Assignment-2

1. What are the data types in Python? Explain.

Python Data Types:

Data types are the classification or categorization of data items. Data types represent a kind of value which determines what operations can be performed on that data. Numeric, non-numeric and Boolean (true/false) data are the most used data types. However, each programming language has its own classification largely reflecting its programming philosophy.

Python has the following standard or built-in data types:

Numeric

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

→ Integer: Positive or negative whole numbers

→ Float: Any real number with a floating point representation in which a fraction component is denoted

→ Complex number: A number with real and imaginary components represented as x+yj

Boolean

Data with one of two built-in values True or False. T and F are capital.
true and false are not valid booleans

Sequence Type.

Sequences Python has sequence data types.

-> String: A string value is a collection of one or more characters put in single, double or triple quotes.

-> List: A list object is an ordered collection of one or more ~~charact~~ data items, not necessarily of the same type, put in square brackets.

-> Tuple: A Tuple object is an ordered collection of one or more data items, not necessarily of the same type, put in parentheses.

2. Briefly explain history of Python.

Python was conceived in the late 1980s by Guido Van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to ABC language, capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989. Van Rossum shouldered sole responsibility for the project, as the lead developer, 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 the project's chief decision-maker.

3 Explain all the operators in Python.

Python operators:

1. Arithmetic Operators:

Arithmetic operators are used to perform

mathematical operations like addition, subtraction, multiplication & division.

Operator	Description	Syntax
+	Addition: adds two operands	x+y.
-	Subtraction: Subtracts two operands	x-y
*	Multiplication: Multiplies two operands	x*y
/	Division (float):	x/y.
//	Division (floor):	x//y
%	first operand is divided by the second	x%y
**	Power: Returns first raised to power second	x**y

2. Relational Operators: Relational operators compares the values. It either returns true or false.

Operator	Description	Syntax
>	Greater than	x>y
<	Less than	x<y
==	equal to	x==y
!=	Not equal	x!=y.
>=	Greater than or equal to	x>=y
<=	Less than or equal to	x<=y

3. Logical operators: Logical operators performs logical AND, Logical OR and logical NOT operations

Operator	Description	Syntax
and	Logical AND: True if both operands are true	x and y
or	Logical OR: True if either of the operands are try	x or y
not	Logical NOT: True if operand is false	not x

4. Bitwise operators: These operators acts on bits and performs bit by bit operation.

Operator	Description	Syntax
&	Bitwise AND.	x & y
|	Bitwise OR	x | y
~	Bitwise NOT	~x
^	Bitwise XOR	x ^ y
>>	Bitwise right shift	x >>
<<	Bitwise left shift	x

5. Assignment operators: Assignment operators are used to assign values to the variables

Operators	Description	Syntax
=	Assign value of right side of expression to left side operand	x = y + z
+=	Add AND:	a += b a = a + b

-=	Subtract AND	a-=b a=a-b
*=	Multiply AND:	a*=b a=a*b.
/=	Divide AND:	a/=b a=a/b
%=	Modulus AND:	a%=b a=a%b
//=	Divide(floor) AND:	a//=b a=a//b
**=	Exponent AND:	a**=b a=a**b
&=	Performs Bitwise AND on operands	a&=b a=a&b
|=	Performs Bit wise OR.	a|=b a=a|b.
^=	Performs Bitwise xOR.	a^=b a=a^b
>>=	Perform Bitwise right shift on operands	a>>=b
	Perform Bitwise Left shift on operands	a

6. Special operators:-

* Identity operators-
is and is not are the identity operators both are used to check if two values are located on the same part of the memory.

* Membership operators-
in and not in are the membership operators;

used to test whether a value
a sequence

4. Explain the features of python.

Python provides lots of features that are listed

1) Easy to learn and use.

Python is easy to learn and use. It is develope
friendly and high level programming languag

2) Expressive language

Python is more understable and readab

3) Interpreted language

It means interpreter executes the code line b
line at a time. This makes debugging easy an
thus suitable for beginners.

4) Cross-platform language:

Python can run equally on different platform
such as Windows, Linux, Unix and Macintosh

5) Free and Open Source:

Python is freely available at official web
address. The source-code is also available

6) Object-Oriented language

Python supports object oriented language and
concepts of classes and objects come into existenc

7) 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.

8) Large Standard Library

Python has a large and broad library and

provides rich set of module and functions for rapid application development.

9. GUI Programming Support

Graphical user interfaces can be developed using Python.

10. Integrated

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

5. Justify why phython is interactive interpreted Language.

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 the code for errors and then interprets the instructions into machine-readable bytecode.

Python is interactive. Why a Python statement is entered, and is followed by the Return key, if appropriate, the result will be printed on the screen, immediately, in the next line. 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.