Indica que las instrucciones de un programa se ejecutan una después de la otra, en el mismo orden en que aparecen en el programa.



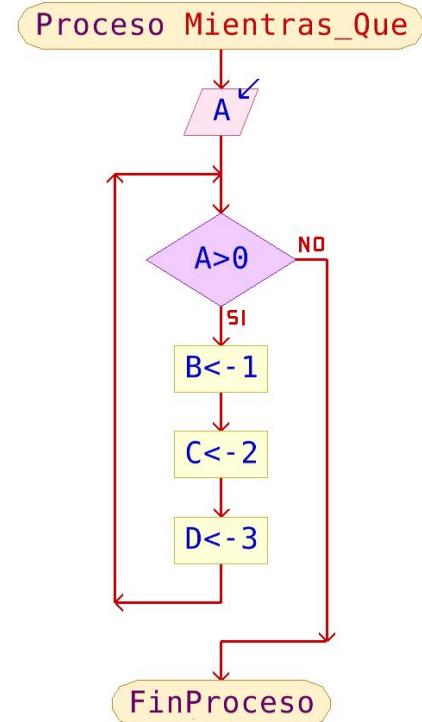
Selección o Decisión

- También se conoce como SI-VERDADERO-FALSO
- Plantea la selección entre dos alternativas con base en el resultado de la evaluación de la condición.
- En lenguajes de programación se conoce como IF.



Ciclo

- Corresponde a la ejecución repetida de una instrucción mientras que se cumple una determinada condición.
- También se conoce como HACER-MIENTRAS-QUE o For



Decisiones - Definición

La estructura condicional o de decisión simple (***if-else***) permite tomar decisiones de acuerdo a una o más condiciones con respuestas: Sí / No.

Por ejemplo:

La pregunta ¿El número es par?, sólo tiene dos posibles respuestas: Sí / No.

Así mismo: ¿Es mayor de edad?, ¿Es estudiante?, etc.

Las preguntas de este tipo se conocen como **condiciones**.

Decisiones - Sintaxis

if condiciones:

→ instrucciones a ejecutar si **se cumplen las condiciones**

else:

→ instrucciones a ejecutar si **no se cumplen las condiciones**

Decisiones - Expresiones lógicas

Las decisiones se basan en expresiones lógicas. Cuando hay varias condiciones, éstas se pueden unir con los operadores lógicos **and** y **or**. Por ejemplo:

```
if (x>y) and (x>0):
```

```
if (x=='A') or (x=='B') or (x=='C'):
```

Decisiones - Expresiones lógicas

Indique la expresión lógica para determinar si dada la edad de una persona, ésta es un niño (considere esta etapa entre los 2 y los 10 años)

Indique la expresión lógica para determinar si un pH dado es incorrecto (el valor de un pH oscila entre 0 y 14)

Decisiones - Ejemplo

Sistema de calculo costo de viajes por diferentes medios de transporte:

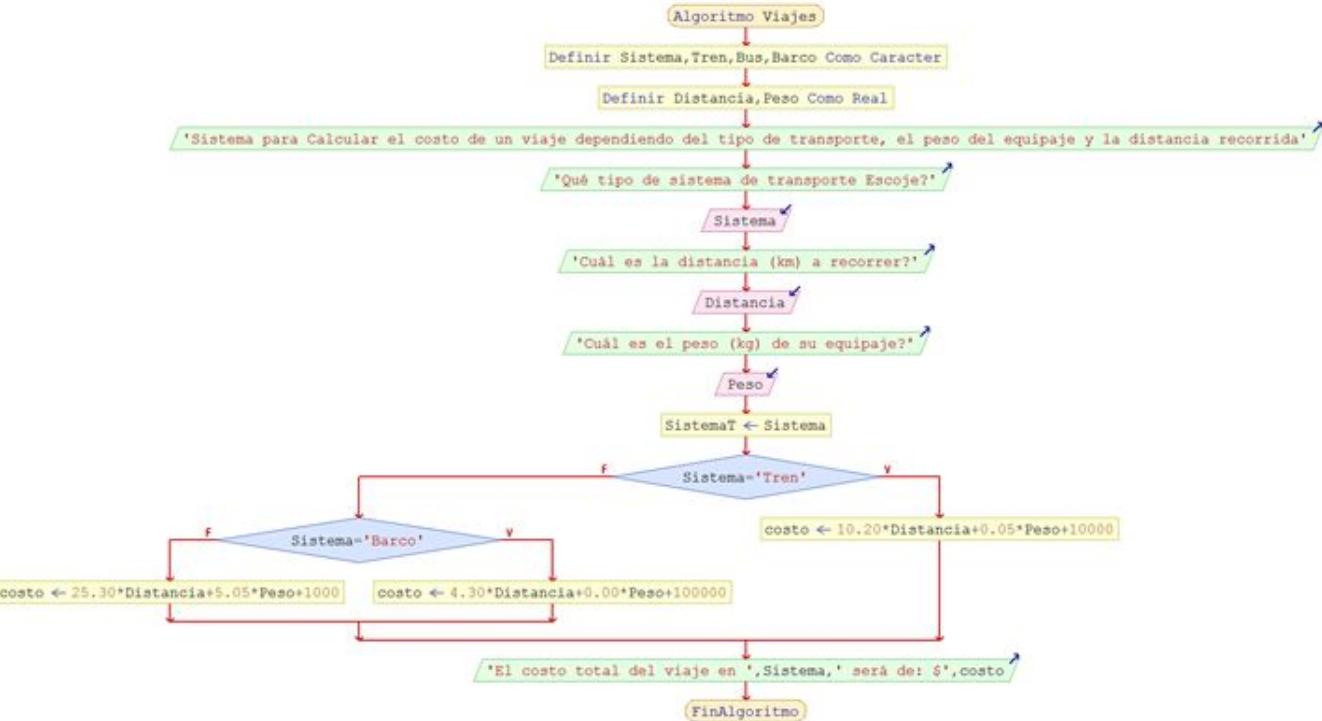
Usted quiere hacer un viaje y necesita evaluar los costos en tres sistemas de transporte: Tren, Bus, Barco. Por lo tanto, usted elaboró un sistema que calcula el costo con respecto a la longitud del viaje, el numero de kilos que llevará y el tipo de sistema que utilizara al viajar.

Sistema	$f_{distancia}$	Distancia (km)	$f_{equipaje}$	Peso (kg)	f_{costo_fijo}
Tren	10,20	1250,3	2,2	75	5.000
Bus	25,30	1420,8	5,0	50	1.000
Barco	4,30	1050,5	1,8	75	10.000

La fórmula para calcular cada sistema:

$$\text{Costo} = f_{distancia} \times distancia + f_{equipaje} \times peso + f_{costofijo}$$

Diagrama de Flujo Propuesto (una aproximación...)



Ciclos

Los Computadores son herramientas rápidas, debidamente usadas son eficientes, no se cansan, no se aburren y no les importa realizar el mismo trabajo una y otra vez, miles y millones de veces segundo a segundo.

Si deseamos realizar la misma tarea como tal vez calcular el promedio de las notas de muchos estudiantes de una universidad, no tendría mucho sentido que hiciéramos la misma operación una vez tras otra.

Para ello en programación tenemos los ciclos, los cuales son estructuras de control que nos permiten realizar la misma acción mientras se cumpla una condición explícita (p.ej. Calcular el salario de 10 empleados) o implícita (hasta que se pulse la tecla Escape).

El Castigo de Gauss

Cuando tan sólo tenía 7 años y asistía a primaria, el profesor de Gauss mandó a sus alumnos que sumaran los **números del 1 al 100** (en otros lugares he leído que este fue el castigo que le impuso a Gauss). Aquel día el profesor no tenía ganas de trabajar, se llevó un libro a clase, y les mandó esta tarea a sus alumnos. Al fin y al cabo, a él le había llevado varias horas hacer esta suma ...

Tomado de: <http://antropicos.blogspot.com.co/2007/03/la-leyenda-de-gauss.html>



Carl Friederich Gauss: “El principio de las matemáticas”

