[CS-468] Lab Manual - 1

June 16, 2021

Python Fundamentals

1 What is Python?

Python is a high-level programming language, and its core design philosophy is all about code readability and a syntax which allows programmers to express concepts in a few lines of code.!

Python is designed by Guido Von Rossum

1.1 Python Advantages

- Readable
- Productive
- Portable
- Rich Libraries
- Ease of Learning

1.2 Use Cases of Python

- Artificial Intelligence
- Web Development
- Data Analysis
- IoT
- Web Scraping
- Computer Vision
- Machine Learning
- Game Development
- Desktop Development

2 Print Statement

We use the `print()` function when you want to print a message/output to console. The `print()` function actually tries to print whatever you pass to it: - when it is given a *string* it will print a *string* - if it is given an *integer* such as 44 it will print 44 - if it is given a *floating* point number such as 20.3 it will print 20.3

[2]:
```
# print a message
print("Welcome to CS-468 Artificial Intelligence Lab")
```

```
Welcome to CS-468 Artificial Intelligence Lab
```

[3]:
```
# another example of printing a message
print('This message is in single quoted string')
```

```
This message is in single quoted string
```

Note: You can use either single quotes or double quotes to display textual information. As a good practice, just stick to one convention. Double quotes are preferred.

[4]:
```
# printing an integer
print(22)
```

```
22
```

[5]:
```
# printing a floating point number
print(33.3333333)
```

```
33.3333333
```

3 Comments

A programmer's comment (usually called a comment for short) is text in the program that does nothing. Which brings up the question . . ." If it doesn't do anything, then why type it in?" As a learner, you can use comments in your code as notes to yourself about what the code is doing. These can help a lot when you're first learning.

- Comments are a way of documenting your code
- Comments are ignored by Python
- A comment can help you figure out your code when you come back to it a month or a year later

3.0.1 Syntax

- Single Line Comments start with a pound `#` sign
- Multi-line comments must be enclosed in `"""` triple quotes

[6]:
```
# this is an example of single line comment
```

[ ]:
```
# print("Hey!")
```

[8]:
```
"""
This is a multiline comment in Python
This type of comment is sometimes called a docstring.
A docstring starts with three double-quotation marks and also ends with
three double quotation marks
"""
```

[8]:
```
'\nThis is a multiline comment in Python\nThis type of comment is sometimes
called a docstring.\nA docstring starts with three double-quotation marks and
also ends with\nthree double quotation marks\n'
```

Understanding Python Data Types

A data type is a **set of values** and a **set of operations** defined on them.

Some common data types in Python are: - Integer `<int>` (2, -3, 0, 20203) - Float `<float>` (3.2, 6.3, -9.333) - Boolean `<bool>` (True, False) - String `<str>` ('Hi', 'a' , "String") - Complex `<complex>` (1+2j)

3.1 Numbers and Arithmetics

Python has two basic number types, integer and float

For example, 2 is an integer and 2.0 is a floating point number which has a decimal attached to it. We can perform arithmetic operations to either of these numbers types. Let's try!

3.1.1 Arithmetic Operators

Symbol	Operator	Description
+	Addition	Adds two or more numbers
-	Subtraction	Subtracts one number from another
*	Multiplication	Multiplies one number by another
/	Division	Divides one number by another and gives the result as a floating-point number
//	Integer Division	Divides one number by another and gives the result as an integer
%	Remainder	Divides one number by another and gives the remainder
**	Exponenet	Raises a number to a power

[1]:
```
# addition
5 + 2
```

[1]: 7

[2]:
```
# Subtraction
5 - 2
```

[2]: 3

[3]:
```
# multiplication
5 * 2
```

[3]: 10

With a forward slash, we can divide numbers in Python (note, two integers are giving floating point in results).

If you are using Python 2, you need to add 2.0 in the expression (e.g. 5 / 2.0) to get the same results.

[6]:
```
# Division
5/2
```

[6]: 2.5

[5]:
```
# Integer Division
5//2
```

[5]: 2

We can compute the power of a number (exponent) with two Asterix ** together

[5]:
```
5 ** 2
```

[5]: 25

Python follow the order of arithmetic operations, for example: for 1 + 2 * 3 + 4 Python will first multiply 2 & 3 then perform the other operations.

