Variables, Expresiones e I/O

Clase #03

IIC1103 – Introducción a la Programación

Marcos Sepúlveda (marcos@ing.puc.cl)

Hay varias versiones de Python

- En este curso, aprenderemos:
  - Python 3.*

Veremos hoy …

- Tipos de datos básicos
- Variables
- Funciones de Input/Output

TIPOS DE DATOS BÁSICOS

Tipos de datos básicos

En Python, los tipos básicos se dividen en:

- **Números – *int*, *float* , *complex***

  Ejemplos:

  - **int** (entero): **3**
  - **float** (reales; de punto flotante): **15.57**
  - **complex** (complejo): **7+5j**

- **Cadenas de texto (strings) – *str***

  Ejemplo:

  - **str**: **"Hola mundo!"**

- **Valores booleanos – *bool***
  - **bool**: **True** (verdadero)
  - **bool**: **False** (falso)

Números

- **Enteros**
  - Números positivos o negativos (además del cero) que no tienen decimales
- **Reales**
  - Son aquellos números que tienen decimales
- **Complejos**
  - Son aquellos que tienen parte imaginaria
  - La mayor parte de los lenguajes de programación **NO** tienen implementado este tipo de dato
  - Ejemplo:

```
1  complejo = 3.0 + 4.5j
2  print(complejo)
```

Nota: la palabra *complejo* en este ejemplo es el nombre de una variable, y no una palabra reservada

Ejemplos

```
5                   # int (número entero)
6.8                 # float (número real)
"Hola Mundo"        # str (texto)
True                # bool (verdadero)
False               # bool (falso)
```

```
>>> type(5)
<class 'int'>
>>> type(6.8)
<class 'float'>
>>> type("Hola Mundo")
<class 'str'>
>>> type(True)
<class 'bool'>
>>> type(False)
<class 'bool'>
```

Operadores

- Los operadores permiten realizar operaciones aritméticas con los números

Operador	Descripción	Ejemplo
`+`	Suma	`suma = 3 + 2  # es 5`
`-`	Resta	`resta = 4 - 7  # es -3`
`-`	Inverso aditivo	`inverso = -7  # es -7`
`*`	Multiplicación	`multiplicacion = 2 * 6  # es 12`
`**`	Potencia	`potencia = 2 ** 6  # es 64`
`/`	División	`division = 3.5 / 2  # es 1.75`
`//`	División entera	`division_entera = 3.5 // 2  # es 1.0`
`%`	Módulo	`modulo = 7 % 2  # es 1`

Nota: El símbolo # sirve para escribir un comentario

Operadores

- En la ***división*** el resultado que se devuelve es un número real, mientras que en la ***división entera*** el resultado que se devuelve es solo la parte entera

`/`	División	`division = 3.5 / 2  # es 1.75`
`//`	División entera	`division_entera = 3.5 // 2  # es 1.0`

- El operador ***módulo*** no hace otra cosa que devolvernos el resto de la división entre los dos operandos. En el ejemplo, **7*/2*** sería **3**, con **1** de resto, luego el módulo es **1**.

`%`	Módulo	`modulo = 7 % 2  # es 1`

Ejemplos de operaciones aritméticas

- Es posible utilizar Python como una calculadora aplicando operaciones aritméticas a los números.

- Veamos algunos ejemplos:

```
Python 3.2.3 (v3.2.3:3d0686d90f55, Apr 10 2012, 11:25:50)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> 3+5
8
>>>
```

Ejemplos de operaciones aritméticas

- ¿Más de un operador aritmético?
  - *Sí, es posible*

```
>>> 3+5+6
14
>>>
```

```
>>> 2-5+6
3
>>>
```

Ejemplos de operaciones aritméticas

- **Asociatividad**
  - El orden en que se efectúan las operaciones es de izquierda a derecha.

```
>>> 3+5+6
14
>>>
```

((3+5)+6)

```
>>> 2-5+6
3
>>>
```

((2-5)+6)

Ejemplos de operaciones aritméticas

- **Asociatividad**
  - El orden en que se efectúan las operaciones es de izquierda a derecha.

