LAB ASSIGNMENT 01

COMSATS University Islamabad
Sahiwal Campus

Usama Sarwar

FA17-BS(CS)-090-B

Mr. Waqar

Artificial Intelligence

October 12, 2020

Table of Contents

1. Artificial Intelligence

Artificial intelligence (AI) is the simulation of human intelligence in machines that are programmed to think and act like humans by mimicking their actions. The term may likewise be applied to any machine that shows qualities related with a human psyche, for example, learning and critical thinking. The ideal trait of Artificial Intelligence is its capacity to justify and take activities that have the most obvious opportunity with regards to accomplishing a particular objective.

2. Machine learning VS Deep Learning VS artificial Intelligence

Artificial Intelligence is a science like arithmetic or science. It contemplates approaches to fabricate insightful projects and machines that can innovatively take care of issues, which has consistently been viewed as a human right.

Machine learning is a subset of artificial intelligence (AI) that gives frameworks the capacity to consequently take in and improve for a fact without being expressly customized. In ML, there are various calculations (for example neural organizations) that help to tackle issues. It has techniques that helps computers to figure things out and deliver the outcome to AI.

Deep Learning is a subset of machine learning, which utilizes the neural organizations to investigate various variables with a structure that is like the human neural framework. It helps computers to solve more complex problems.

3. Problem solving technique

According to psychology, "*a problem-solving refers to a state where we wish to reach to a definite goal from a present state or condition.*

According to computer science, *a problem-solving is a part of artificial intelligence which encompasses a number of techniques such as algorithms, heuristics to solve a problem.*

3.1 Problem Searching

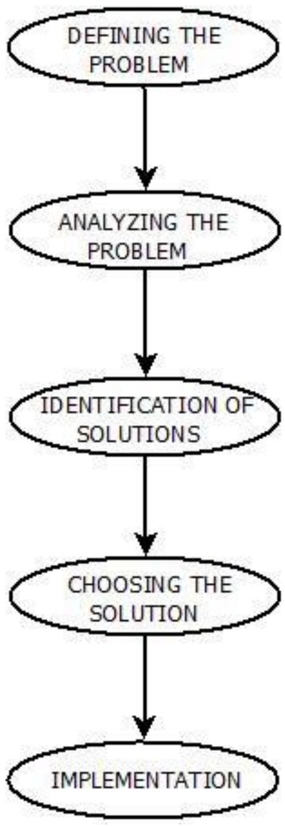
- In general, searching refers to as finding information one needs.
- Searching is the easiest and commonly used technique of problem solving in artificial intelligence.
- Searching algorithm helps us to search for solution of particular problem.

3.1.1 Problem

- Problems are the issues which comes across any system at any point of time. A solution is always needed to solve that particular problem.

3.2 Steps involve in Solving Problems Using Artificial Intelligence

- The process of solving a problem consists of five steps. These are:



Problem Solving in Artificial Intelligence

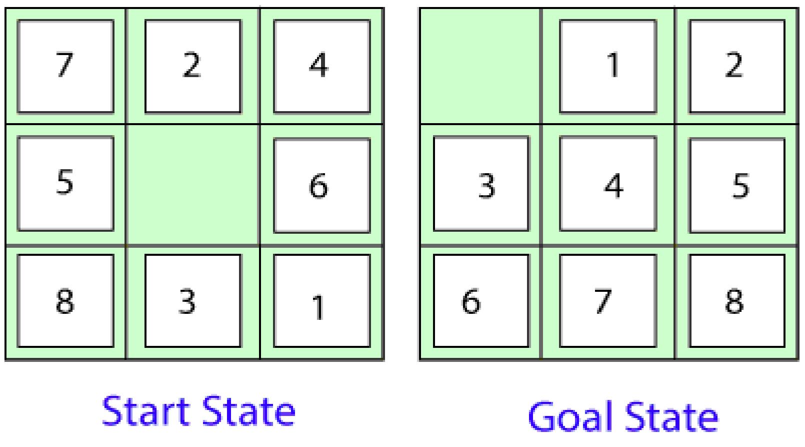
Defining the Problem: The definition of the problem must be included precisely. It should contain the possible initial as well as final situations which should result in acceptable solution.

