1. Explain features of python

1) interactive: You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
2) Easy to learn and use: Python is easy to learn and use. It is developer-friendly and high level programming language.
3) Expressive language: Python language is more expressive means that it is more understandable and readable.
4) Interpreted language: Python is an interpreted language, ie., interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.
5) Platform independent: Python can run equally on different platforms such as Windows, Linux, Unix and Macintosh etc. So, we can say that Python is a portable language.
6) Free and Open source: Python language is freely available at official web address. The source-code is also available. Therefore it is open source.
7) Object-Oriented language: Python supports object oriented language and concepts of classes and objects come into existence.
8) Large Standard Library: Python has a large and broad library and provides rich set of modules and functions for rapid application development.
9) GUI Programming Support: Graphical user interfaces can be developed using python.
10) Python also features dynamic type system and automatic memory management.
11) Extensible: It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our python code.

2. Define

a. Keywords :

- Keywords are reserve words which convey a special meaning to interprter.
- keywords can’t be used as identifiers or variables or function name.
- Eg: if ,else,elif,while,True,False,import (Total 33 keywords)

b. Variable :

- Python variables are the reserved memory locations used to store values with in a Python Program.
- Variable names should have combination of lowercase(a to z) & uppercase(A to Z) letters or digits(0 to 9)or underscore(_).

c. Comments

- Comments are statements that are ignored by interpretor in python program.

Comments

Single-line comment | multi-line comment

Denotion : # | Denotion : “”” “””

For single line comment | used to provide detail explaination about code,block.

d. Indentation

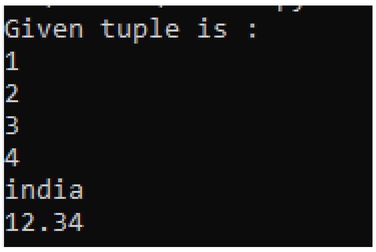
- Indentation refers to spaces at the beginning of a code line.
- Python indentation is used to declare block of code.
- Incorrect indentation will create indentation error.

3. Explain any two data type in detail

a. Tuple

- Tuple is a collection of objects which are immutable and ordered.
- Tuple is inbuilt data structure in python
- A tuple consist of number of values separated by commas & enclosed in ()
- Some inbuilt function used with tuple are :
  - Len() : return no. of elements in a tuple
  - max() : return maximum element in a tuple
  - min() : return minimum element in a tuple
  - sum() : return sum of all elements of a tuple
  - index(x) : return index of element x
  - count(x) : return number of occurences of element x
- Syntax : my_tuple=(item1,item2…..itemn)
- <u>example :</u>

```
tup=(1,2,3,4,'india',12.34)
print("Given tuple is : ")
for i in range (len(tup)):
        print(tup[i]
```

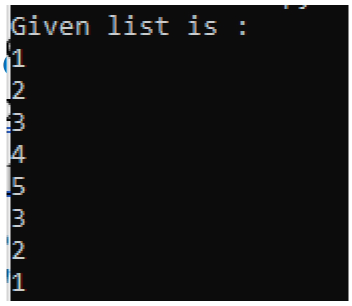


```
Given tuple is :
1
2
3
4
india
12.34
```

b. List

- A list is created by placing all the items (elements) inside square brackets [], separated by commas.
- List is mutable which mean We can update the list such as adding and removing elements during execution.
- A list is an object that contains multiple data items (elements).
- It can have any number of items and they may be of different types (integer, float, string etc.)
- Duplicate elements are also allowed into list.
- Syntax : my_list=[item1,item2…..itemn]
- Example :

```
l1=[1,2,3,4,5,3,2,1]
print("Given list is : ")
for i in range (len(l1)):
        print(l1[i])
```



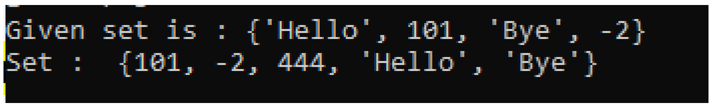
```
Given list is :
1
2
3
4
5
3
2
1
```

c. Set

- The set in python can be defined as an unordered collection of various items enclosed within the curly braces({ }).
- Set is unordered collection collection of elements without duplicates
- Sets are immutable
- The elements of the set cannot be duplicated and cannot be changed.
- There is no index allocated to the elements of the set, 1.e., we cannot directly access any element of the set by the index.
- We can print all the elements by looping through the set.