1	2	3	...	98	99	100
100	99	98		3	2	1
101	101	101	...	101	101	101

5050

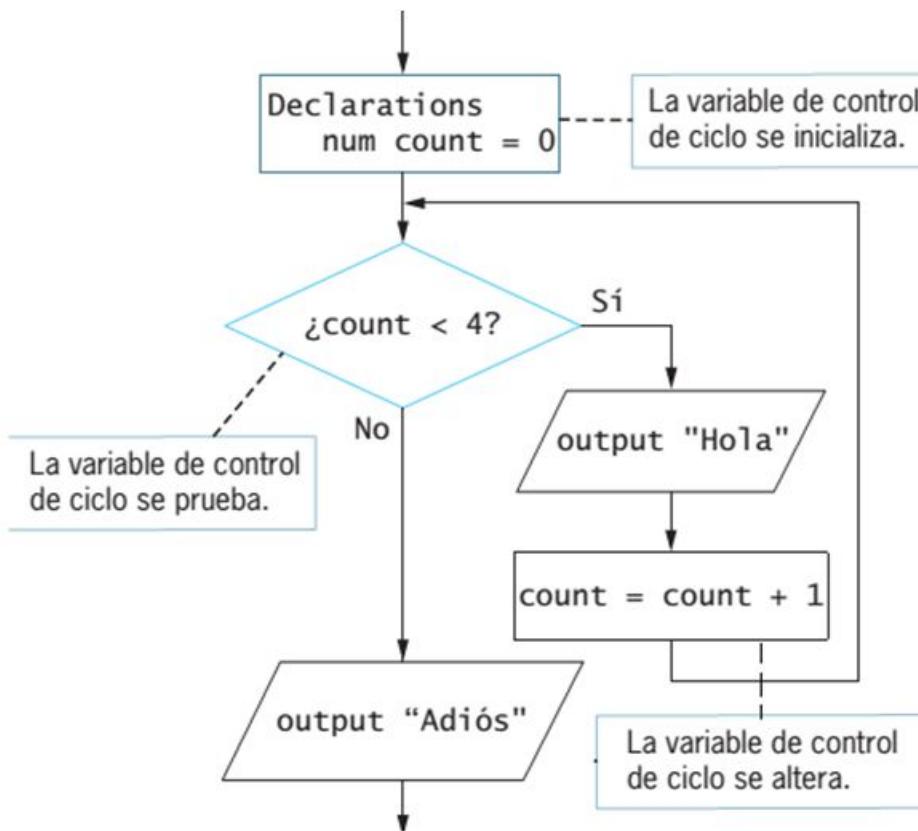
Inspira Crea Transforma

UNIVERSIDAD
EAFIT[®]

