# Python

## Assignment - 2

Reg.No: 321810306052

① What the data types in python? Explain

Ans 1) **Numbers:** Number -data types store numeric values. Number objects are created when you assign a value to them.

2) **Strings:** Strings in python are identified as a contiguous set of characters represented in quotation marks. Python allows either pair of single or double quotes.

3) **Lists:** Lists are most versatile of python's compound data types. A list contains items separated by commas. and enclosed within square brackets([ ])

4) **Tuples:** A tuple is another sequence data type that is similar to list. A tuple consists of number of values seperated by commas, here tuples are enclosed within parenthesis.

5) **Dictionary:** Python's dictionaries are kind of hash-table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. Values on the other hand, can be any arbitrary python object. Dictionaries are enclosed with curly braces.

② Briefly explain history of python.

Ans: Python is conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde & Informatica(CWI) in Netherlands as a successor to the ABC language capable of exception handling and interfacing with Amoeba operating system.

Its implementation began in December 1989. Van Rossum shouldered sole responsibility for project ~~as~~ until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as python's Benevolent

Dictator For Life, a title the python community bestowed upon him to reflect his long-term commitment as project's chief decision-maker.

In January 2019, active python core developers elected Brett Cannon, Nick Coghlan, Barry Warsaw, Carol Willing and Van Rossum to five-member "steering Council" to lead project. python 2.0 ~~w~~ was released on 16 october 2000 with many major new features, including a cycle-detecting garbage collector.

Python 3.0 was released on 3 December 2008. Many of its major features were backported to python 2.6x and 2.7.x version series.

python 2.7's ends-of-life date was initially set ~~at~~ 2015 then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to python 3.

(3) Explain all the Operators in python.

Ans Arithmetic operators:- Arithmetic operators used to perform mathematical operations.

| Operator | Meaning | Example |
|---|---|---|
| + | Add two operands | x+y+2 |
| - | Subtract right operand from left | x-y-2 |
| * | Multiply two operands | x*y |
| / | Divide left operand by right one | x/y |
| % | Modulus- remainder of division | x%y |
| // | Floor division - division results into whole number adjusted to left in number line | ~~x~~ x//y |

| | | |
|---|---|---|
| ** | Exponent - left operand raised to the power of right | x**y<br>(x to power y) |

Comparison operators : Comparative operators are used to compared value.

| operator | Meaning | Example |
|---|---|---|
| > | Greater than - True if left operand is greater than right | x > y |
| < | Less than - True if left operand is less than right | x<y |
| == | Equal to - True if both operands are equal | x==y |
| != | Not equal to - True if operands are not equal | x!=y |
| >= | Greater than or equal to - True if left operand is greater than or equal to right | x>=y |
| <= | Less than or equal to - True if left operand is less than or equal to the right | x<=y |

Logical operators :-

| operator | Meaning | Example |
|---|---|---|
| and | True if both the operands are true | x and y |
| or | True if either of the operands is true | x or y |
| not | True if operand if false | not x |

Bitwise operators:- These act on operands as if they were strings of binary digits.

| Operators | Meaning | Example |
|---|---|---|
| & | Bitwise AND | $x \& y = 0$ (0000 0000) |
| \| | Bitwise OR | $x \| y = 14$ (0000 1110) |
| ~ | Bitwise NOT | $\sim x = -11$ (1111 0101) |
| ^ | Bitwise XOR | $x \wedge y = 14$ (0000 1110) |
| >> | Bitwise right shift | $x >> 2 = 2$ (0000 0010) |
| << | Bitwise left shift | $x << 2 = 40$ (0010 1000) |

Assignment operators:- These are used in python to assign values to variables

| Operator | Example | Equivalent to |
|---|---|---|
| = | x=5 | x=5 |
| += | x+=5 | x=x+5 |
| -= | x-=5 | x=x-5 |
| *= | x*=5 | x=x*5 |
| /= | x/=5 | x=x/5 |
| %= | x%=5 | x=x%5 |
| //= | x//=5 | x=x//5 |
| **= | x**=5 | x=x**5 |
| &= | x&=5 | x=x&5 |
| \|= | x\|=5 | x=x\|5 |
| ^= | x^=5 | x=x^5 |

| | | |
|---|---|---|
| >>= | x >>=5 | x = x >>5 |
| <<= | x<<=5 | x=x<<5 |

(6)

Identity operators.

| Operator | Meaning |
|---|---|
| is | True if the operands are identical |
| is not | True if the operands are not identical |

'is' and 'is not' are identity operators in python. They used to check two values are located on some part of memory

Membership operators

'in' and 'not in' are the membership operators. They are used to test whether a value or variable is found in a sequence.

| Operator | Meaning | Example |
|---|---|---|
| in | True if value/variable is found in sequence | 5 in x |
| not in | True if value/variable is not found in sequence | 5 not in x |

(4) Explain the features of python.

Easy to code: python is a very developer-friendly language which means that anyone and everyone can learn to code it in a couple of hours or days. Languages like Java, C, C++ and C#, python is one of easiest to learn.

Open Source and free :- python is an open-source programming language which means that anyone can create and contribute to its development. Python is free to download and use in any operating system, be it Windows, Mac or Linux.

Support for GUI :-

GUI or Graphical user Interface is one of the key aspects of language because it has ability to add flair to code and make the results more visual. It support a wide array of GUIs which can easily be imported to interpreter

Object -oriented Approach :-

This basically means that python recognizes the concept of class and object encapsulation thus allowing programs to be efficient in long run.

High level language :

When you code in python you don't need to be aware of coding structure, architecture as well as memory management.

Integrated by Nature :-

Python is an integrated language by nature. Python Interpreter executes codes one line at a time. Python code thus making debugging process much easier and efficient. Another is, that upon execution the python code is immediately converted into an intermediate form known as byte-code.

Highly portable :-

Suppose you are running python on windows and you need to shift the same to either a Mac or Linux

14

then you can easily achieve the same in python without having to worry about changing the code. Thus making python one of the most portable languages available in industry.

Highly Dynamic :-

The type of a variable is decided at the run time and not in advance. Due to presence of this feature, we do not need to specify type of variable during coding, thus ~~solving~~ saving time and increasing efficiency.

Extensive Array of Library :-

out of the box, Python comes inbuilt with a large number of libraries than can be imported at any instance and used in a specific program. you don't need to write all code yourself and can import the same from those that already exist in libraries

5) Justify why python is interactive interpreted language.

Ans Unlike C/C++ etc., python is an interpreted object-oriented programming language. By interpreted it is meant that each time a program is run the interpreter checks through code for errors and then interprets the instructions into machine-readable ~~bytecode~~. bytecode

An interpreter is a translator in computer's language which translates the given code line-by line in machine readable bytecodes. And if any error is encountered it stops translation until the error is fixed. Unlike 'C' language, which is a compiled programming language. The compiler translates the whole code in one-go rather than line-by-line This is the reason why in C language, all the errors are listed during compilation only.

Ex:-

```
print "\n\n---This line is Correct---\n\n" #line1
print Hello     # this is wrong            #line2
```

```
:~/Desktop/test$ python g.py
------This line is Correct----\n
Traceback(most recent call last):
   File "g.py", line 5, in
        print Hello # this is wrong      #line2
Name Error: name 'Hello' is not defined
```