- Syntax : my_set={item1,item2…..itemn}
- Example :

```
s1 = {'Hello', 101, -2, 'Bye'}
print("Given set is :",s1)
s1.add(444)
print("Set : ",s1)
```

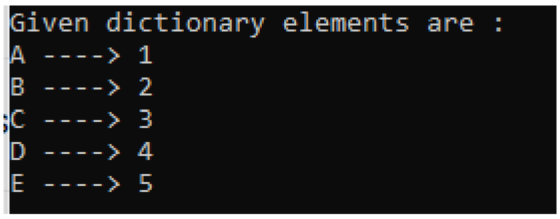


```
Given set is : {'Hello', 101, 'Bye', -2}
Set :  {101, -2, 444, 'Hello', 'Bye'}
```

d. Dictionary

- A dictionary in Python is defined using key-value pairs, separated by commas (,) enclosed within curly braces({ }). The key and value are separated using a colon (:).
- A for loop is used to transverse all keys & values of dictionary.
- The keys defined for a dictionary need to be unique, for keys.
- The values in a dictionary can be mutable
- Dictionary items are unordered, changeable, and does not allow duplicates.
- Syntax:

```
mydict = {
"key1 ": "value1 ",
"key2 ": "value2 ",
……………..}
```



```
Given dictionary elements are :
A ----> 1
B ----> 2
C ----> 3
D ----> 4
E ----> 5
```

- Example :

```
dict={'A':1,'B':2,'C':3,'D':4,'E':5}
print("Given dictionary elements are : ")
for key in dict:
        print(key,"---->",dict[key])
```

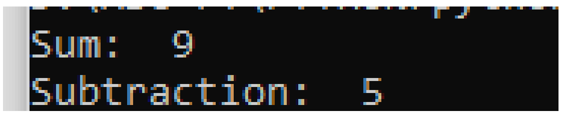
4. Explain operators in python

1. Python Arithmetic Operators

- Arithmetic operators are used to perform mathematical operations like addition, subtraction, multiplication, etc.
- + , - ,* , / , // ,** ,%

Example :

```
a = 7
b = 2
print ('Sum: ', a + b)
print ('Subtraction: ', a - b)
```

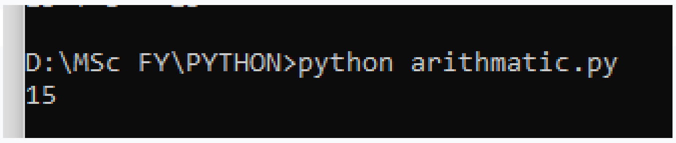


```
Sum:   9
Subtraction:   5
```

2. Python Assignment Operators

- Assignment operators are used to assign values to variables.
- = , += ,-= , *= ,/+ , %= , //= ,**= ,&= ,>>= , <<=
- For eg :

```
a = 10
b = 5
a += b      # a = a + b
print(a)
```



```
D:\MSc FY\PYTHON>python arithmatic.py
15
```

3. Python Comparison Operators

- Comparison operators compare two values/variables and return a boolean result: True or False.
- == ,!= ,> , < .>= ,<=
- For eg :

```
a=23

b=34
```

```
print(a>b)

print(a<b)
```

```
o/p : False

    True
```

4.Relational operator

>,<,<=,>=,==,!=

5. Python Logical Operators

Logical operators are used to combine conditional statements

AND , OR , NOT

Example :

N1=2

N2=5

If(n1)

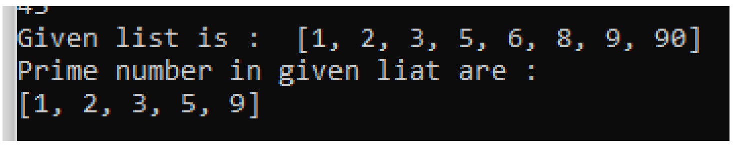
5. Explain any 2 function with example

- A function is a block of code which only runs when it is called.

