PYTHON

UNI – FIIS
Glen Rodríguez R.

Python

- Bautizado por la comedia británica “Monty Python’s Flying Circus”
- Página oficial: http://www.python.org
- https://www.python.org/about/gettingstarted/
- Ventajas:
  - Gratuito
  - Potente
  - Bastante usado (Google, NASA, Yahoo, Electronic Arts, etc.)
  - Buen soporte (librerias)

Historia

- Desarrollado al inicio de los 90’s por Guido y Rossum.
- Python fue diseñado para hacerle la vida fácil al programador:
- Pro: escribir rápido los programa (y aprender rápido el lenguaje)
- Contra: los programas en Python no están optimizados (no corren tan rápido) como en otros lenguajes (ej.: C/C++).

Como practicar?

- Bajarse la versión 3 de preferencia
  - http://www.python.org/download/
- Puedes editar programas con cualquier editor (notepad) y correrlo de línea de comando
- O puedes bajarte un IDE e integralo con Pyhton
  - NetBeans
  - Eclipse: http://www.pydev.org/download.html

Como practicar?

- Configurar el path:
  - http://docs.python.org/using/windows.html
  - http://docs.python.org/using/unix.html
- Si no te gusta, debes grabar y correr los programas en el mismo directorio donde está el ejecutable de Python
- Pero no conviene! (mucho desorden)
- http://docs.python.org/py3k/tutorial/index.html

Fases de la programación

- Diseñar el programa
- Editarlo y grabarlo (el código fuente)
- “Traducirlo”
- Ejecutarlo

Tipos de traductores

- Interpretes (como Python)
  - “Traduce” en tiempo de ejecución
  - Si hay un error en el programa, la ejecución aborta cuando el interprete llega a la línea con el error
- Compiladores (como C++)
  - El programa se “traduce” antes de ejecutar, una sola vez, y se crea el ejecutable.
  - Si el programa tiene errores, no se crea ejecutable y por lo tanto no se puede ejecutar el programa.
  - Pero pueden haber errores en ejecución (ejemplo, división entre cero)

El primer programa

- Salvarlo como “primer.py”
- print(“hola”)
- Ejecutarlo como “python primer.py”
- Editarlo: print(“hola”,end=””)

Variables

- Los nombres de una ubicación en memoria donde se guarda información (temporalmente)
- Si se pone una nueva pieza de información en lavariable donde ya había una, la anterior se borra para poner la nueva en su lugar.
- Tipos de variables: integer, floating point, strings
- Cómo se crean?
- <nombre de variable> = <valor>
- Ejemplos:
- num1=10
- num2=10.0
- nombre="juan"

El operador de asignación =

- “=“ en programación no indica igualdad matemática, indica asiganción
- Ejemplo
  - y=3
  - x=y
  - x=6
  - y=13
- Cuál es el valor final de las variables?

Reglas para bautizar variables

- Python fuerza algunas reglas
- Otras son "reglas de estilo": no son obligatorias pero recomendables para tener un prograam fácil de leer

No pueden ser palabras reservadas de Python

and	as	assert	break
class	continue	def	del
elif	else	except	exec
finally	for	from	global
if	import	in	is
lambda	not	or	pass
print	raise	return	try

Reglas de estilo

- Nombre debe significar algo al ser leido. Ej.: edad, x, y
- Deben empezar con una letra (regla de Python) pero no con barra abajo (estilo). Ej: 2x, peso, _peso
- Python diferencia mayúsculas y minúsculas, pero el lector no lo hace a veces. no usar dos variables Peso y peso, o edadNac y edadnac.