1. **Analyzing the Problem:** Analyzing the problem and its requirement must be done as few features can have immense impact on the resulting solution.
2. **Identification of Solutions:** This phase generates reasonable amount of solutions to the given problem in a particular range.
3. **Choosing a Solution:** From all the identified solutions, the best solution is chosen basis on the results produced by respective solutions.
4. **Implementation**: After choosing the best solution, its implementation is done.

3.3 Example:

3.3.1 8 Puzzle Problem:

- Here, we have a 3×3 matrix with movable tiles numbered from 1 to 8 with a blank space. The tile adjacent to the blank space can slide into that space. The objective is to reach a specified goal state similar to the goal state, as shown in the below figure.
- In the figure, our task is to convert the current state into goal state by sliding digits into the blank space.



In the above figure, our task is to convert the current (Start) state into goal state by sliding digits into the blank space.
The problem formulation is as follows:

- **States:** It describes the location of each numbered tiles and the blank tile.
- **Initial State:** We can start from any state as the initial state.
- **Actions:** Here, actions of the blank space are defined, i.e., either **left, right, up, or down**
- **Transition Model:** It returns the resulting state as per the given state and actions.
- **Goal test:** It identifies whether we have reached the correct goal-state.
- **Path cost:** The path cost is the number of steps in the path where the cost of each step is 1.

4. IDE for Python

An IDE (or Integrated Development Environment) is a program devoted to programming development. As the name suggests, IDEs coordinate a few devices explicitly intended for programming development. These devices normally include:

- An editor intended to deal with code (with, for instance, language structure featuring and auto-fulfillment)
- Construct, execution, and investigating devices
- Some type of source control

There are two major categories of Python IDEs as follows:

- General Editors and IDEs with Python Support

- Python Specific Editors and IDEs

4.1 Comparison Chart

IDE	Price	Supported Platform	Category
Pycharm	US $ 199 per User – 1st year for Professional Developer.	WINDOWS, LINUX, MAC etc.	Python Specific
Spyder	Open Source	QT, WINDOWS, LINUX, MAC OS etc.	Python Specific
PyDev	Open Source	QT, WINDOWS, LINUX, MAC OS etc.	General
Idle	Open Source	WINDOWS, LINUX, MAC OS etc.	General
Wing	US $ 95 to US $ 179 PER USER FOR COMMERCIAL USE.	WINDOWS, LINUX, MAC OS etc.	General
Eric Python	Open Source	WINDOWS, LINUX, MAC OS etc.	General
Rodeo	Open Source	WINDOWS, LINUX, MAC OS etc.	General
Sublime Text	USD $80	WINDOWS, LINUX, MAC OS etc.	General
Atom	Open Source	WINDOWS, LINUX, MAC OS etc.	General
Vim	Open Source	WINDOWS, LINUX, Mac OS, IOS, Android, UNIX, AmigaOS, MorphOS etc.	General
Visual Studio Code	Open Source	WINDOWS, LINUX, Mac OS etc.	General

5. AIML (Artificial Intelligence Markup Language)

AIML stands for Artificial Intelligence Modelling Language. AIML is an XML based markup language meant to create artificial intelligent applications. AIML makes it possible to create human interfaces while keeping the implementation simple to program, easy to understand and highly maintainable.

5.1 Use of AIML

- ✓ AIML stands for Artificial Intelligence Modelling Language.
- ✓ AIML is an XML based markup language meant to create artificial intelligent applications.
- ✓ AIML makes it possible to create human interfaces while keeping the implementation simple to program, easy to understand and highly maintainable.