- Aplicando paréntesis…

```
>>> 2-(5+6)
-9
>>>
```

…que es diferente a…

```
>>> 2-5+6
3
>>>
```

Ejemplos de operaciones aritméticas

- ¿Se puede poner espacios en blanco?
  R: Sí, es posible

```
>>> 4  +  6  -  (2  -  1)
9
>>>
```

- …¿Y se puede poner espacios en medio de los números?
  R: No, no es posible

```
>>> 10 + 20 + 30  #correcto
60
>>> 10 + 2 0 + 30 #incorrecto
SyntaxError: invalid syntax
>>>
```

- …¿Y se puede poner espacios al comienzo de los números?
  R: No, no es posible

```
>>>     10 + 20+30  #incorrecto por los espacios al comienzo

SyntaxError: unexpected indent
>>>
```

…más operadores aritméticos…

- **Multiplicación y división**

```
>>> 2*3
6
>>> 4/2
2.0
>>> 3*4/2
6.0
>>> 12/3*2
8.0
>>>
```

La expresión 3*4/2, equivale a ((3*4)/2) $= \frac{3 \cdot 4}{2}$

La expresión 12/3*2, equivale a ((12/3)*2) $= \frac{12}{3} \cdot 2$,

…más operadores aritméticos…

- Recordar que la asociatividad es de izquierda a derecha.
- Además, hay que aplicar ***precedencia***

```
>>> 2*4+5
13
>>> 2+4*5
22
>>>
```

La expresión 2*4+5, equivale a ((2*4)+5)

La expresión 2+4*5, equivale a (2+(4*5))

…más operadores aritméticos…

- Recordar que la asociatividad es de izquierda a derecha.
- Además, hay que aplicar ***precedencia***

```
>>> 2**3
8
>>>
```

La expresión 2**3, equivale a $2^3=8$

```
>>> 2**3**2
512
>>>
```

La expresión 2**3**2, equivale a:

$$2^{(3^2)} = 2^9 = 512, \text{ y no a } (2^3)^2 = 8^2 = 64$$

Nota: La exponenciación es asociativa por la derecha

Precedencia de operadores en expresiones aritméticas

Operación	Operador	Aridad	Asociatividad	Precedencia
Exponenciación	**	Binario	Por la derecha	1
Identidad	+	Unario	—	2
Cambio de signo	-	Unario	—	2
Multiplicación	*	Binario	Por la izquierda	3
División	/	Binario	Por la izquierda	3
Módulo (o resto)	%	Binario	Por la izquierda	3
Suma	+	Binario	Por la izquierda	4
Resta	-	Binario	Por la izquierda	4

Tabla 2.1: Operadores para expresiones aritméticas. El nivel de precedencia 1 es el de mayor prioridad y el 4 el de menor.

Bool

- Los objetos de tipo ***bool*** son variables que pueden tomar sólo dos valores: ***True*** o ***False***.

Operadores booleanos:

- **==**, **>=**, **<=**, **>**, **<**, ***!=***
  - Válido tanto para tipos ***int***, ***float***, ***complex*** y ***str***, entre otros.

- ***and***, ***or***, ***not***
  - Válido para tipos ***bool***.
  - Permiten operadores lógicos sobre variables o expresiones booleanas.

Ejemplos

```
variable1 = 2>5  # Es False
variable2 = 3<5  # Es True
variable3 = True # Es True
variable4 = 5==5 # Es True ya que 5 es igual a 5
text = "hola"
var5 = text == "hola" # Es True
var6 = 5=="5" # Es False pues el entero 5 es distinto al string "5"
var7 = 5>=4   # Es True pues 5 es mayor o igual a 4
dist = 5!=4   # Es True pues 5 es distinto de 4
nodist = not (5!=4) # Es False pues negamos el valor de 5!=4
hola = 5>4 and 5<4  # Es False pues and exige que todas sean True
chao = 5>4 or 5<4   # Es True pues or exige al menos una sea True
```

Precedencia de operadores en expresiones booleanas

- Entre los operadores del tipo **bool**, las precedencias son las siguientes:

  Mayor precedencia
  - `<, <=, >, >=, !=, ==`
  - `not`
  - `and`
  - `or`

  Menor precedencia

- Ejemplo:
  - `expr1 and not expr2 or expr3` es equivalente a
  - `(expr1 and (not expr2)) or expr3`

