Assignment - 2

1. what the data types in Python? Explain

A) Data types are classification or categorization of data types items. Data types represent a kind of value which determines what operations can be performed on that data.

Numeric, non-numeric & Boolean (true/false)

python has the following standard or built-in datatypes

Numeric

A numeric value is any representation of data which has a numeric ~~d~~ value. python identifies three types of numbers:

Integer

Positive or Negative whole numbers

Ex: signed integers like 10, 30, 526 etc.

Float

Any real numbers with a floating point representation in which a fractional component is denoted by a decimal symbol

ex: 1.9, 9.902, 15.2 etc

Complex number

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 is called an imaginary numbers)

ex: 2.14j, 2.0+2.3j etc.

Boolean data type

Data with one of two built in values True or False

Notic that 'T' and 'F' are capital. true and false are not valid booleans and python will throw an error for them.

Ex:
```
a = 2
b = 4
c = 2<4
```
o/p: True

Sequence Type

A sequence is an orderd 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.

Ex:
```
str1 = 'hello javatpoint'
str2 = ' how are you'
print( str1+str2)
```
o/p: hello javatpoint how are you.

list

A list object is an ordered collection of one or more data ~~type~~ items, not necessarily of the same type, put in square brackets. we can use slice[:] operator to access and concatinate (+) and repetition (*)

ex:
```
l = [1, "hi", "python", 2]
print(l[3:])
print (l[0:2])
print (l);
```

```
print(l+l)
print(l*3)
o/p: [2]
     [1, 'hi']
     [1, 'hi', 'python', 2]
     [1, 'hi', 'python', 2, 1, 'hi', 'python', 2]
     [1, 'hi', 'python', 2, 1, 'hi', 'python', 2, 1, 'hi', 'python', 2]
```

Tuple

A Tuple object is an Ordered Collection of one or more data items, not necessary of the same type, put in paranthesis.

Ex:
```
t = ("hi", "python", 2)
print(t[1:]);
print(t[0:1]);
print(t);
print(t+t);
print(t*3);
print(type(t))
t[2] = "hi"
```

```
O/P: ('python', 2)
     ('hi',)
     ('hi', 'python', 2)
     ('hi', 'python', 2, 'hi', 'python', 2)
     ('hi', 'python', 2, 'hi', 'python', 2, 'hi', 'python', 2)
     <type 'tuple'>
     Traceback(most recent call last):
         File "main.py", line 8, in <module>
             t[2] = "hi"
     TypeError: 'tuple' Object ~~Class~~ does not support item
```

Scanned with CamScanner

assignment.

Dictionary

Dictionary is an ordered Set of a key-value pair of items. It is like an associative array or a hash table where each key stores a specific value. key can hold any primitive data type whereas value is an arbitrary Python object.

* The items in dictionary are seperated with the comma and enclosed in the curly braces {}.

Ex

```
d= {1:'jimmy', 2:'Alex', 3:'john', 4:'mike'};
Print ("1st name is" +d[1]);
Print ("2nd name is" +d[4]);
Print (d);
Print (d.keys());
Print (d.values());
```

O/P:

```
1st name is Jimmy
2nd name is mike
{1: 'Jimmy', 2:'Alex', 3:'John', 4 :'mike'}
[1,2,3,4]
['Jimmy', 'Alex', 'John', 'mike']
```

2). Briefly explain history of python.

* Python laid its foundation in the late 1980s.

* The implementation of python was started in the December 1989 by Guido van Rossum at CWI in netherland.

* In febrauary 1991, van Rossum published the code (labeled version 0.9.0) to alt.sources.

* In 1994, Python 1.0 was released with new features like : lambda, map, filter, & reduce.

* python 2.0 added new features like : list comprehensions, garbage Collection. system.

* On December 3, 2008, Python 3.0 (also called "py3k") was released. It was designed to rectify fundamental flaw of the language.

* ABC programming language is said to be the predecess or of python language which was Capable of Exception Handling and interfacing with Amoeba OS.

* python influenced by following programming languages.
  * ABC language
  * modula-3

Version-List

Version	Released date
Python 1.0	January 1994
Python 1.5	Dec 31, 1997
Python 1.6	Sep 5, 2000
Python 2.0	Oct 16, 2000
Python 2.1	Apr 17, 2001
Python 2.2	Dec 21, 2001
Python 2.3	Jul 29, 2003
Python 2.4	Nov 30, 2004
Python 2.5	Sep 19, 2006

Python 2.6	oct 1, 2008
Python 2.7	July 3, 2010
Python 3.0	Dec 3, 2008
python 3.1	June 27, 2009
Python 3.2	feb 20, 2011
Python 3.3	Sep 29, 2012
Python 3.4	Mar 16, 2014
Python 3.5	Sep 13, 2015
Py thon 3.6	Dec 23, 2016
Python 3.7	June 27, 2018

3. Explain all the operators in python.

The operator can defined as a symbol which is responsible for a particular operation b/w two operands. operators are the pillars of program on which the logic is built in a particular programming language. Python provides a variety of operators

→ Arithematic operators.

→ Comparison operators

→ Assignment operators

→ Logical operators

→ Bitwise operators

→ Membership operators.

→ Identity operator

Arithematic Operators

Arithematic operators are Used to perform arithematic operations between two Operands. It includes +, -, *, /, %, //, **

+ (addition)

It is used to add two operands.

For example, if a = 20, b = 10 => a+b = 30

- (subtraction)

It is used to subtract the second operand from the first operand. If the first operand is less than the Second operand, the value result negative. for ex, if a = 20, b = 10 => a-b = 10

/ (divide)

It returns the quotient after diving the first operand by the Second operand. For ex, if a = 20, b = 10 => a/b = 2

* (multiplication)

It is used to multiply one operand with the other.

for ex, If a = 20, b = 10 => a*b = 200.