a. Filter()

- The filter() function is utilized to apply a function to each element of an iterable (like a list or tuple) and return another iterable containing just the elements for which the function brings True back.
- The filter() function returns an iterator where the items are filtered through a function to test if the item is accepted or not.
- **Syntax :** filter(*function*, *iterable*)

```
import functools
def prime(n):
 for i in range (2,n//2+1):
         if(n%2==0):
                 return(False)
 return(True)
li=[1,2,3,5,6,8,9,90]
value=filter(prime,li)
print("Given list is : ",li)
print("Prime number in given list are : ",list(value))
```



```
Given list is :  [1, 2, 3, 5, 6, 8, 9, 90]
Prime number in given liat are :
[1, 2, 3, 5, 9]
```

o/p : [1,2,3,5,9]

b. Map()

- The map function in Python is a built-in function that applies a given function to each element of an iterable (list, tuple etc.) and returns an iterator containing the results.
- The map() function executes a specified function for each item in an iterable. The item is sent to the function as a parameter.
- **Syntax :** map(*function*, *iterables*)

```
def fact(n):
        f=1
        for i in range (1,n+1):
                f=f*i
        return f
li=[1,2,3,4,5]
res=map(fact,li)
print(list(res))
o/p : [1,2,6,24,120]
```

c. Reduce()

- Reduce fn is used to apply a fn to all the element of a sequence.
- We have to import package functools for this function.
- Syntax : functools.reduce(function,sequence)
- Example :

```
from functools import reduce
def fun(a,b):
    return(a+b)
li=[1,2,3,4,5,6,7,8,9]
res=reduce(fun,li)
print(res)
o/p : 45
```

6. Explain any two array operation with example

a. Transversel

- Traversal Traversing an array means visiting each element of the array in a specific order. The most common way to traverse an array is using a for loop
- For example, the following code traverses an array and prints each element to the console:

```
my_array = [1, 2, 3, 4, 5]
 for element in my_array:
      print(element)
```

o/p : 1,2,…5

b. Insertion :- NOTEBOOK

- Inserting an element into an array means adding the element to the array at a specific position.
- To insert an element into an array, you can use the insert() method.
- The insert() method takes two arguments: the index at which to insert the element and the element to insert

c. Deletion :- NOTEBOOK

d. Search

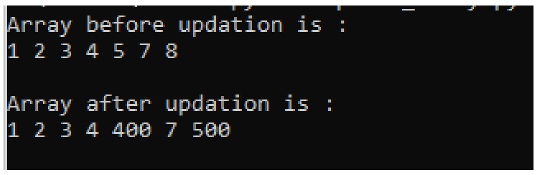
```
import array
arr=array.array("i",[1,2,3,4,5,6])
for i in arr:
  print(i,end=" ")
print("\n")
print("which number you want to search:")
n=int(input())
```

```
f=0
for i in range(0,len(arr)):
  if(arr[i]==n):
    print("Number is found at position",arr.index(arr[i]))
    f=1
    break
if(f==0):
    print("Number is not found")
```

e. Update

You can update elements in an array by accessing the element using its index and assigning it a new value.

```
import array as arr
a=arr.array("i",[1,2,3,4,5,7,8])
print("Array before updation is :")
for i in a:
       print(i,end=" ")
print("\n")

#update array
a[4]=400
a[6]=500
print("Array after updation is :")
for i in a:
       print(i,end=" ")
print("\n")
```



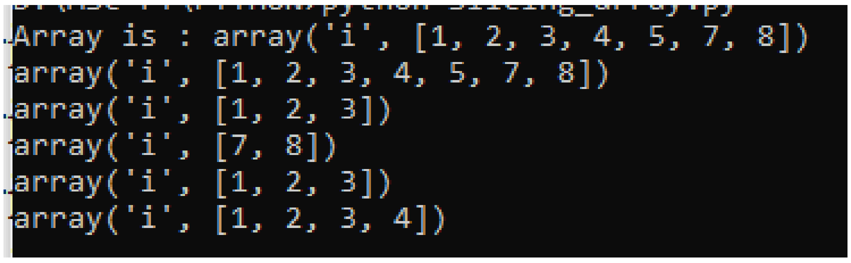
```
Array before updation is :
1 2 3 4 5 7 8

Array after updation is :
1 2 3 4 400 7 500
```