5.2 Basic tags of AIML

- **<aiml>** − defines the beginning and end of a AIML document.

```
<aiml version = "1.0.1" encoding = "UTF-8"?>
   ...
</aiml>
```

- **<category>** − defines the **unit of knowledge** in Alicebot's knowledge base.
- **<pattern>** − defines the pattern to match what a user may input to an Alicebot.
- **<template>** − defines the response of an Alicebot to user's input.

```
<category>
  <pattern> Usama Sarwar </pattern>
 <template>Hello World </template>
</category>
```

6. Data types in python

6.1 Numeric

It is used to represent data which is in numeric form. Three types of numeric data:

Integers(int): Positive or negative whole numbers without decimal part **i.e.** 1,2,3 ,etc/

Float: Positive or negative numbers with decimal part **e.g.** 10.1, 30.5, etc

Complex Numbers: A number with a real and imaginary component represented as x+yj. x and y are floats and j is -1(square root of -1 called an imaginary number).

6.2 Boolean

Data with one of two built-in values True or False. true and false are not valid booleans and Python will throw an error for them.

6.3 Sequence

A sequence is an ordered collection of similar or different data types. Python has the following built-in sequence data types:

- **String**: A string value is a collection of one or more characters put in single, double or triple quotes. **E.g**. Usama Sarwar, etc.
- **List** : A list object is an ordered collection of one or more data items, not necessarily of the same type, put in square brackets. **E.g.** [1,2,4], etc.
- **Tuple**: A Tuple object is an ordered collection of one or more data items, not necessarily of the same type, put in parentheses. **E.g. (**a,b,c**) , etc.**