Reglas de estilo

- Ser todo en minúsculas, excepto cuando más de una palabra se junta. Ej.: edad, peso, tallaRequerida, sueldo_base
- En inglés o castellano? Depende del público que puede usar el código fuente. Ej.: edad vs. age

Función print()

- Puede tener 0, 1 o más argumentos (inputs)
- Los argumentos se separan por comas. Los imprime en pantalla y salta el cursor a la sgte. linea
- El modificador "end" no es obligatorio pero se puede usar para cambiar el salto de cursor
- Sin argumetos: solo muestra una línea en blanco
- print("hola")
-
- print("UNI-",end="")
- print("FIIS")

Ejemplo

- Formato:
- print(arg1,arg2 …)1
- Ejemplo: output2.py
- num = 10.0
- name = "pepe"
- print("Listo?")
- print("Num=", end="")
- print(num)
- print()
- print("Me llamo: ", name)
-

Variable vs texto

- aString = "Saludos"
- print(aString)
- print("aString")

Formato de salida

- num= 2/3
- print(num)
-
- Sale 0.6666666666666
-
- print ("%4.1f" %num)
- print ("%5.1f" %num)
- print ("%3.1f" %num)

Otro ejemplo

- num = 123
- st = "Lima 231"
- print("num=%d" %num)
- print("lugar: %s" %st)
- num = 12.5
- print("%f %d" %(num,num))

Columnado

- print ("%3s%-3s" %("ab", "ab"))
- print ("%-3s%3s" %("ab", "ab"))
- Positivo: alineado a la derecha
- Negativo: alineado a la izquierda

Caracteres especiales

Secuencia	Descripción
\a	Alarma: “beep”.
\n	Newline: mueve cursor al inicio de la sgte linea
\t	Tab
\'	Comilla simple
\"	Comilla doble
\\	Backslash

Constantes

- En Python: variables que no pueden cambiar
- Costumbre: nombre en MAYÚSCULAS
- ej.: PI=3.1416
- Qué pasa si la quiero cambiar?
- NADA… Python no vigila esto, el prograamdor debe hacerlo

Por qué usarlas?

- Qué es más fácil de entender / leer / mantener?

```
populationChange = (0.1758 – 0.1257) * currentPopulation
```

Vs.

```
#YES
BIRTH_RATE = 17.58
MORTALITY_RATE = 0.1257
currentPopulation = 1000000
populationChange = (BIRTH_RATE - MORTALITY_RATE) * currentPopulation
```

Operadores aritméticos

Operador	Descripción	Ejemplo
=	Asignación	num = 7
+	Suma	num = 2 + 2
-	Resta	num = 6 - 4
*	Multiplicación	num = 5 * 4
/	División	num = 9 / 2
//	División entera	num = 9 // 2
%	Modulo	num = 8 % 3
**	Exponeciación	num = 9 ** 2

Precedencia

- De abajo hacia arriba en la tabla
- Por los paréntesis
- De izquierda a derecha
- 3 ** 2 ** 3
- Es buena costumbre usar paréntesis para aclarar el orden de las operaciones

Input

- Pidiendo información al usuario por teclado
- Lee texto (strings)
- Formato:
- <variable name> = input()
- <variable name> = input("<Prompting message>"
- Ejemplo: input1.py
- print("What is your name: ")
- name = input()
-
- name = input("What is your name: ")
-
- print("What is your name: ", end="")
- name = input()

Ojo con los tipos !

- Qué pasa ?

```
x = '100'

y = '-10.5'

print(x + y)

print(int(x) + float(y))
```

Ojo

- Y aquí?

```
print(12+33)
print('12'+'33')
x = 12
y = 21
print(x+y)
print(str(x)+str(y))
```

Ojo

- Cuál es el error? Corríjalo

```
num = input("Entre un número")
numHalved = num / 2
```

Ojo

- Y acá? Cuál sería la salida del programa?

```
HUMAN_CAT_AGE_RATIO = 7
age = input("Entre su edad: ")
catAge = age * HUMAN_CAT_AGE_RATIO
print ("Edad equivalente en gato: ", catAge)
```

