



Basics of Programming through Python Variables, Expressions and Statements

Introduction to Programming

COMP102

Term 3 2022-2023



Learning outcomes

- Introduction to variables
- Defining Variables and Constants
- Write expressions and statements
- Distinguish between the different data types in Python
- Reading Input from the Keyboard





Test your Knowlwadge

- What is the difference between variables and constants?
- What do you know about data type in PL?
- What do you know about the input of any program?



Variables





Defining variables

- A placeholder for a piece of information that **can change** in a later statement.
- Variables in PL are just named pieces of memory.
- A Python variable is a reserved memory location to store values. This means that when you create a variable you reserve some space in memory.
- General format is: `name_variable = expression`

Example: `age=29`



Variables

- When a variable is declared and assigned to a value, four fundamental attributes associated with it:
 - Its name
 - Its type
 - Its value(content)
 - Its address



Naming rules

- Variable names can contain these characters:
 - a through z
 - A through Z
 - The underscore character _
 - Digits 0 through 9
 - Can not contain **keywords or space**
- A variable name must start with letter or underscore character and cannot start with number.
- Variable names are case-sensitive(age, AGE, Age are 3 different variables).

Try and check:

Name	Correct/incorrect
M-1	
M_2	
2X	
x1	
a@c	
Total	
_carname	
Car name	
22=x	
else	

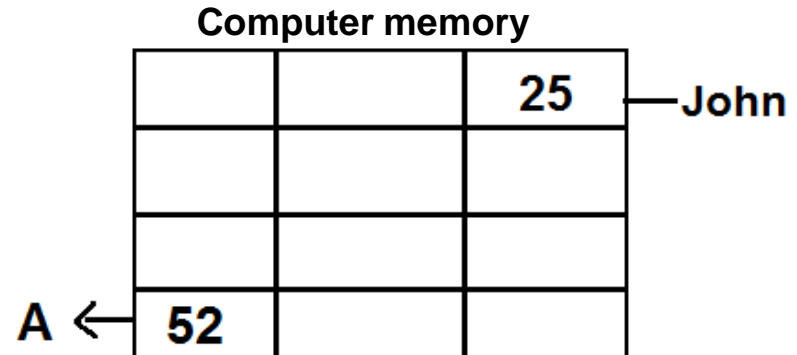
Assigning values to variables

- The equal sign(=) is used to assign values to variables.
- The operand to the left of the =operator is the name of the variable and the operand to the right of the =operator is the value stored in the variable.

John=25

A=52

So, “A ”is the name of the variable where the data 52 has been stored .
and “John” is the name of the variable where the data 25 has been stored .



Assigning values to variables

X=12.2

Y=14

X=100

		14	← Y
X →	12 2	100	

What are the values of X, Z and Y?

X=2

Z=X

X=5

Y=Z*X+1

print (X)

print (Z)

print(Y)

or print(X,Z, Y)

Try and check

Try 1

Write a Python program to calculate the salary of an employee knowing that **salary=nb of hours*rate.**

Sample input: hours=35

Rate=12.5

Solution:

```
hours=35.0
```

```
Rate=12.5
```

```
salary=hours*Rate
```

```
print(salary)
```



Constants

Constants

- Fixed values such as numbers, letters, and strings, are called “constants” because their value does not change
- Numeric constants are as you expect
- String constants use single quotes (') or double quotes (")

```
>>> print(125)
125
>>> print(77.6)
77.6
>>> print('Hello world')
Hello world
```




Numeric Expressions

- Because of the lack of mathematical symbols on computer keyboards - we use “computer-speak” to express the classic math operations
- Asterisk is multiplication
- Exponentiation (raise to a power) looks different than in math

Operator	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
**	Power
%	Remainder





More operators

Operator	Operation
not	Boolean-logical operators
and	
or	
< <=	Mixed (comparison) operators
> >=	
== !=	

Order of Evaluation

- When we string operators together, Python must know which one to do first
- This is called “operator precedence”
- Which operator “takes precedence” over the others?