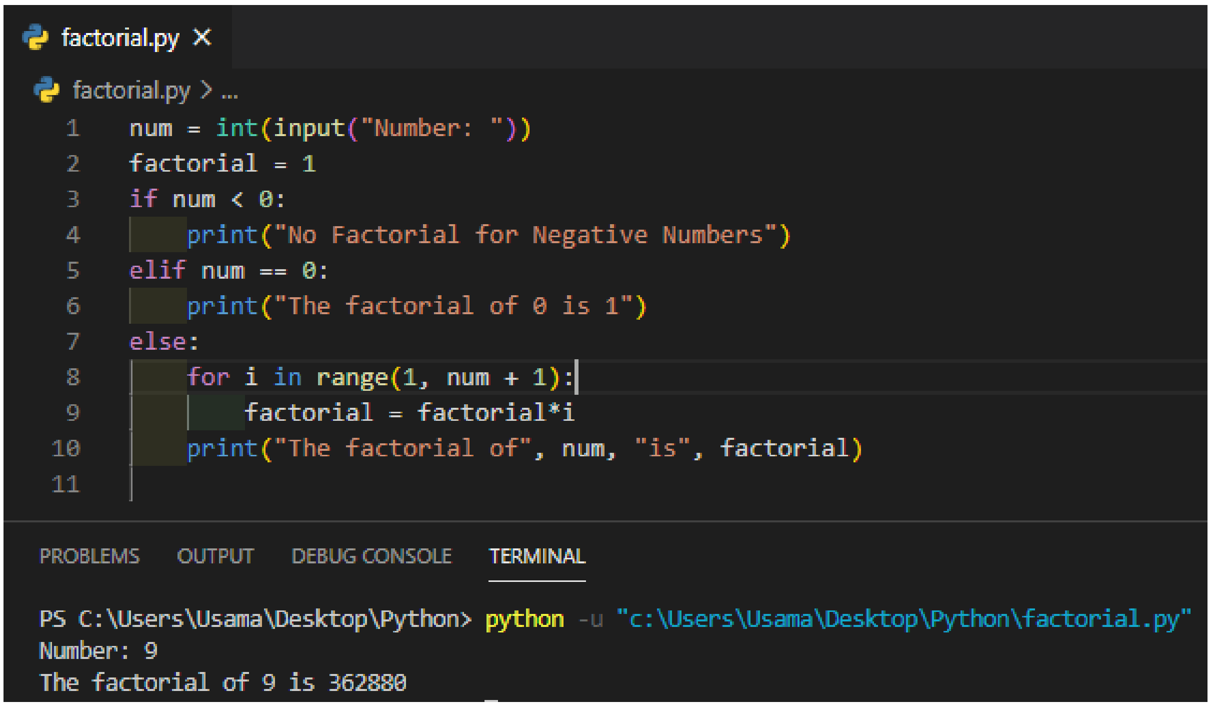
6.4 Dictionary

A dictionary object is an unordered collection of data in a key:value pair form. A collection of such pairs is enclosed in curly brackets. For example: {1:"Steve", 2:"Bill", 3:"Ram", 4: "Farha"}

7. Programs in python

7.1 Take a number from user and find the Factorial of this Number

```python
num = int(input("Number: "))
factorial = 1
if num < 0:
    print("No Factorial for Negative Numbers")
elif num == 0:
    print("The factorial of 0 is 1")
else:
    for i in range(1, num + 1):
        factorial = factorial*i
    print("The factorial of", num, "is", factorial)
```


factorial.py ×

factorial.py > ...
```python
num = int(input("Number: "))
factorial = 1
if num < 0:
    print("No Factorial for Negative Numbers")
elif num == 0:
    print("The factorial of 0 is 1")
else:
    for i in range(1, num + 1):
        factorial = factorial*i
    print("The factorial of", num, "is", factorial)
```

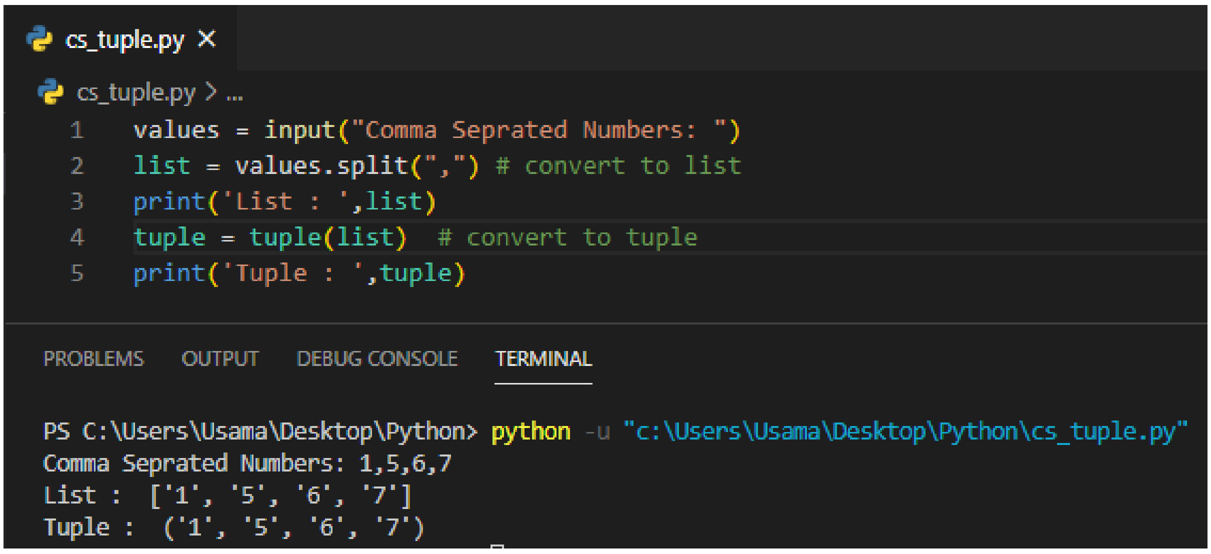
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS C:\Users\Usama\Desktop\Python> python -u "c:\Users\Usama\Desktop\Python\factorial.py"
Number: 9
The factorial of 9 is 362880
```

7.2 Write a Python program which accepts a sequence of comma-separated numbers from user and generate a list and a tuple with those numbers. Also update tuple elements.

Tuples can not be updated.