- Más sobre precedencia de operadores:
  - https://docs.python.org/3/reference/expressions.html#operator-precedence

Precedencia de operadores

- Mi recomendación: ***"Usa siempre paréntesis para dejar claro lo que quieres"***
- En vez de `2*4+5`, escribe `(2*4)+5`
- En vez de `2**3**2`, escribe `2**(3**2)`
- En vez de `año%4 == 0 and año%100 != 0 or año%400 == 0`, escribe `((año%4 == 0) and (año%100 != 0)) or (año%400 == 0)`

VARIABLES Y EXPRESIONES…

Contexto

- En ocasiones se necesita que el computador “recuerde” ciertos valores para ser utilizados más adelante.

- Ejemplo: calcular el perímetro y área de un círculo de radio **1.298373**.

- Solución: aplicando las formulas **2*π*r** y **π*r**2**, obtenemos el perímetro y área, respectivamente.

```
>>> 2 * 3.14159265359 * 1.298373
8.157918156839218
>>> 3.14159265359 * 1.298373 ** 2
5.296010335524904
>>>
```

Variables

- Se introdujeron dos veces los valores de **π** y **r**, con muchos decimales. Es fácil cometer errores al manipular tantos números…

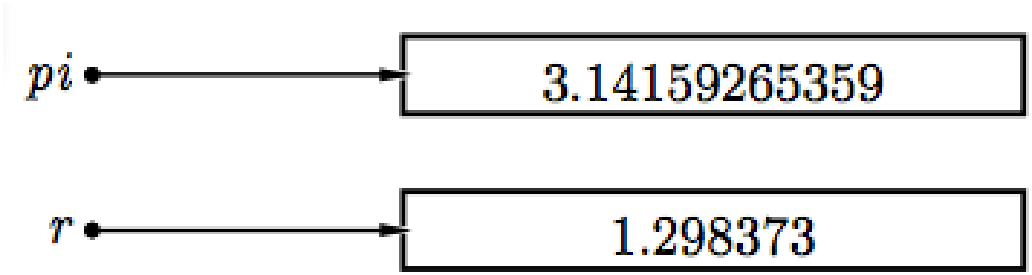
Para prevenirlos, se usarán ***Variables***

```
>>> pi = 3.14159265359
>>> r = 1.298373
```

```
>>> 2 * pi * r
8.157918156839218
>>> pi * r ** 2
5.296010335524904
>>>
```

Variables

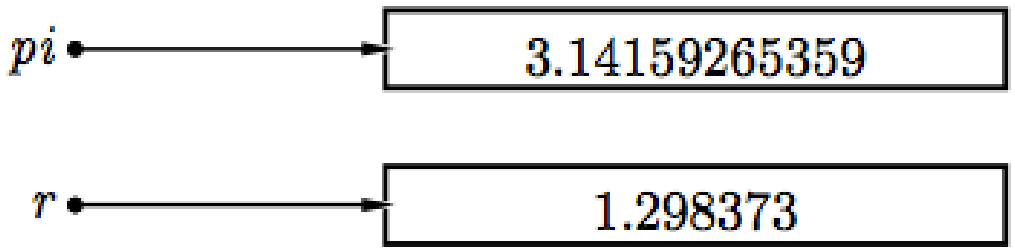
- ¿Qué es lo que sucede internamente en el computador?
  - Se crea la variable **pi** con el valor **3.14159265359**
  - Se crea la variable **r** con el valor **1.298373**
  - Al asignar el valor a una variable que no existía, Python reserva un espacio en la memoria, almacena el valor en él, y crea una asociación entre el nombre de la variable y la dirección de memoria de dicho espacio.