$x = 1 + 2 * 3 - 4 / 5 ** 6$

Operator Precedence Rules PEMDAS

Highest precedence rule to lowest precedence rule:

- **Parentheses** are always respected
- **Exponentiation** (raise to a power)
- **Multiplication, Division, and Remainder**
- **Addition and Subtraction**
- **Left to right**

Parenthesis

Power

Multiplication, division, remainder

Addition/subtraction

Left to Right

Example

```
>>> x = 1 + 2 ** 3 / 4 * 5
>>> print(x)
11.0
>>>
```

Parenthesis

Power

Multiplication, division, remainder

Addition/subtraction

Left to Right



1 + 2 ** 3 / 4 * 5

1 + 8 / 4 * 5

1 + 2 * 5

1 + 10

11



Assignment expressions

Assignment in Python

`x = 5` ← **Assignment statement**

`x = x * 3` ← **Assignment with expression**

`print(x)` ← **Print statement**

- An assignment statement consists of an expression on the right-hand side and a variable to store the result.
- What is the output of this program?

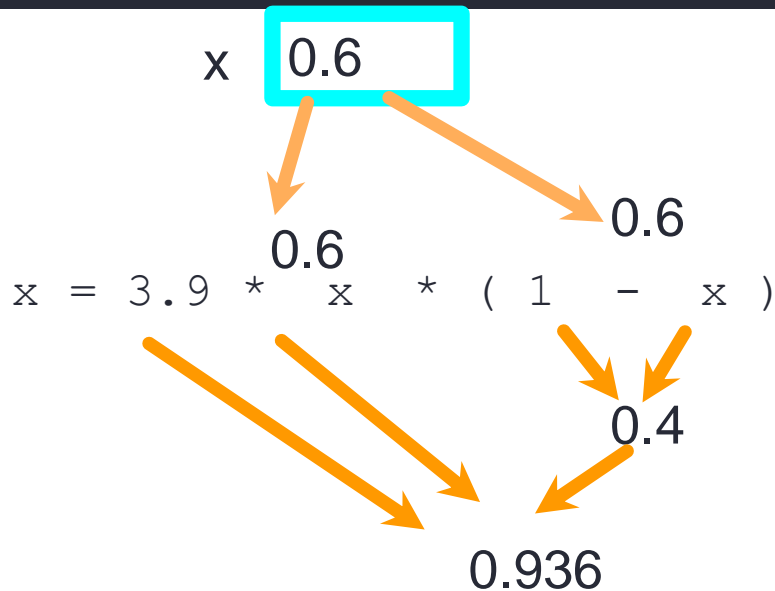
```
>>>x=0.6
```

```
>>>x = 3.9 * x * ( 1 - x )
```

```
>>>print(x)
```

Assignment expressions

- The right side is an expression.
Once the expression is evaluated, the result is placed in (assigned to) x.



A variable is a memory location used to store a value. The value stored in a variable can be updated by replacing the old value (**0.6**) with a new value (**0.936**).

x ~~0.6~~ 0.936

Try and check

1. Write a Python program which display the area of circle with radius of circle=1.1
2. Write Python Program to find Area and Perimeter Of a Rectangle
 1. Height=4.5
 2. Width=2.6

Solutions

1

```
pi=3.14
```

```
r = 1.1
```

```
Area=pi*r*r
```

```
print (Area)
```

OR

```
import math
```

```
r = 1.1
```

```
Area=math.pi*r**2
```

```
print (Area)
```

2

```
width = 2.6
```

```
height = 4.5
```

```
Area = width * height
```

```
Perimeter = 2 * (width + height)
```

```
print("\n Area of a Rectangle is:",  
Area)
```

```
print(" Perimeter of Rectangle is: ",  
Perimeter)
```




Data type



Standard Data type

- Data types: categorize value in memory
 - e.g., `int` for integer, `float` for real number, `str` used for storing strings in memory.
- Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.
- Python knows the difference between an integer number and a string.
- For example “+” means “addition” if something is a number and “concatenate” if something is a string.

