UNIT-1

1. Python:

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readible.

2. Python applications:

1. web applications, Desktop GUI applications, Software development, Scientific and Numeric, Enterprise applications, Applications for Images, Audio or Video based applications, 3D CAD applications.

3. Assignment statement:

Python variables do not need explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable. The equal sign (=) is used to assign values to variables.

4. Keywords:

and, as, elif, else, if, print, raise, import, assert, break, class, continue, def, del, except, exec, finally, for, from, global, in, is, lambda, not, or, pass, return, try, while, with, yield.

5. Identifiers:

An identifier is a name given to an entity. It is defined as a user-defined name to represent the basic building blocks of python. It can be a variable, class, module, or any object.

6. Comment statement :

Comments in python begin with a hash mark (#) and whitespace character and continue to the end of the line. Comments are in the source code for humans to read, not for computers to execute.

7. Variables :

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

8. Features :

Easy to learn, Easy to read, Easy to maintain, Portable, Extendable, Database, Scalable, GUI programming.

9. Python Interpreter :

Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to ~~R~~ PERL and PHP.

10. Rules for writing an identifier.

* An identifier cannot start with a digit.
* Keywords cannot be used as identifiers.
* we connot use special symbols like !, @, #, $, % etc.

11. Rules for naming a variable.

* A variable name must start with a letter or underscore character.
* A variable name cannot start with a number.
* A variable name can only contain alpha-numeric characters and underscores (A-z , 0-9 and _).

UNIT - II

1. Datatypes in python:

Numbers , string , Boolean , List , tuple , set , Dictionary.

2. Operators in python:

Arithmetic operators
Comparison "
Assignment "
Logical "
Bitwise "
Membership "
Identity "

3. While loop:

A while loop statement in python programming language repeatedly executes a target statement as long as a given condition is true.

Syntax : While expression :
Statement (s)

4. Conditional If :

The if statement contains a logical expression using which data is compared and a decision is made based on the result of the comparison .

Syntax : if condition :

```
if condition :
            Statements
```

5. String with example :

Strings in python are identified as a continuous set of characters represented in the quotation marks . Python allows for either pairs of single or double quotes .

Ex :

```
str= "Welcome"
print str
print str[0]
print str [2:5]
```

Output :

```
Welcome.
W
LCO.
```

6. Swapping Of two Numbers :

```
x=5
y=10
temp = x
x=y
y= temp
print('The value of x after swapping:')
print ('The value of y after swapping:')
```

Th

Th

7. Python

Number

a valu

Ex

8. Arit

used

add

and

on

Con

Output:

```
The value of x after swapping : 10
The value of y after swapping : 5
```

7. Python Numbers:

Number data types ~~o~~ store numeric values. Number objects are created when you assign a value to them.

```
Ex :  a = 3
      b = 2.65
      c = 2+5j
      print ('int is', a)
      print ('float is', b)
      print ('complex is', c)
```

Output :

```
int is 3
float is 2.65
Complex is 2+5j
```

8. Arithmetic operator:

The arithmetic operators in python are used to perform math operations, such as addition, subtraction, multiplication, division and so on. You can apply these operators on python as well as variables to perform corresponding operations.

9. Syntax & usage of for loop:

The for loop is useful to iterate over the elements of a sequence. It means, the for loop can be used to execute a group of statements repeatedly upon the no. of elements in the sequency.

Syntax:

```
        for var in sequence:
            statements
```

10. Break:

It terminates the current loop and resumes execution at the next statements.

Syntax : break

```
Ex: for letter in 'Python':
        if letter == 'h':
            break
        print('Current letter:', letter)
```

Output:

```
        Current letter: P
        Current letter: y
        Current letter: t
```