Asignación

- Acto de dar valor a una ***Variable***
  - El valor de la expresión que está a la derecha del signo **=**, se asigna a la variable de la izquierda

```
>>> pi = 3.14159265359
>>> r = 1.298373
```



- Otros ejemplos:
  - `salario = 23000`
  - `impuesto = 27.4`
  - `medallas_de_oro = 31`

Asignación

- Resultados de operaciones aritméticas también pueden ser asignados a variables

```
>>> pi = 3.14159265359
>>> r = 1.298373
>>> perimetro = 2 * pi * r
>>> area = pi * r ** 2
>>>
```

Desplegando valor de una variable

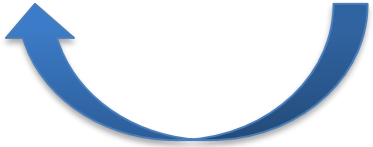
- Se puede desplegar el valor de las variables con tan sólo escribirlas en el área de trabajo o utilizando el comando ***print***

```
>>> pi = 3.14159265359
>>> r = 1.298373
>>> perimetro = 2 * pi * r
>>> area = pi * r ** 2
>>> area
5.296010335524904
>>> print(area)
5.296010335524904
>>> print(perimetro)
8.157918156839218
>>> perimetro
8.157918156839218
```

Resumiendo

- Para asignar valor a una variable, la sentencia es la siguiente:

```
variable = expresion
```



Asignación de derecha a izquierda

Asignación versus Comparación

== no es = (comparar no es asignar)

Al aprender a programar, muchas personas confunden el operador de asignación, =, con el operador de comparación, ==. El primero se usa exclusivamente para asignar un valor a una variable. El segundo, para comparar valores.

Observa la diferente respuesta que obtienes al usar = y == en el entorno interactivo:

```
>>> a = 10 ↲
>>> a ↲
10
>>> a == 1 ↲
False
>>> a ↲
10
```

Asignación versus Ecuación

- Una asignación no es una ecuación

```
>>> x = 3
>>> x = 3 + 1
>>> x
4
>>>
```

Nombres de variables

- Palabras que no se pueden ocupar como nombre de variables

```
and, asset, 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,
while, y field
```

  - Estas palabras pertenecen al lenguaje Python, por lo tanto son reservadas

- Nombres de variables válidos:
  - `Aceleracion_1`
  - `alumno_23`
  - `prueba_123`
  - `Integral15`

Asignaciones con operador

- Fijarse que la sentencia ***i=i+1*** aplica un incremento unitario al contenido de la variable ***i***. Esta forma de incremento es frecuente, por lo que se puede resumir de la siguiente manera:

```
>>> i += 1
Traceback (most recent call last):
  File "<pyshell#11>", line 1, in <module>
    i += 1
NameError: name 'i' is not defined
```

Error, ya que no estaba declarada la variable ***i***.

Primero debemos declarar la variable y asignarle un valor

```
>>> i=0
>>> i += 1
>>> i
1
>>>
```

Notar que entre el signo **+** y el signo **=** no debe haber espacios

Asignaciones con operador

- Se puede incrementar una variable con cualquier cantidad, incluso con una que resulte de evaluar una expresión:

```
>>> a = 3
>>> b = 2
>>> a += 4 * b
>>> a
11
>>>
```

a* += *4*b es equivalente a la asignación
a* = *a* + *4* * *b

Asignaciones con operador

- Todos los operadores aritméticos tienen su asignación con operador asociado.

- Supongamos que tenemos la variable ***Z***

```
z = 5
z += 2
z *= 2
z /= 2
z -= 2
z %= 2
z **= 2
```

Solución:
1.0

Ejercicios

- ¿Cuál es el resultado final de las siguientes sentencias?

```
incognita = 5
incognita += 4
incognita *= 2
incognita /= 6
incognita **= 3
incognita %= 9
```

Solución:
0.0

Ejercicios

Evalúa el polinomio $x^4 + x^3 + 2x^2 - x$ en $x = 1.1$.

Evalúa el polinomio $x^4 + x^3 + \frac{1}{2}x^2 - x$ en $x = 10$.

I/O – INPUT/OUTPUT

I/O – Input/Output

- Python incluye algunos comandos básicos para desplegar información al usuario o solicitarle datos.
  - Son **funciones** que vienen implementadas en Python.
  - Una **función** es una entidad que recibe ciertos parámetros, y en base a eso, ejecuta ciertas acciones que pueden terminar con el retorno de un valor.

- ***print()***: función que recibe como parámetro un valor y lo imprime en la consola.
  - En general, usaremos strings
  - Para combinar valores de variables y textos, usaremos otra función llamada ***str()***.

- ***str()***: función que transforma una expresión a un string.