Data type in Python

- Python has five standard data type:
 1. Numbers(integer, floats, complex)
 2. String (str)
 3. List
 4. Tuple
 5. Dictionary

```
>>> var = 4 + 6
>>> print(var)
10
>>> s="Good"+" "+"morning"
>>> print(s)
Good morning
>>>print('@'*5)
@@@@@
```

Data **type** in Python

- Python knows what “type” everything is.
- Some operations are prohibited.
- You cannot “add 1” to a string.
- We can ask Python what type something is by using the **type()** function:
- **type()** returns the type of any data value

```
>>> print(type(15))
<class 'int'>
>>> print(type(3.))
<class 'float'>
>>> x=34.8
>>> print(type(x))
<class 'float'>
>>> s=3 + 1j
>>> print(type(s))
<class 'complex'>
>>> x="learning"
>>> print(type(x))
<class 'str'>
```

Data type in Python

- Mixed type (integer and float) expressions are converted to floats:

```
>>> 4 * 2.0 / 6
```

```
1.3333333333333333
```

- Explicit casts are also supported:

```
>>> y = 4.999
```

```
>>> x = 8
```

```
>>> int(y)
```

```
>>> float(x)
```

```
4
```

```
8.0
```

- Integer division produces a floating point result:

```
>>> print(10 / 2)
```

```
>>> print(9 / 2)
```

```
>>> print(int(9/2))
```

```
5.0
```

```
4.5
```

```
4
```

Python Strings

- In Python string can be created simply by enclosing characters in the double quote. For example: "Hello world".
- We use square brackets[] for slicing along with the index or indices to obtain a substring.

```
var1="preparatory year"  
print("var1[0]:", var1[0])  
var1[0]: p  
print("var1[1:5]:", var1[1:5])  
var1[1:5]: repa  
print("var1[0:11]:", var1[0:11])  
var1[0:11]: preparatory  
print(var1[:6])  
Prepar  
print(var1[1:2])  
r
```

Try and check

1. Write a Python program to display the first character of your first name concatenated to the first character of your last name:

❖ **Sample string:** First name: Sarra , Last name: Ali

❖ **Expected output:** SA

```
my_name="Sara Ali"
```

```
print(my_name[0]+my_name[5])
```

```
F_name="Sara"          #input()
```

```
L_name="Ali"
```

```
print(F_name[0]+L_name[0])
```



Python Lists

- List is a collection which is ordered and changeable. Allows duplicate members.
- A list contains items separated by commas and enclosed within [].
- The values stored in a list can be accessed using the slice operator ([] and [:]) with indexes starting at 0 in the beginning of the list. Their elements and size can be changed.
- The plus(+) sign is the list concatenation operator and the asterisk (*) is the repetition operator.



Python Lists

```
List1=['abcd',123,2.5,'Ali','Sarrah',70.2]
```

```
List2=[125,'ahmad']
```

```
print (List1)
```

Output:['abcd', 123, 2.5, 'Ali', 'Sarrah', 70.2]

```
print(List2)
```

Output:[125, 'ahmad']

```
print (List1[0])
```

Output:abcd

```
print (List1[1:2])
```

Output:[123]

```
print (List2*2)
```

Output:[125, 'ahmad', 125, 'ahmad']

```
print (List1+List2)
```

Output:['abcd', 123, 2.5, 'Ali', 'Sarrah', 70.2, 125, 'ahmad']

#UPDATE AN ITEM OF THE LIST

```
List1[0]="hi"
```

```
print(List1)
```

Output:['hi', 123, 2.5, 'Ali', 'Sarrah', 70.2]




Add items to a Lists

- The **append()** method adds an item to the **end of the list**.
- The **insert()** method can add an element at a given position in the list. but like append(), it can add only one element at a time.

Mylist.insert(position,item to add)

Example:

```
currencies = ['Dollar', 'Euro', 'Pound']  
# append 'Sar' to the list  
currencies.append('sar')  
print(currencies)  
  
#insert at index 2 of the list "Dinar"  
currencies.insert(2,"dinar")  
print(currencies)
```



```
['Dollar', 'Euro', 'Pound', 'sar']  
['Dollar', 'Euro', 'dinar', 'Pound', 'sar']
```

List Operations in Python

- The **remove()** method is used to remove an element from the list. In the case of multiple occurrences of the same element, only the first occurrence is removed.

```
myList = [1,4, 2, 6, 5, 0, 1]
myList.remove(1)
print(myList)
#output:[4, 2, 6, 5, 0, 1]
```

- The **sort ()** method sorts the list in ascending order. This operation can only be performed on homogeneous lists, i.e. lists having elements of similar type.

```
myList = [4, 2, 6, 5, 0, 1]
myList.sort()
print(myList)
#output:[0, 1, 2, 4, 5, 6]
```