Comentarios

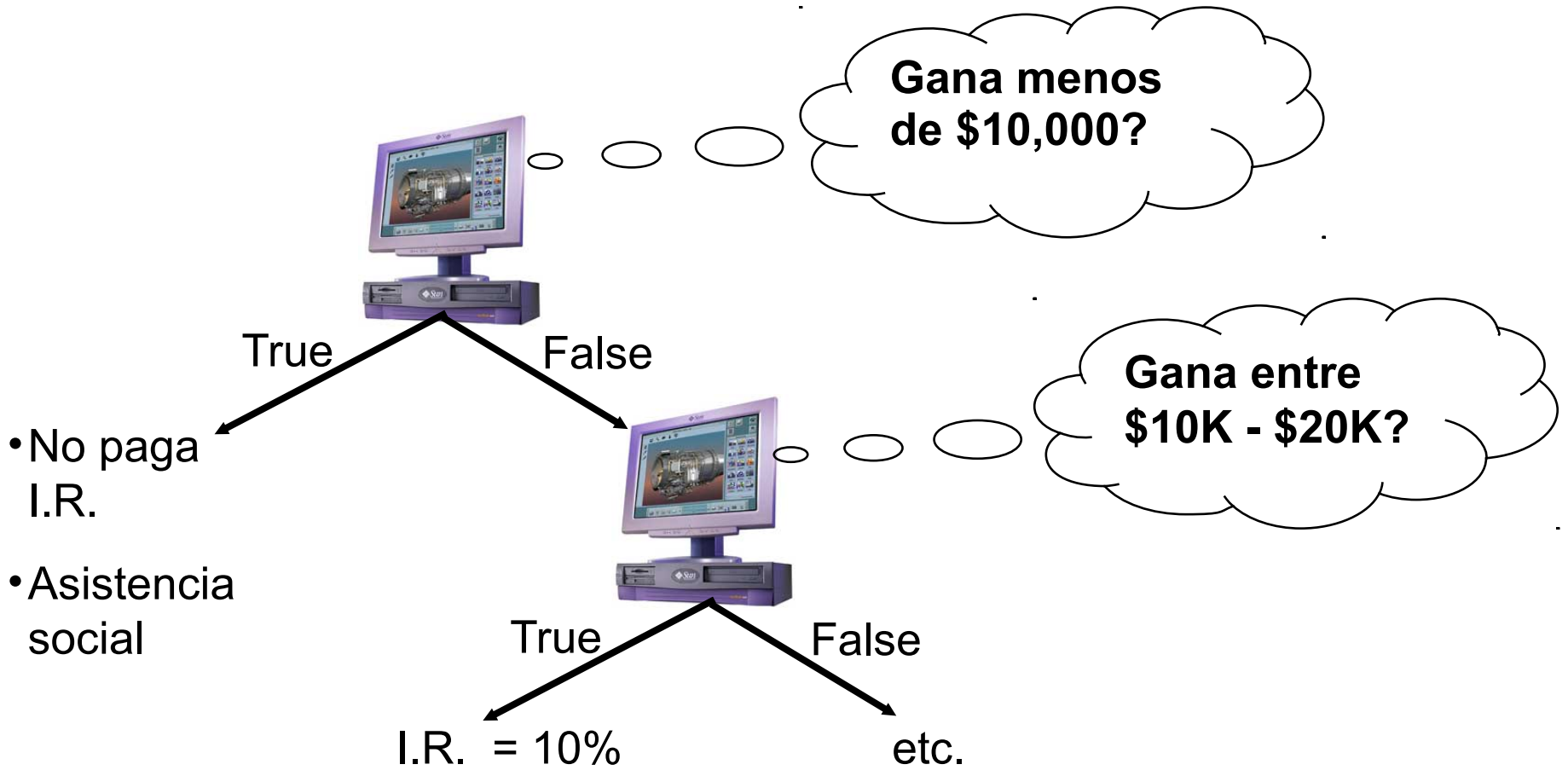
Formato:

```
# <Documentation>
```

Ejemploss:

```
# Tax-It v1.0: programa para calcular impuestos
# solo para Perú
# hecho por Pedro Navaja
```