```python
values = input("Comma Seprated Numbers: ")
list = values.split(",") # convert to list
print('List : ',list)
tuple = tuple(list)  # convert to tuple
print('Tuple : ',tuple)
```


cs_tuple.py ×

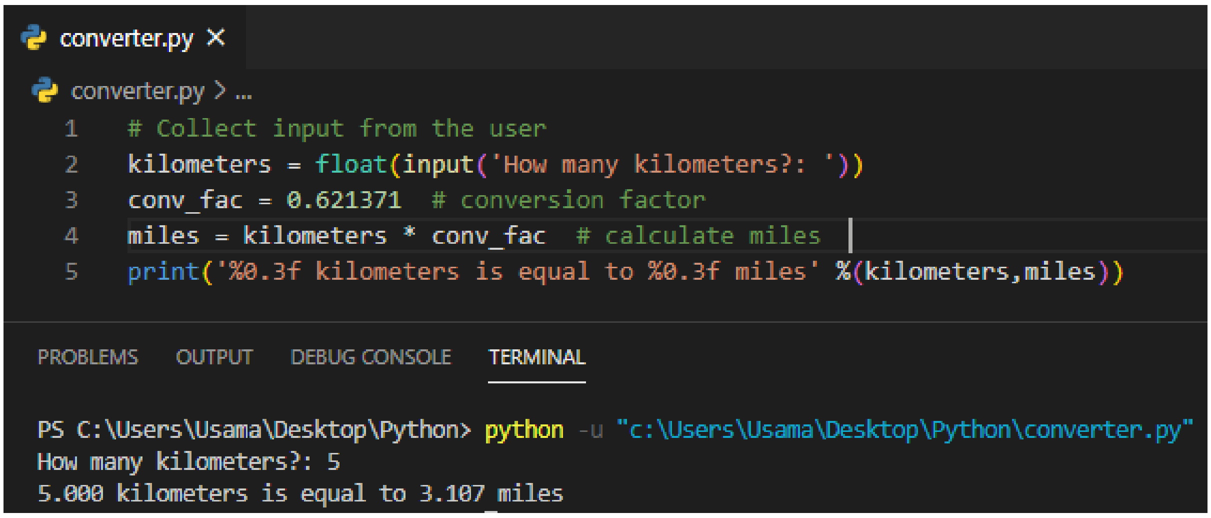
cs_tuple.py > ...
```python
values = input("Comma Seprated Numbers: ")
list = values.split(",") # convert to list
print('List : ',list)
tuple = tuple(list)  # convert to tuple
print('Tuple : ',tuple)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS C:\Users\Usama\Desktop\Python> python -u "c:\Users\Usama\Desktop\Python\cs_tuple.py"
Comma Seprated Numbers: 1,5,6,7
List :  ['1', '5', '6', '7']
Tuple :  ('1', '5', '6', '7')
```

7.3 Write a program to take kilometers from the user convert it into miles