- ***input()***: función que recibe como parámetro un string y pide otro string en consola.
  - Este string puede ser asignado a una variable.
  - Si queremos pedir un número, debemos transformar el string ingresado mediante ***int()*** o ***float()***. Se debe realizar lo mismo al recibir un ***bool()***.

Ejemplos

```
# ejemplos de print
print("hola")                    # imprime en consola hola
print("Mi nombre es: "+"Juan") # imprime en consola Mi nombre es: Juan
a = 3
print(a)                         # imprime en consola 3
print("El numero es: "+a)        # genera error de tipos
print("El numero es: "+str(a)) # imprime en consola El numero es: 3

# ejemplos de input
a = input("Diga un nombre : ")
b = int(input("Diga un número : "))
c = int(input("Diga otro número : "))
print("La suma de los números de "+a+" es: "+str(b+c))
```

I/O – Input/Output

- El ingreso de texto a través del teclado es capturado por Python a través de ***sys.stdin***. Para utilizarlo hay que importar ***sys***.

- ***readline()***: función de ***sys.stdin*** que lee una línea del input.

- ***strip()***: función que permite eliminar espacios y saltos de línea de un string.

```
# ejemplos de readline
import sys                    # solo se requiere la primera vez
print("Ingrese un texto y luego presione 'Enter'.")
s = sys.stdin.readline()
Está es una línea de texto. Termina cuando presiono 'Enter'.
s
"Está es una línea de texto. Termina cuando presiono 'Enter'.\n"
s = s.strip()
s
"Está es una línea de texto. Termina cuando presiono 'Enter'"
```

Ejemplos

```
# ejemplos de readline
import sys                    # solo se requiere la primera vez
print("Diga un nombre : ")
a = sys.stdin.readline()
nombre = a.strip()
print("Diga un número : ")
b = int(sys.stdin.readline())
print("Diga otro número : ")
c = int(sys.stdin.readline())
print("La suma de los números de "+nombre+" es: "+str(b+c))
```

EJERCICIOS

Ejercicio 1 – describiendo un cuadrado

- Escriba un programa que pida al usuario el lado de un cuadrado, y luego despliegue el área, el perímetro y la diagonal de dicho cuadrado.

```
lado = float (input ("Ingrese el lado del cuadrado: "))
perimetro = lado*4
area = lado**2
diagonal = lado * (2**(1/2))
print("El perímetro es: "+str(perimetro))
print("El área es: "+str(area))
print("La diagonal es: "+str(diagonal))
```

```
import sys
print("Ingrese el lado del cuadrado: ")
lado = float(sys.stdin.readline())
perimetro = lado*4
area = lado**2
diagonal = lado * (2**(1/2))
print("El perímetro es: "+str(perimetro))
print("El área es: "+str(area))
print("La diagonal es: "+str(diagonal))
```

Ejercicio 2 – ¿pasó el curso?

- Un curso tiene 3 pruebas que ponderan un 20% cada una, y un examen que pondera el 40% restante. El curso se aprueba si el promedio no aproximado del curso es mayor o igual que 4.0.
- Escriba un programa que reciba las 4 notas e imprima ***True*** si usted aprueba el curso y ***False*** en caso contrario.

```
import sys
print("Ingrese las notas de las tres pruebas: ")
n1 = float(sys.stdin.readline())
n2 = float(sys.stdin.readline())
n3 = float(sys.stdin.readline())
print("Ingrese la nota del examen: ")
ex = float(sys.stdin.readline())
pasaste = (0.2* n1 +0.2* n2 +0.2* n3 +0.4* ex ) >= 4.0
print(pasaste)
```

Ejercicio 3 – “Adivino lo que piensas”

- Piensa un número del 2 al 10 ...
- Multiplica el número por 9
- Suma las dos cifras
- Réstale 5
- Transforma número en letra:
  - 1->A, 2->B, 3->C, ... etc.
- Piensa un país que empiece con esa letra ...
- Piensa un animal que empiece con la 2a letra del país ...

¡¡ Pero ... si en Dinamarca no hay Iguanas !!