Condiciones / bifurcaciones



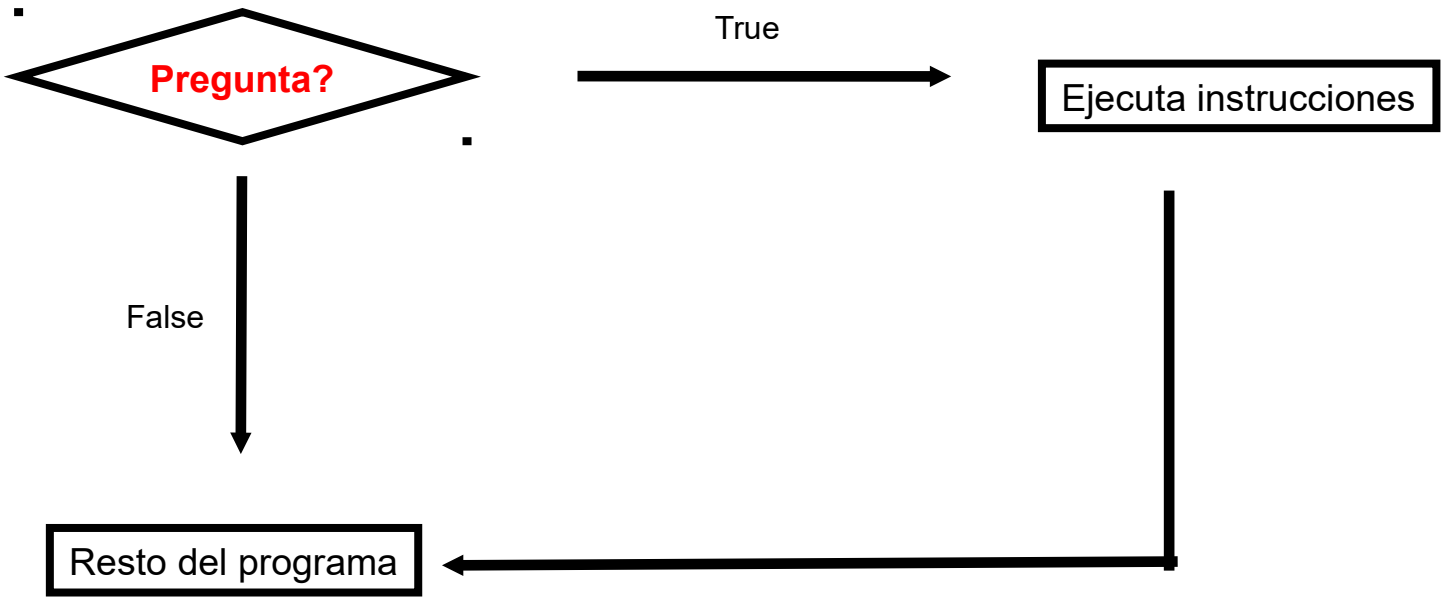
Otros ejemplos

- Usar cuando ciertos procesos sólo se ejecutan (dependen) en ciertas condiciones
- Si el usuario no acepta las condiciones, no dejarlo entrar al sitio web
- Si la rentabilidad de una línea de productos es menor de 1%, despedir al gerente de producto
- Si el usuario ha marcado la casilla de “más información”, mandarle emails con propaganda
- Si el usuario ha entrado una edad negativa o mayor de 120, mostrar un mensaje de error

Condicionales en Python

- Instrucciones para condicionales:
- If (ejecución de un caso especial para una condición)
- If-else (dos condiciones mutuamente excluyentes)
- If-elif-else (múltiples condiciones excluyentes)

If


Pregunta?
True
Ejecuta instrucciones
False
Resto del programa

If

- Toma de decisión: verificar si la condición es verdadera (en caso afirmativo, hacer algo).
- Format:

(Formato genérico)

```
if (Boolean expression):
    body
```

(Más detallado)

```
if (<operando> <operador relacional> <operando>):
    body
```

Expresión booleana

Nota: Indentación obligatoria

Ejemplos

- Programa (if1.py):

```
age = int(input("Edad: "))
if (age >= 18):
    print("Es un adulto")
```

```
if (operand     relational operator    operand):
if (age            >=                   18):
```

Operandos

- Pueden ser variables o constantes de tipo:
  - Integer
  - Floats (real)
  - Boolean (true, false)
  - String
- Ambos deben ser del mismo tipo (no se pueden comparar papás con pescados)

Operadores relacionales

Python (operador)	Equivalente matemático	Significado	Ejemplo
<	<	Menor que	5 **<** 3
>	>	Mayor que	5 **>** 3
==	=	Igual a	5 **==** 3
<=	≤	Menor o igual que	5 **<=** 5
>=	≥	Mayor o igual que	5 **>=** 4
!=	≠	Diferente que	x **!=** 5