[6]:
```
1 + 2 * 3 + 4
```

[6]: 11

A good practice is to use parentheses "( )" to tell the Python, which operation needs to be performed and to clarify the order. The operations in the parentheses will be performed first.

[7]:
```
(1 + 2) * (3 + 4)
```

[7]: 21

Modulus or **mod** operation is a "%" (percentage sign) in Python. It returns what remains after a division. *With mod operation, we can check if the number is even or odd. Mod (%) returns 0, for even number and returns 1 for odd numbers.*

[8]:
```
6 % 2 # returns 0, 6 is even no.
```

[8]: 0

[9]:
```
5 % 2 # returns 1, 5 is odd no. (if we divide 5 by 2, remainder is 1)
```

[9]:
```
1
```

4 Variables:

- Variables are containers for storing data values.
- Unlike other programming languages, Python has no command for declaring a variable.
- A variable is created the moment you first assign a value to it.

At many occasions, we pick a variable with some name and assign object or a data type to that variable. We can do this in Python using equal sign operator "=". For example, if we assign a value of 5 to a variable "x", whenever we call x, this will return 5 in the output.

It is a common practice to create a **variable with multiple names**. A good way is to separate them with underscore **"_" between the words**. By doing this, you will easily identify the variables e.g. total_profit, total_loss, first_name etc.

[10]:
```
name_of_the_variable = 20
```

[11]:
```
name_of_the_variable
```

[11]:
```
20
```

Important regarding variable name: Before we move on to the next data type, we should know that, in Python: * variable name can not be started with number (e.g. 1var, 2x etc) * variable name can not be started with special characters (e.g. !, @, #,!y,+,- etc) * variable name are case sensitive, "Name" is not the same as "name" * reserved words can not be a variable name, e.g. "class" is a reserve word in Python, it can not be used as a variable name, but "klass" and "Class" work fine.

[12]:
```
# Let's try x = 5 and y = 3
x = 5
y = 3
```

[13]:
```
# what is x now!
x
```

[13]:
```
5
```

[14]:
```
# what is y now!
y
```

[14]:
```
3
```

4.1 Assignment Operator

The = sign in the assignment operator is so named because it assigns the value (on the right) to the variable (on the left). For example:

[ ]:
```
a = 20
```

In the above example, we are storing the number 20 in a variable named `x`. Or, in other words, we're assigning the value 10 to the variable named `x`

4.1.1 Simultaneous Assignment

Python allows simultaneous assignment.

Syntax

```
var1, var2, var3,... = val1, val2, val3,...
```

This is called simultaneous assignment. Semantically, this tells Python to evaluate all the expressions on the right-hand side and then assign these values to the corresponding variables named on the left-hand side!

We can perform arithmetic operations using these variable.

[15]:
```
# Adding x and y
x + y
```

[15]: 8

[16]:
```
# Multiplying x and y
x * y
```

[16]: 15

After performing any arithmetic operation, we can assign the result to a new variable.

[17]:
```
# Subtraction of the variables and assigning the result to a new variable
z = x - y        # 5 - 3
```

[18]:
```
z
```

[18]: 2

We can re-assign a value to the same variables. This will replace the existing value to its new value.

[19]:
```
x = y + 2
x = x * x
```

[20]:
```
x
```

[20]: 25

Let's try some invalid variable names!

[21]:
```
1var = 1
```

```
  File "<ipython-input-21-460332d99e8f>", line 1
    1var = 1
    ^
SyntaxError: invalid syntax
```

[ ]:
```
$var = 3
```

5 Interactive Hello World

Let us make our program a little more interesting; lets get it to ask us our name and say hello to us personally.

`input()` is a built-in function that is a part of Python language. As the name suggests, `input()` function is used to take input from the user. Generally, we pass a string to the input function which is used as a prompt message.

Note: The `input()` function is a value-returning function and it always returns a `string`

[1]:
```
# Ask the user to enter name
name = input("Enter your name: ")
```

```
Enter your name: Rumman
```

[3]:
```
# Display greeting message
print("Hello", name)
```

```
Hello Rumman
```

5.1 Booleans

Booleans are simply True and False with capital T and Capital F – just a customized version of 1 and 0.