```
# Collect input from the user
kilometers = float(input('How many kilometers?: '))
conv_fac = 0.621371  # conversion factor
miles = kilometers * conv_fac  # calculate miles
print('%0.3f kilometers is equal to %0.3f miles' %(kilometers,miles))
```

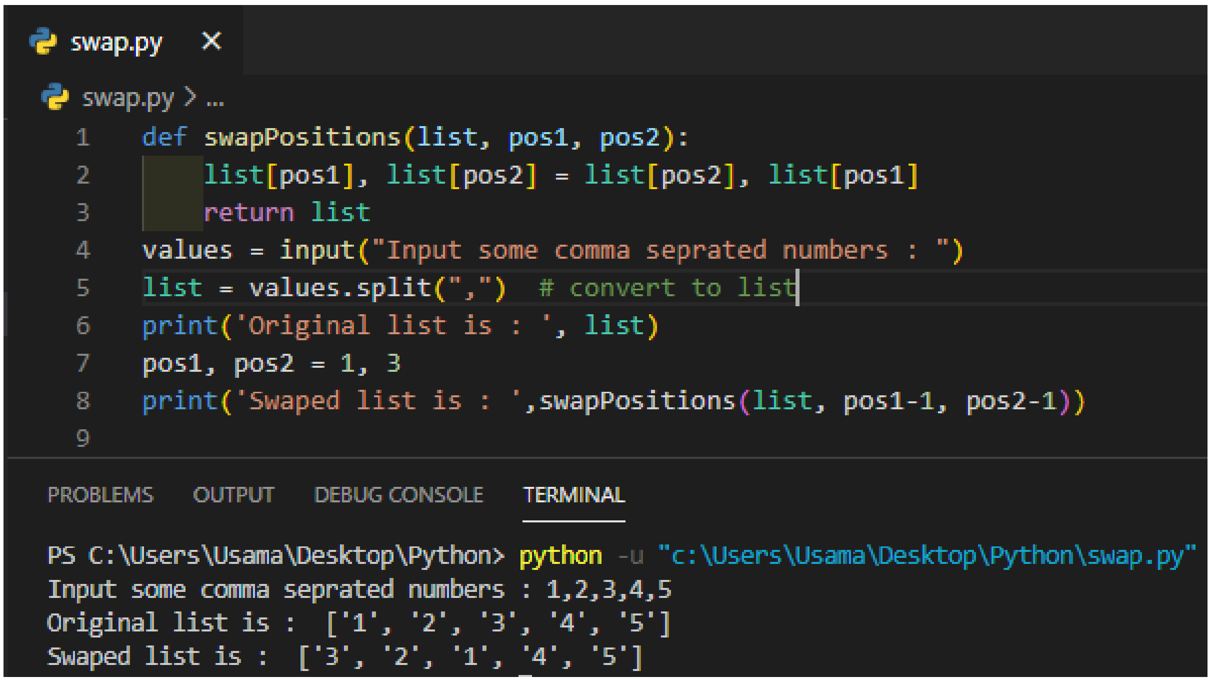


```
# Collect input from the user
kilometers = float(input('How many kilometers?: '))
conv_fac = 0.621371  # conversion factor
miles = kilometers * conv_fac  # calculate miles
print('%0.3f kilometers is equal to %0.3f miles' %(kilometers,miles))
```

```
PS C:\Users\Usama\Desktop\Python> python -u "c:\Users\Usama\Desktop\Python\converter.py"
How many kilometers?: 5
5.000 kilometers is equal to 3.107 miles
```

7.4 Write a program to swap two elements in a list