Solution:

1. Write a Python program to concatenate the 2nd, 4th and 5th elements of 2 lists

Sample lists: L1:red,green,black,white,pink,yellow

L2:black,blue,red,purple,orange

Expected output : green white pink blue purple orange

```
L1=['red','green','black','white','pink','yellow']
```

```
L2=['black','blue','red','purple','orange']
```

```
print(L1[1]+' '+L1[3]+' '+L1[4]+' '+L2[1]+' '+L2[3]+' '+L2[4])
```

Python Tuples

- A tuple consists of a number of values separated by commas. Unlike lists, tuples are **enclosed with parentheses ()** and their elements and size cannot be updated.
- Tuples can be thought of as **read-only lists**.
- `tuple1=('abcd',123,2.5,'Ali','Sarra',70.2)`

Python Dictionary

- A dictionary is used to map or associate things you want to store the keys you need to get them. A dictionary is a collection which is **ordered**, **changeable** and do **not allow duplicates**.
- Python Dictionary are **defined into two elements keys and values**.
- Dictionaries are used to store data values in **key:value** pairs
- **Keys will be a single element**
- **Values can be a list or list within a list, numbers, etc.**
- Example:

```
Dict={'Zained':18,'Ali':20,'sarra':22}  
print(Dict['sarra'])
```

22


-The Keys and values can be number or string
-Call element value by using key



Remove Items from a Dictionary

- The Python **del** statement deletes an object. Because key-value pairs in dictionaries are objects, you can delete them using the “del” keyword. The “del” keyword is used to delete a key that does exist. It raises a **KeyError** if a key is not present in a dictionary.
- The **del** keyword removes the item with the specified **key name** .
- **Example:**

```
1 car = {  
2     "brand": "BMW",  
3     "model": "X5",  
4     "year": 2022  
5 }  
6 print(car)  
7 del car["model"]  
8 print(car)
```



```
{'brand': 'BMW', 'model': 'X5', 'year': 2022}  
{'brand': 'BMW', 'year': 2022}
```



User Input



Input() function

- Most programs need to read input from the user.
- Built-in `input` function reads input from keyboard:
 - Returns the data as a **string**
 - Format: `variable = input(prompt)`
 - `prompt` is typically a string instructing user to enter a value
- Does not automatically display a space after the prompt.
- Built-in functions convert between data types:
 - `int(item)` converts `item` to an `int`
 - `float(item)` converts `item` to a `float`

Input()

Inputs a string

```
>>> name = input("enter a  
name")  
    >>> print(name)  
    Ali  
    >>> number = input("Enter  
        an integer")  
    >>> print(number)  
    '32'
```

Inputs an integer

```
number = int(input("enter an  
integer: "))  
>>> print(number)  
87
```

Inputs a float

```
F = float(input("enter a  
decimal: "))  
>>> print(F)  
2.5
```



Input() function

- If we want to read a number from the user, we must convert it from a string to a number using a type conversion function
- If types don't match (e.g., if you type 4.5 and try to cast it as an integer) you will get an error:



Multiple inputs:

```
>>> x = int(input("enter an integer: "))  
      y=float(input("enter a float: "))
```

```
enter an integer: 3
```

```
enter a float: 4.5
```

```
>>> print("x is", x, " y is ", y)
```

```
x is 3  y is  4.5
```

- Instead of the cast you can use the eval() function and Python choose the correct types:

```
>>> x, y = eval(input("Enter two numbers: "))
```

```
Enter two numbers: 3.7, 98
```

```
>>> x, y  
(3.7, 98)
```

Try and check

1. Write a Python program which prompts the user for the radius of a circle and compute the area.
2. Write Python program to find the Area and the Perimeter Of a Rectangle. The values of height and width are given by the user.
3. Write Python Program to convert temperature in celsius to Fahrenheit. The degree in Celsius is given by the user.

$$\text{Fahrenheit} = (\text{celsius} * 1.8) - 32$$

4. Write a Python program to Calculate Simple Interest of loan.

$$\text{Simple Interest} = (\text{Principal Amount} * \text{Rate of Interest} * \text{Number of years}) / 100$$