Indentación

- En Python es obligatorio para distinguir los bloques.

```
# Single statement body
if (num == 1):
    print("Body of the if")
print("After body")

# Multi-statement body (program ‘if2.py’)
taxCredit = 0
taxRate = 0.2
income = float(input("What is your annual income: "))
if (income < 10000):
     print("Eligible for social assistance")
     taxCredit = 100
tax = (income * taxRate) – taxCredit
print("Tax owed $%.2f" %(tax))
```

Ojo

- Operador = vs Operador ==

```
if (num = 1):   # No es lo mismo que
if (num == 1):
```

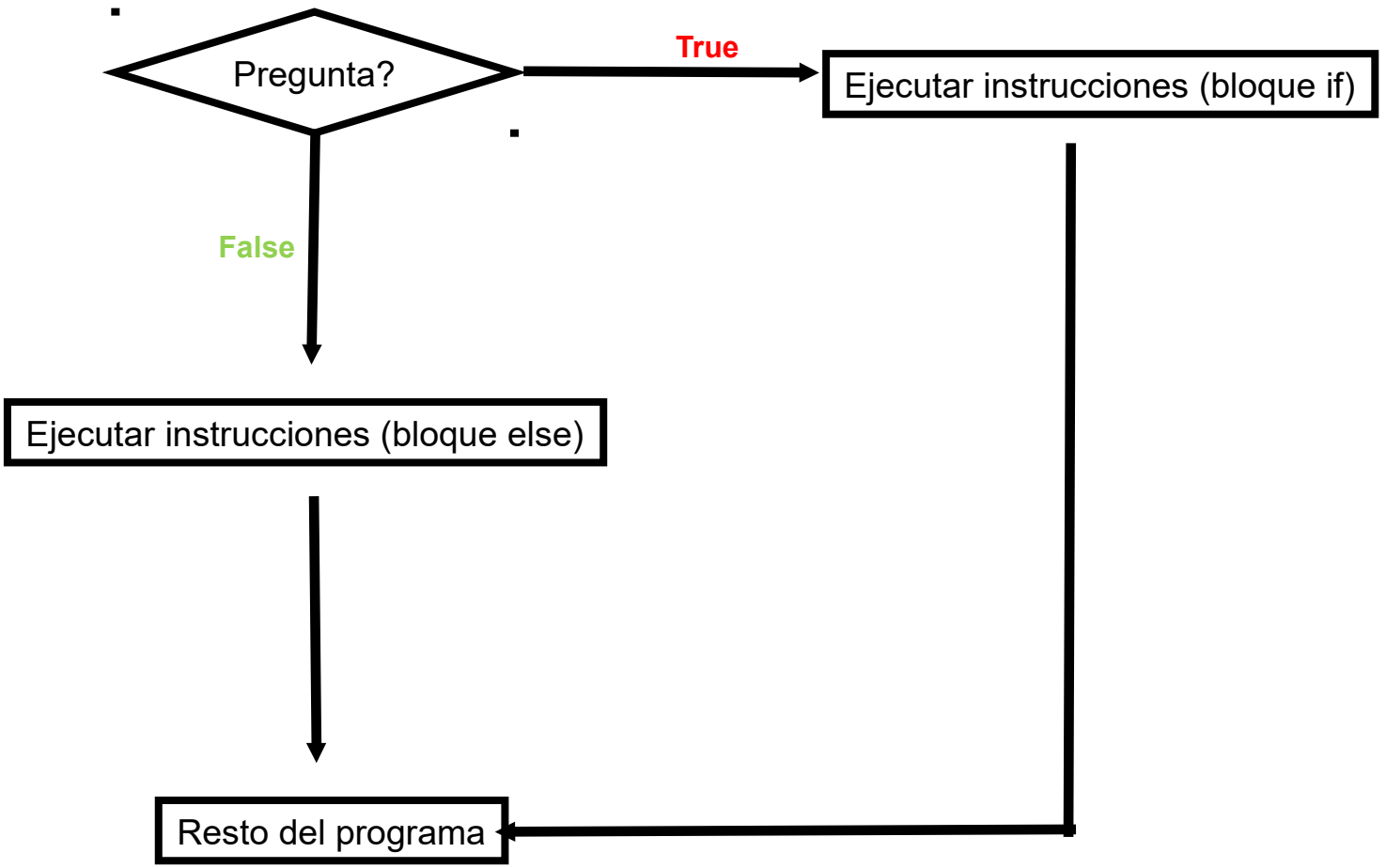
- Costumbre en otros lenguajes: primer operador que sea la variable. En Python puede ser útil invertir la costumbre

```
if (1 == num)
if (num == 1)
if (1 = num)
if (num = 1)
```

Ojo

- num = 1 asigna el valor 1 a la variable num
- num == 1 internamente ejecuta la comparación y obtiene true o false. El valor de num no cambia.