% (remainder)

It returns the remainder after dividing the first operand by the Second operand. for ex, if a = 20, b = 10 => a%b = 0

** (Exponent)

It is an exponent operator represented as it calculates the first operand power to second operand.

// (floor divison).

It gives the floor value of the quotient produced by dividing the two operands.

Comparison operator

comparison operators are used to Comparing the value of the two operands & returns boolean true or false accordingly. The Comparison operators are described in the following table

Operator	Description
==	If the value of two operands is equal then the condition becomes true.
!=	If the value of two operands is not equal then the condition becomes true.
<=	If the first operand is less than or equal to the second operand, then the condition becomes true.
>=	If the first operand is greater than or equal to the second operand, then the condition becomes true.
>	If the first operand is greater than the second operand, then the condition becomes true.
<	If the first operand is less than the second operand, then the condition becomes true.

~~Bo~~ Assignment Operators

→ Assignment operators

The assignment operators are used to assign the value of the right expression to the left operand. The assignment operators are described as follows

Operator	Description
=	It assigns the value of the right expression to the left operan
+=	It increases the value of the left operand by the value of the right operand and assign the modified value back to left operand. for ex, if $a=10, b=20 \Rightarrow a+=b$ will be equal to $a=a+b$ and $\therefore, a=30$
-=	It decreases the value of the left operand by the value of the right operand and assign the modified value back to left operand. for ex, if $a=20, b=10 \Rightarrow a-=b$ will be equal to $a=a-b$ and $\therefore, a=10$
*=	It multiplies the value of the left operand by the value of the right operand and assign the modified value back to left operand. for ex, if $a=10, b=20 \Rightarrow a*=b$ will be equal to $a=a*b$ and therefore, $a=200$.
%=	It divides the value of the left operand by the value of the right operand and assign the remainder back to left operand. for ex, if $a=20, b=10 \Rightarrow a\%=b$ ~~[illegible]~~ will be equal to $a=a\%b$ and $\therefore, a=0$.

= a=b will be equal to a=a**b, for ex, if a=4, b=2, a**=b will assign 4**2=16 to a.

//= A//=b will be equal to a=a//b, for ex, if a=4, b=3, a//=b will assign 4//3=1 to a.

Bitwise operator

The bitwise operators performs bit by bit operation on the values of two operands.

Ex if a=7;
b=6;

Operator	Description
& (binary and)	If both the bits at the same place in two operands are 1, then 1 is copied to the result. Otherwise, 0 is copied.
| (binary or)	The resulting bit will be 0 if both the bits are zero otherwise the resulting bit will be 1.
^ (binary xor)	The resulting bit will be 1 if both the bits are different otherwise the resulting bit will be 0.
~ (negation)	It calculates the negation of each bit of the operand, i.e., if the bit is 0, the resulting bit will be 1 and vice versa

<< (left shift)	The left operand value is moved left by the number of bits present in the right operand
>> (right shift)	The left operand is moved right by the number of bits present in the right operand.

Logical operators

The Logical operators are used primarily in the expression evaluation to make a decision.

operator	Description
and	If both the expression are true, then cond will be true. If a & b are two expressions, a → true, b → true => a and b ⇒ true.
or	If one of the exp is true, then the con will be true. if a & b are two exp, a → true, b → false ⇒ a or b → true
not	If an exp a is true then not(a) will be false & vice versa

Membership Operators

Python membership operators are used to check the membership of value inside a python data structure. If the value is present in the data structure, then the resulting value is true otherwise it returns false.

operator	Description
in	It is evaluated to be true, if the first operand is found in the second operand (List, tuple dictionary)
not in	It is evaluated to be true if the first operand is not found in the second operand (List, tuples dictionary)

Identity operators

Operator	Description
is	It is evaluated to be true if the reference present at both sides point to the same Object
is not	It is evaluated to be true if the reference present at both side do not point to The same Objects

4. Explain The features of Python?

1. Easy to learn & Use.

python is easy to learn and use. It is developer-friendly and high level programming language.

2. Expressive language

python language is more expressive means that it is more Understandable & readable.

3. Interpreted language

python is an interpreted language i.e. interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.

4. cross-platform language

python can run equally on different Platforms such as Windows, Linux, Unix & Macintosh etc. So, we can say that python is a portable language.

5- Free & open source

Python language is freely available at official web address. The Source Code is also available. Therefore It is open source.

6. Object-oriented Language

Python Supports object oriented language & concepts of classes & objects come into exsistence.

7. extensible

It implies that other languages such as C/c++ can be used to compile the code & thus it can be used furthur in our python code.

8. Large standard library

Python has a large and broad library & provides rich set of module & functions for rapid application development.

9. GUI programming Support

GUI can be devoloped by using python

10. Integrated

It can be easily integrated with languages like C, C++, JAVA etc.

5). Justify why Python is interactive interpreted language

A). Unlike C/c++ etc, Python is an interpreted object-oriented programming language. unlike C language, which is Compiled programming language. The compiler translates The whole code in one-go rather than line-by-line. This the reason

Python is interactive. when python statement is entered and is followed by the return key, if appropriate, the result will be printed on the Screen, immediately in the nextline. This is particularly advantageous in the debugging process. In interactive mode of operation, Python is used in a Similar way as the Unix Command line or the terminal.

→ Interactive python is very much helpful for the debugging purpose. It simply returns the >>> prompt to the Corresponding output of the Statement if appropriate & returns error for incorrect statements. In this way if you have any doubts like: whether a syntax is correct, whether the module you are importing exists or anything like that, you can be Sure within Seconds using python interactive mode.