Try and check

Python Program to find Area Of Circle using Radius and Circumference Of a Circle

PI = 3.14

radius = float(input(' Please Enter the radius of a circle: '))

area = PI * radius * radius

circumference = 2 * PI * radius

print(" Area Of a Circle = %.2f" %area)

print(" Circumference Of a Circle =

Python Program to find Area and Perimeter of a Rectangle

width = float(input('Please Enter the Width of a Rectangle: '))

height = float(input('Please Enter the Height of a Rectangle: '))

calculate the area

Area = width * height

calculate the Perimeter

Perimeter = 2 * (width + height)

print("\n Area of a Rectangle is: %.2f" %Area)

print(" Perimeter of Rectangle is: %.2f"

%Perimeter)



Try and check

#Python Program to convert
temperature in celsius to fahrenheit

```
celsius = float(input('Please Enter the  
degree Celsius: '))
```

```
# calculate fahrenheit
```

```
fahrenheit = (celsius * 1.8) + 32
```

```
print('%0.1f degree Celsius is equal to  
%0.1f degree Fahrenheit'
```

```
%(celsius,fahrenheit))
```

Python Program to Calculate Simple
Interest

```
princ_amount = float(input(" Please Enter  
the Principal Amount : "))
```

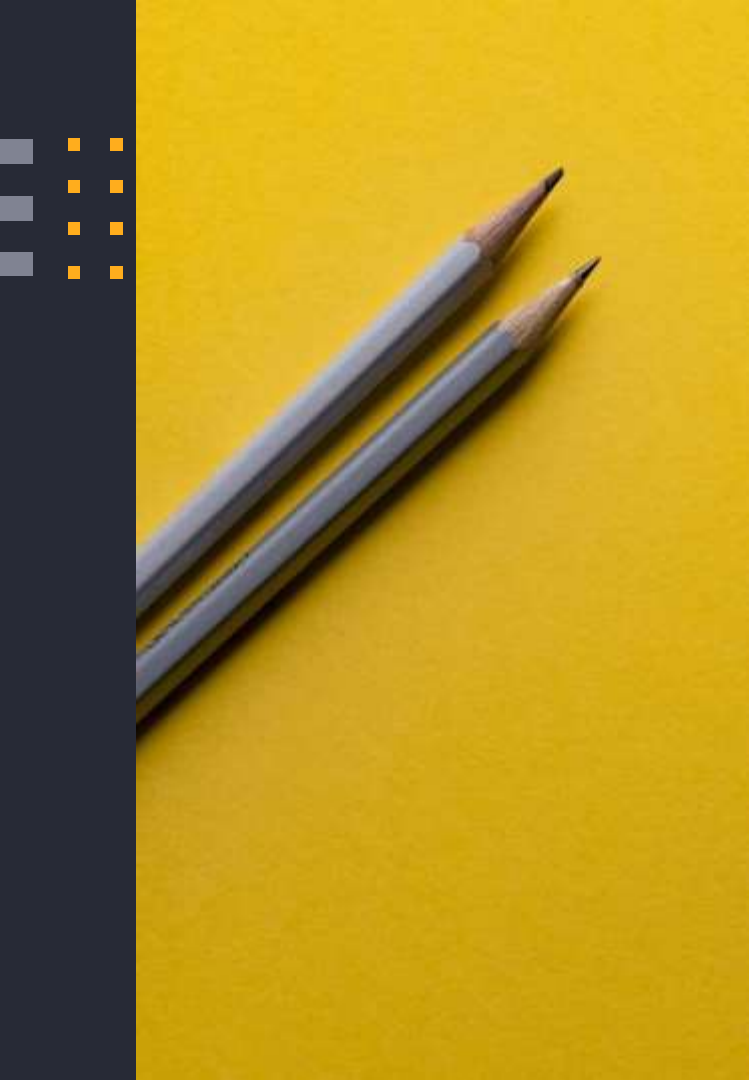
```
rate_of_int = float(input(" Please Enter  
the Rate Of Interest  : "))
```

```
time_period = float(input(" Please Enter  
Time period in Years  : "))
```

```
simple_interest = (princ_amount *  
rate_of_int * time_period) / 100
```

```
print("\n simple interest=",  
simple_interest)
```





Thanks!

Any questions?