If-else


Pregunta?
True
Ejecutar instrucciones (bloque if)
False
Ejecutar instrucciones (bloque else)
Resto del programa

If-else

- Ejecutar un blqoue de intrucciones si la condición es verdadera, y otro bloque si es falsa
- Formato:

```
if (operando  operador relacional  operando):
    bloque del 'if'
else:
    bloque del 'else'
instrucciones del resto del programa
```

Ejemplo

- Programa if-else1.py

```
if (edad < 18):
    print("menor de edad")
else:
    print("Adulto")
print("Procedamos con la encuesta")
```

Ojo

- Hay más de una forma de hacer lo mismo

```
if (edad >= 18):
    print("Adulto")
else:
    print("menor de edad")
print("Procedamos con la encuesta")
```

Ojo: es lo mismo?

```
if (edad >= 18):
    print("Adulto")
else:
    print("menor de edad")
    print("Procedamos con la encuesta")
```

Ejemplo

```
if (ingresos < 800):
    print("Eligible para Beca 18")
    tasaImpuestos = 0.01
else:
    print("No elegible para Beca 18")
    tasaImpuestos = 0.12
impuestos = (ingresos * tasaImpuestos) - deducible
```

Operaciones lógicas

- Las 3 más comunes:
  - Logical AND
  - Logical OR
  - Logical NOT

Ejercicio

- Asumir que a=2, b=4, c=6
- Rellenar:

Expresión	Resultado
a == 4 or b > 2	
6 <= c and a > 3	
1 != b and c != 3	
a >-1 or a <= b	
not (a > 2)	

Ejemplos

- Formato genérico:

```
if (expr.booleana) operación logica (expr.booleana):
        body
```