```
def swapPositions(list, pos1, pos2):
    list[pos1], list[pos2] = list[pos2], list[pos1]
    return list
values = input("Input some comma seprated numbers : ")
list = values.split(",")  # convert to list
print('Original list is : ', list)
pos1, pos2 = 1, 3
print('Swaped list is : ',swapPositions(list, pos1-1, pos2-1))
```



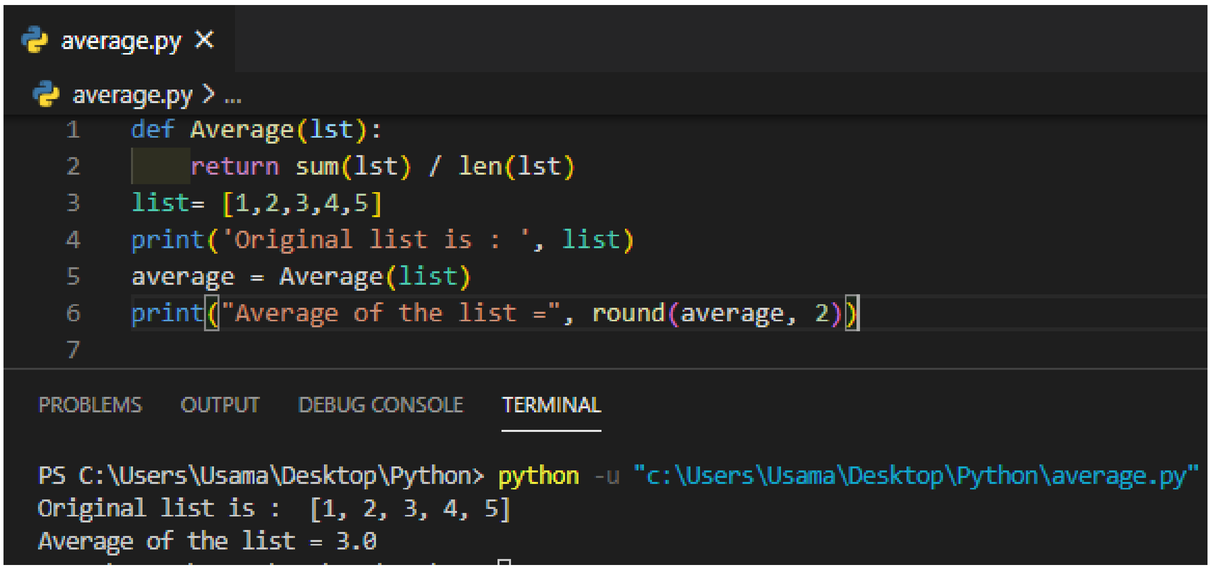
```
swap.py ×
swap.py > ...
def swapPositions(list, pos1, pos2):
    list[pos1], list[pos2] = list[pos2], list[pos1]
    return list
values = input("Input some comma seprated numbers : ")
list = values.split(",")  # convert to list
print('Original list is : ', list)
pos1, pos2 = 1, 3
print('Swaped list is : ',swapPositions(list, pos1-1, pos2-1))
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

PS C:\Users\Usama\Desktop\Python> python -u "c:\Users\Usama\Desktop\Python\swap.py"
Input some comma seprated numbers : 1,2,3,4,5
Original list is :  ['1', '2', '3', '4', '5']
Swaped list is :  ['3', '2', '1', '4', '5']
```

7.5 The program takes the elements of the list one by one and displays the average of the elements of the list

```
def Average(lst):
    return sum(lst) / len(lst)
list= [1,2,3,4,5]
print('Original list is : ', list)
average = Average(list)
print("Average of the list =", round(average, 2))
```



```
average.py ×
average.py > ...
def Average(lst):
    return sum(lst) / len(lst)
list= [1,2,3,4,5]
print('Original list is : ', list)
average = Average(list)
print("Average of the list =", round(average, 2))
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