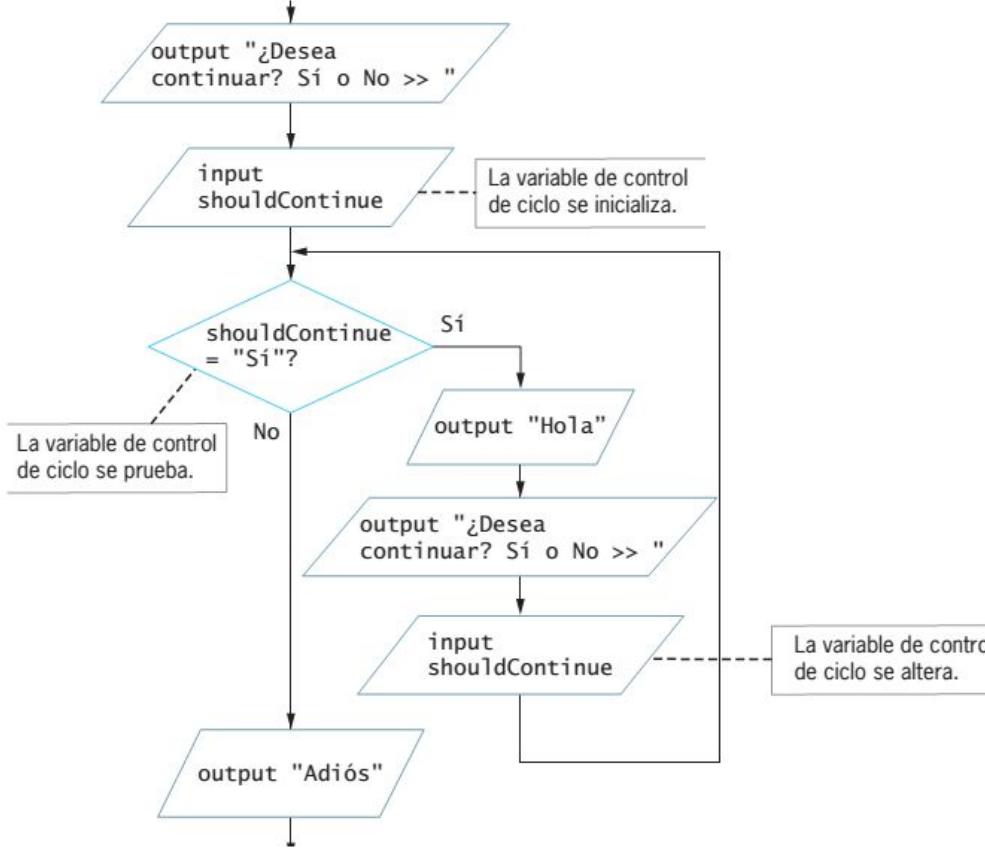
Ciclos - Variable de control

Se recomienda tener una variable que controle el número de repeticiones. Para evitar un “ciclo infinito”, con esta variable se debe hacer lo siguiente:

- Darle un valor inicial antes de entrar al ciclo.
- Probarla en una condición, p.ej., si es verdadera se ingresa al ciclo.
- Modificarla dentro del ciclo para que la condición sea falsa alguna vez.



Ejemplo tomado del libro Introducción a la Programación. Lógica y Diseño. JOYCE FARRELL. P172



Ejemplo tomado del libro Introducción a la Programación. Lógica y Diseño. JOYCE FARRELL. P174

Ciclos - while

El ciclo while (mientras) ejecuta un conjunto de instrucciones mientras se cumpla una condición.

```
n = int(input())
```

```
while n != 42:
```

```
    print(n)
```

```
    n = int(input())
```

Ciclos - for

El ciclo for (para) ejecuta un conjunto de instrucciones un número determinado de veces.

```
print("Enteros del 1 al 5:")
```

```
for i in range(1, 6):
```

```
    print(i)
```

Ciclos

Plantear **solo el ciclo** para los siguientes problemas:

1. Leer enteros hasta que se digite un valor negativo.
2. Leer la edad de 50 personas.
3. Leer un dato cada 15 minutos durante 2 horas.
4. Leer valores de ph hasta que se digite uno incorrecto.
5. Leer una temperatura cada hora durante 3 días.

Ejercicio en Clase y por fuera de Clase

- Haga un programa que muestre los números del 1 al 1000
- Haga un programa que sume todos los números pares del 1 al 10000?
- Para resolver el problema impuesto a Gauss (sumar todos los números del 1 al 100) existe la solución dada por Gauss y otra realizada a partir de ciclos y/decisiones
 - Realice el programa a partir de la solución dada por Gauss
 - Realice el mismo programa pero esta vez sin la ecuación de Gauss sino con ciclos y/o decisiones
 - ¿Cuál de las posibles soluciones piensa usted que es más eficiente y **POR QUÉ?**

Taller en Clase (5%)

Actividad Individual, Duración 30 minutos al finalizar las 3 horas semanales.

Objetivos de Evaluación: identificar el avance personal según los objetivos propuestos para las dos primeras semanas

Contenido a Evaluar: Expresiones, Variables, Tipos, Entrada y Salida básica, Decisiones, Ciclos

Recursos Adicionales

<https://spoj.com/>

<https://ideone.com/>

<https://www.pythontutorial.net/>

<http://python.swaroopch.com/>

<https://pragprog.com/book/gwpy2/practical-programming>

<https://coderbyte.com/course/learn-python-in-one-week>

<https://developers.google.com/edu/python/>



Para la Próxima Clase/Semana

1. Leer Capítulos 4 y 5 del libro
2. Buscar un algoritmo para generar números primos
3. Consultar que es el triángulo de pascal
4. Exponer problemas que hayan tenido en el foro de la materia.
5. Realizar las tareas dejadas en clase



Referencias y Lecturas Adicionales

- History of Computing - <http://slideplayer.com/slide/5126850/>
- Python Google Class - <https://developers.google.com/edu/python/>
- Computer Programming Khan Academy -
<https://www.khanacademy.org/computing/computer-programming>
- <https://code.org>
- <http://www.skulpt.org/>
- <https://repl.it/languages/python3>
- <https://www.khanacademy.org>
- <https://www.codecademy.com/es/learn/python>
- <https://www.codeschool.com/courses/try-python/>
- <https://www.edx.org/course/introduction-computer-science-harvardx-cs50x>