- **EJEMPLO:**

```
if (x > 0) and (y > 0):
   print("Todos son positivos")
```

Ejemplo

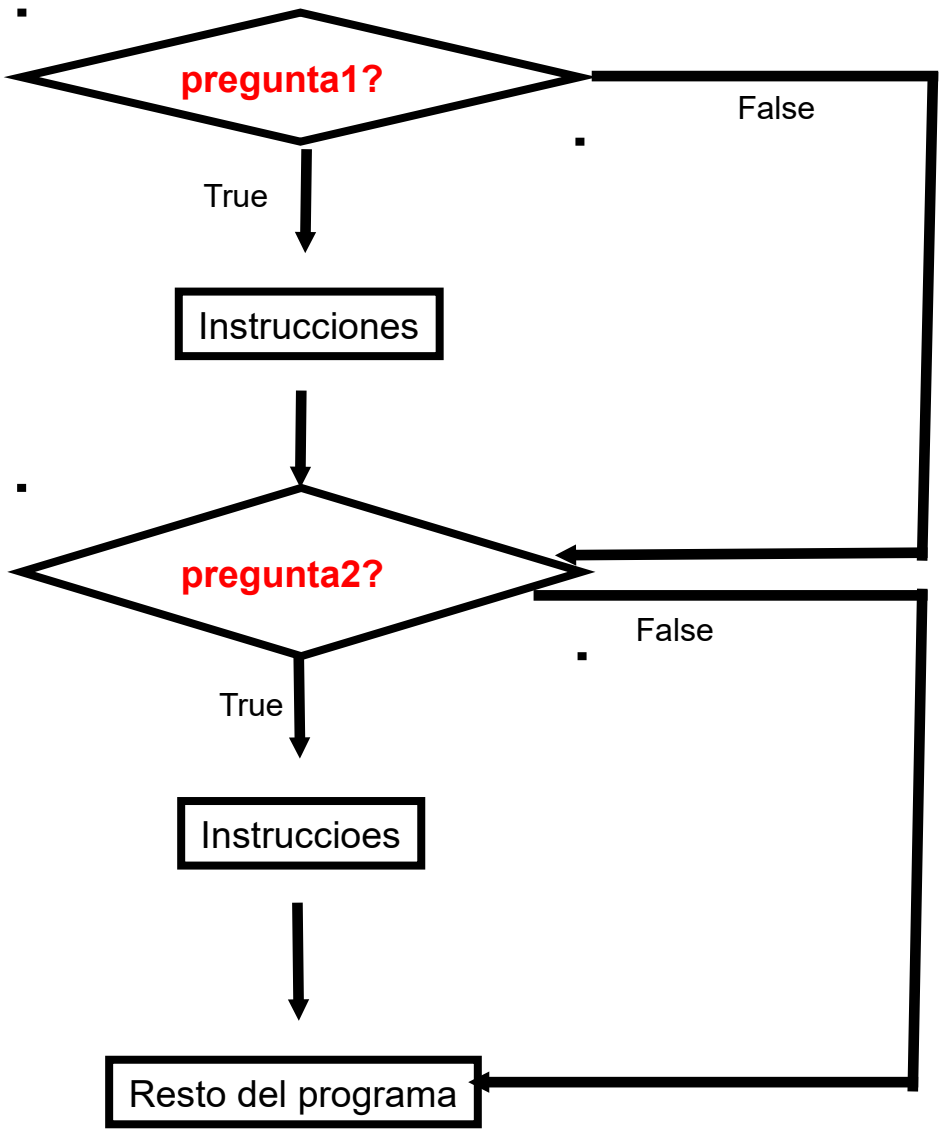
```
notas = float(input("Nota promedio (0-20.0): "))
tpoExperiencia = int(input("Número de años de experiencia: "))
if (notas > 14) or (tpoExperiencia > 5):
    print("Contratado")
else:
    print("Nivel insuficiente")
```

If anidados

```
if (income < 10000):
    if (citizen == 'y'):
        print("This person can receive social assistance")
        taxCredit = 100
tax = (income * TAX_RATE) - taxCredit
```

- Se pudo haber usado un AND y ahorrarse un if! Inténtelo

Múltiples if



Múltiples if

Formato:

```
if (expr. booleana 1):
     bloque 1
if (expr. booleana 2):
     bloque 2
          :
resto del programa
```