Fuente: https://elabacodemadera.files.wordpress.com/2012/08/iguana.jpg?w=682

Ejercicio 3 – “Adivino lo que piensas”

```
print("")
print("TE ADIVINO LO QUE PIENSAS")
print("")

n1 = int(input("Piensa un número del 2 al 10; y escríbelo: "))
print(">> ", n1)

input("Multiplica el número por 9 y aprieta <enter>")
n2 = n1 * 9
print(">> ", n2)

input("Suma las dos cifras y aprieta <enter>")
cifra1 = n2 // 10
cifra2 = n2 % 10
n3 = cifra1 + cifra2
print(cifra1, "+", cifra2, "=", n3)

input("Réstale 5 y aprieta <enter>")
n4 = n3 - 5
print(">> ", n4)

input("Transforma número en letra: 1->A, 2->B, 3->C, ... <enter>")
print(">> ", n4, "-> D")

input("Piensa un país que empiece con esa letra ... <enter>")

input("Piensa un animal que empiece con la 2a letra del país ... <enter>")

# Leer mente
print("")
input("Aprieta <enter> cuando estés listo para que te lea la mente")
print("." * 10)
print("." * 10)
print("Pero si en DINAMARCA no hay IGUANAS !!!!")
```

Ejercicio 4 – “Te adivino la edad ... con Chocolates”

- ¿Cuántos chocolate comes al día? (elige un número entre 1 y 10)
- Multiplica ese número por 2
- Suma 5 al resultado anterior
- Multiplica eso por 50
- Súmale el año actual
- Si ya pasó tu cumpleaños de este año, entonces réstale 250
- Si todavía no pasa, réstale 251
- Réstale tu año de nacimiento

- Terminarás con un número de 3 o 4 dígitos.
- Los dos últimos dígitos son tu edad.
- La otra parte del número es las veces que comes chocolate al día.

Fuente: https://lamagiadelamagia.wikispaces.com/Trucos+matemágicos

Ejercicio 4 – “Te adivino la edad ... con Chocolates”

```
print("")
print("TE ADIVINO LA EDAD ... CON CHOCOLATES")
print("")

print("¿Cuántos chocolate comes al día?")
n = int(input("Elige un número entre 1 y 10: "))
print(">> ", n)
print("Multiplica ese número por 2")
n *= 2
print(">> ", n) # 2 * n
print("Suma 5 al resultado anterior")
n += 5
print(">> ", n) # 2* n + 5
print("Multiplica eso por 50")
n *= 50
print(">> ", n) # 100 * n + 250

añoActual = int(input("¿En qué año estamos?: "))
print("Súmale el año actual")
n += añoActual
print(">> ", n) # 100 * n + añoActual + 250

yaFue = input("¿Ya fue tu cumpleaños este año? (si - no): ")
if yaFue == "si":
    print("Dado que ya pasó tu cumpleaños de este año, ")
    print("entonces réstale 250")
    n -= 250
    print(">> ", n) # 100 * n + añoActual
else:
    print("Dado que todavía no pasa, réstale 251")
    n -= 251
    print(">> ", n) # 100 * n + añoActual - 1

añoNacimiento = int(input("¿En qué año naciste?: "))
print("Réstale tu año de nacimiento")
n -= añoNacimiento
print(">> ", n) # 100 * n + edad

print("")
print("Terminarás con un número de 3 o 4 dígitos.")
print("Los dos últimos dígitos son tu edad.")
print("La otra parte del número es las veces que comes chocolate al día.")
print("")
print(">> ", n)
print("")

edad = n % 100
cantidad = n // 100
print("Tu edad es ", edad,
      "años ... y te gusta comer", cantidad,
      "chocolates al día")
```

Ejercicios propuestos

- Calcule la hipotenusa de un triángulo rectángulo:
  1) Los lados **a** y **b** deben ser seleccionados al azar
  2) Usted solicita los lados al usuario

- Determinar si un número es par o impar

- Determinar si año es bisiesto:
  - Un año es bisiesto si es divisible entre 4, a menos que sea divisible entre 100. Sin embargo, si un año es divisible entre 100 y además es divisible entre 400, también resulta bisiesto

- Calcular el mínimo entre 2 números:
  1) Los números deben ser seleccionados al azar
  2) Usted solicita los números al usuario

- Calcule la sumatoria de **1** hasta **N**
  1) Usted solicita el valor de **N** al usuario