PS C:\Users\Usama\Desktop\Python> python -u "c:\Users\Usama\Desktop\Python\average.py"
Original list is :  [1, 2, 3, 4, 5]
Average of the list = 3.0
```

7.6 Write a python program to implement List operations

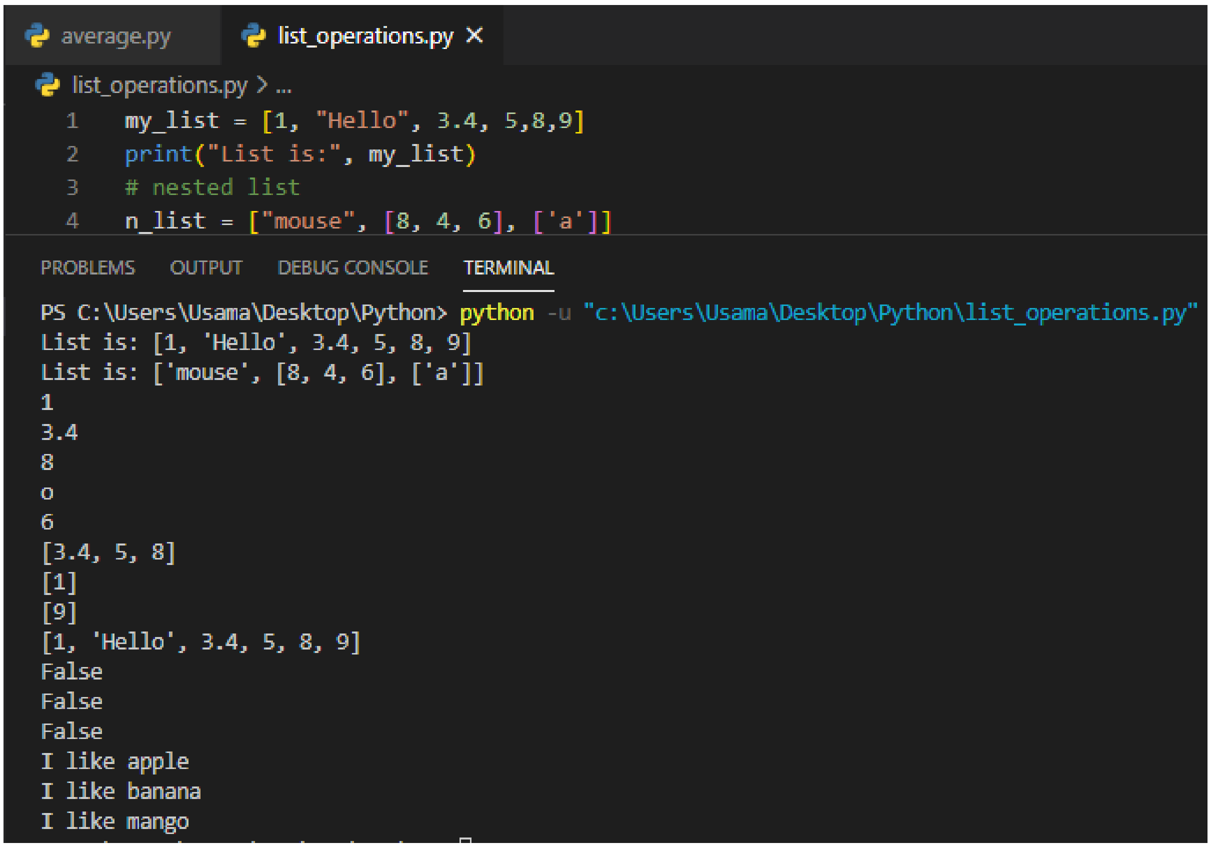
(Nested List, Length, Concatenation, Membership, Iteration, Indexing and Slicing)

```
my_list = [1, "Hello", 3.4, 5,8,9]
print("List is:", my_list)
# nested list
n_list = ["mouse", [8, 4, 6], ['a']]
print("List is:", n_list)
# List indexing
print(my_list[0])
print(my_list[2])
print(my_list[4])
# Nested List
# Nested indexing
print(n_list[0][1])
print(n_list[1][2])

# List slicing in Python
# elements 3rd to 5th
print(my_list[2:5])
#elements beginning to 4th
print(my_list[:-5])
# elements 6th to end
print(my_list[5:])
# elements beginning to end
print(my_list[:])

# List Membership using in operator
print('mouse' in my_list)
print('a' in my_list)
print(9 not in my_list)

#iteration
for fruit in ['apple','banana','mango']:
    print("I like",fruit)
```


average.py list_operations.py ×

list_operations.py > ...

```python
my_list = [1, "Hello", 3.4, 5,8,9]
print("List is:", my_list)
# nested list
n_list = ["mouse", [8, 4, 6], ['a']]
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS C:\Users\Usama\Desktop\Python> python -u "c:\Users\Usama\Desktop\Python\list_operations.py"
List is: [1, 'Hello', 3.4, 5, 8, 9]
List is: ['mouse', [8, 4, 6], ['a']]
1
3.4
8
o
6
[3.4, 5, 8]
[1]
[9]
[1, 'Hello', 3.4, 5, 8, 9]
False
False
False
I like apple
I like banana
I like mango
```