Multiples if

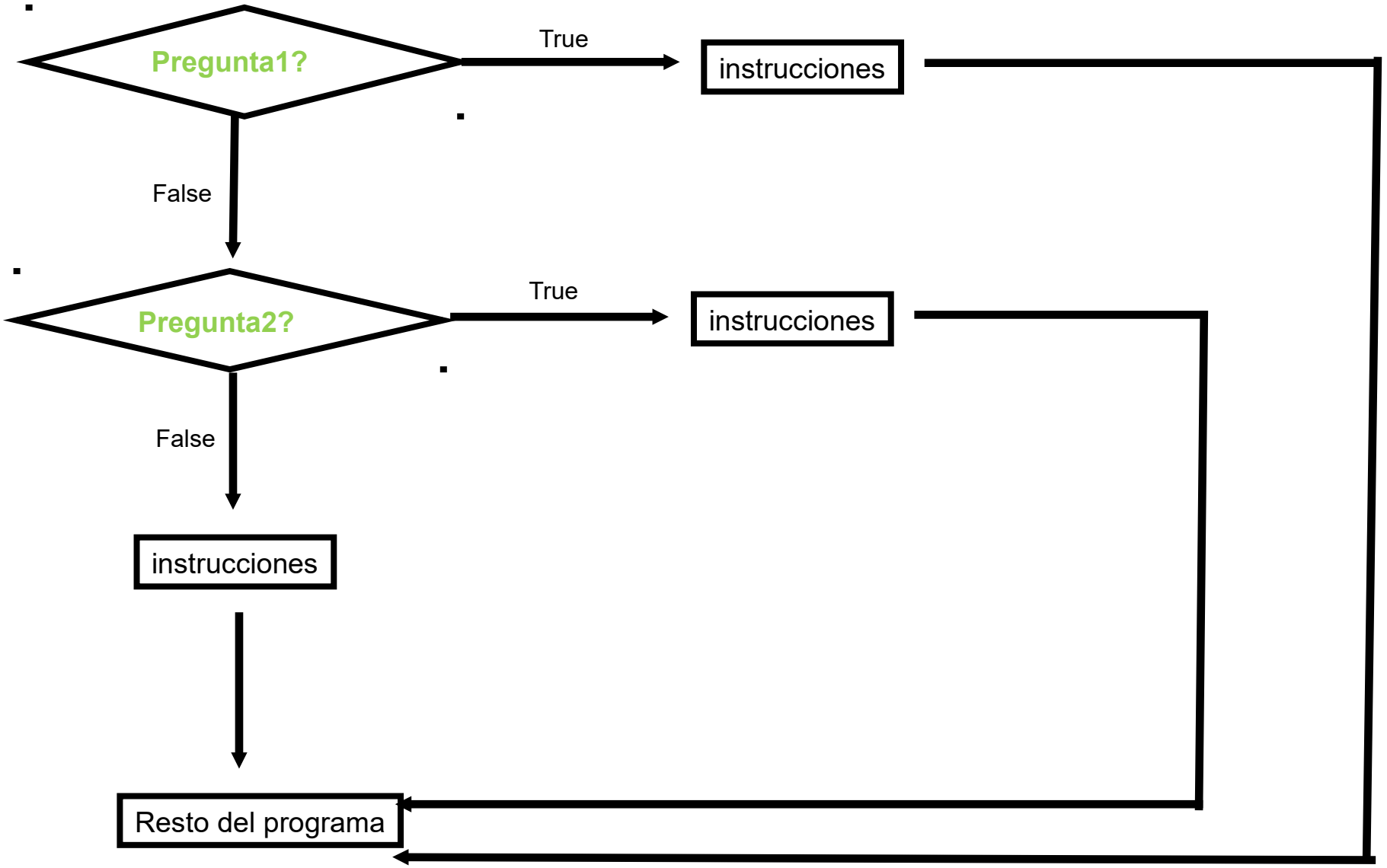
- Pueden ser exclusivos o no:

```
if (gpa == 4):
    letter = 'A'
if (gpa == 3):
    letter = 'B'
if (gpa == 2):
    letter = 'C'
if (gpa == 1):
    letter = 'D'
if (gpa == 0):
    letter = 'F'
```

Otro ejemplo

```
if (notaMatemáticas > 10):
     print("Aprobaste matemáticas!")
if (notaFísica > 10):
     print("Aprobaste Física!")
if (notaQuímica > 10):
     print("Aprobaste Química!")
```

If-elif-else


Pregunta1?
True
instrucciones
False
Pregunta2?
True
instrucciones
False
instrucciones
Resto del programa

Es exclusivo !

- Formato:

```
if (Boolean expression 1):
     body 1
elif (Boolean expression 2):
     body 2
          :
else
     body n
statements after the conditions
```

Es exclusivo !

Ejemplo

```
if (gpa == 4):
    letter = 'A'
elif (gpa == 3):
    letter = 'B'
elif (gpa == 2):
    letter = 'C'
elif (gpa == 1):
    letter = 'D'
elif (gpa == 0):
    letter = 'F'
else:
    print("GPA must be one of '4', '3', '2', '1' or '1'")
```

Enfoque más eficiente. Si se cumple la 1ra condicion, no tiene que comprara el resto

Beneficio extra:

El bloque del else se jecuta si y solo sí las condiciones todas fueron falsaa (útil para manejar errores, para testear).

Ejercicio

- Hacer un programa que pida un número del 1 al 10 y de su equivalente en números romanos. Si el número ingresado no está en ese rango, emitir un mensaje de error

Ojo con las comparaciones con floats

- Las operaciones con floats usan “matemática del CPU”, no matemática del cerebro humano.
- No hay garantía que 1-0.45 = 0.55 exactamente (puede salir 0.5500000001)
- Use enteros, o use comparaciones con margen de error

Ejercicio

- Escriba un programa que convierta notas "peruanas" a notas de letras: A(20 a 18), B(hasta 16), C (hasta 14), D (hasta 10), F (menor que 10)