7. Difference betn array and list :- :- NOTEBOOK

8. Explain slicing of array with example

- array **slicing** is a way to get a subarray of the given array.
- Multiple values stored within an array can be accessed simultaneously with array *slicing*. To pull out a section or slice of an array, the colon operator : is used when calling the index. The general form is:
- **Syntax: subarray =** <array>[start:stop]

```
import array as arr
a=arr.array("i",[1,2,3,4,5,7,8])

print("Array is :",a)

#slicing

print(a[ : ])

print(a[ :3])

print(a[-2:])

print(a[0:3])

print(a[0:-3])
```



```
Array is : array('i', [1, 2, 3, 4, 5, 7, 8])
array('i', [1, 2, 3, 4, 5, 7, 8])
array('i', [1, 2, 3])
array('i', [7, 8])
array('i', [1, 2, 3])
array('i', [1, 2, 3, 4])
```

9. What is function ? explain

- A function is a block of code that performs a specific task. Functions can take arguments and return values
- To define a function in Python, you use the def keyword. The def keyword is followed by the function name and the function's arguments.
- The function body is then defined using indentation. For example, the following code defines a function called fact(), which takes a number as argument and returns the factorial of a number.

```
#Find factorial of number
def fact(n):
        f=1
        for i in range(1,n+1):
                f=f*i
        return f
n=int(input("Enter a number :"))
print("Factorial of ",n,"is : ",fact(n))
```

10. WAPP to implement stack using list

```
stk=[]
def push():
        ele=int(input("ENter a no :"))
        stk.append(ele)
        print("element is inserted")
def pop_ele():
        if not stk:
                print("Stack is empty")
        else:
                e=stk.pop()
                print("element is deleted")
while(True):
        print("1.Push\n2.pop\n3.exit")
        print("Enter your choice")
        n=int(input())
        if(n==1):
                push()
        elif(n==2):
                pop_ele()
        elif(n==3):
                exit()
        else:
                print("Enter right choice")
```

11. WAPP to implement queue using list

```
que=[]
def enque():
        ele=int(input("ENter a no :"))
        que.append(ele)
        print("element is inserted")
def deque_ele():
        if not que:
                print("Queue is empty")
        else:
                e=que.pop()
                print("element is deleted")
while(True):
        print("1.Enque(add element)\n2.deque(to delete
element)\n3.exit")
        print("Enter your choice")
        n=int(input())
        if(n==1):
                enque()
        elif(n==2):
                deque_ele()
        elif(n==3):
                exit()
        else:
                print("Enter right choice")
```

12. WAPP to print given number is prime or not using function

```
#prime no
def prime(n):
    f=0
    for i in range(2,n):
        if(n%i==0):
            f=1
        break
    if(f==0):
        print(n," is prime number")
    else:
        print(n," is not prime number")
n=int(input("Enter a number :"))
prime(n)
```

13. WAPP to find factorial of given number using function

```
def fact(n):
                f=1
                for i in range(1,n+1):
                        f=f*i
                return f
n=int(input("Enter a number :"))
print("Factorial of ",n,"is : ",fact(n))
```

14. WAPP for array operations.

```
#array operation
import array as arr
#creating a array
a=arr.array("i",[1,2,3,4,5,6])
b=arr.array("i",[11,22,33,44,55,66])
print("Given Array is : ")
print("[",end=" ")
for i in a:
        print(i,end=" ")
print("]\n")
#appending elements
a.append(56)
print(a)
#insert group of elements
a.extend([100,23])
print(a)
#inserting elements by position
a.insert(3,500)
print(a)
#remove last element
ele=a.pop()
```

```
print(a)
#slicing of array
print(a[0:4])
#updating element
a[3]=30
print(a)
```