[ ]:
```
True
```

[ ]:
```
False
```

[ ]:
```
10 == 10
```

[ ]:
```
5 == 10
```

[ ]:
```
a = 10
b = 5

a == b
```

6 Comparison Operators

Comparison Operators allows to compare two elements to each other, such as, greater than, less than, equal to, not equal to etc.

Operator	Usage	Description
==	x == y	x is equal to y
!=	x != y	x is not equal to y
>	x > y	x is greater than y
<	x < y	x is less than y
>=	x >= y	x is greater than or equal to y
<=	x<=	x is less than or equal to y

[22]:
```
x = 10
y = 20
```

[23]:
```
x == y # x is equal to y
```

[23]:
```
False
```

[24]:
```
x != y # x is not equal to y
```

[24]:
```
True
```

[25]:
```
x > y # x is greater than y
```

[25]:
```
False
```

[26]:
```
x < y # x is less than y
```

[26]:
```
True
```

[27]:
```
x >= y # x is greater than or equal to y
```

[27]:
```
False
```

[28]:
```
x <= y # x is less than or equal to y
```

[28]:
```
True
```

6.1 Logical Operators

There are three logical operators: and, or, and not

and Operator		
A	B	A and B
True	True	True
True	False	False
False	True	False
False	False	False

or Operator		
A	B	A or B
True	True	True
True	False	True
False	True	True
False	False	False

not Operator	
A	not A
True	False
False	True

[31]:
```
x = 10
print (x > 0 and x < 10)
```

```
False
```

[32]:
```
print (not (x%2 == 0))
```

```
False
```

[1]:
```
print (10 or True) # 10
print (10 and True) # True
print (10 and False) # False
print (10 or False) # 10
```

```
10
True
False
10
```

7 Exercise

7.1 Sales Tax

Write a program that will ask the user to enter the amount of a purchase. The program should then compute the state and county sales tax. Assume the state sales tax is 5 percent and the county sales tax is 2.5 percent. The program should display the following: * The amount of the purchase * The state sales tax * The county sales tax * The total sales tax * and the total of the sale (which is the sum of the amount of purchase plus the total sales tax).

7.2 Tip, Tax and Total

Write a program that calculates the total amount of a meal purchased at a restaurant. The program should ask the user to enter the charge for the food, and then calculate the amount of a 18 percent tip and 7 percent sales tax. Display each of these amounts and the total.

7.3 Ingredient Adjuster

A cookie recipe calls for the following ingredients: * 1.5 cups of sugar * 1 cup of butter * 2.75 cups of flour

The recipe produces 48 cookies with this amount of the ingredients. Write a program that asks the user how many cookies he or she wants to make, and then displays the number of cups of each ingredient needed for the specified number of cookies.

7.4 Male and Female Percentages

Write a program that asks the user for the number of males and the number of females registered in a class. The program should display the percentage of males and females in the class.

Hint: Suppose there are 8 males and 12 females in a class. There are 20 students in the class. The percentage of males can be calculated as 8 4 20 5 0.4, or 40%. The percentage of females can be calculated as 12 4 20 5 0.6, or 60%.

7.5 Stock Transation Program

Last month Joe purchased some stock in Acme Software, Inc. Here are the details of the purchase: * The number of shares that Joe purchased was 2,000. * When Joe purchased the stock, he paid 40.00 Dollar per share. * Joe paid his stockbroker a commission that amounted to 3 percent of the amount he paid for the stock.

Two weeks later Joe sold the stock. Here are the details of the sale: * The number of shares that Joe sold was 2,000. * He sold the stock for 42.75 Dollar per share. * He paid his stockbroker another commission that amounted to 3 percent of the amount he received for the stock.

Write a program that displays the following information: * The amount of money Joe paid for the stock. * The amount of commission Joe paid his broker when he bought the stock. * The amount that Joe sold the stock for. * The amount of commission Joe paid his broker when he sold the stock. * Display the amount of money that Joe had left when he sold the stock and paid his broker (both times). If this amount is positive, then Joe made a profit. If the amount is negative, then Joe lost money.