2.multiple assignments:

(a)Assigning single value to the multiple variables:

Example:

```
>>> a=b=c=56
>>> print(a)
56
>>> print(b)
56
>>> print(c)
56
```

(b)Assigning multiple values to the multiple variables

Example:

```
>>> a,b,c=78,89,90
>>> print(a)
78
>>> print(b)
89
>>> print(c)
90
```

3.literals:

String literals:

String means group of characters

String in python represented with single quotes and double quotes.

1.single line string: strings that are terminated within a single line are known as single line strings.

2.multiline strings: A piece of text that is spread along multiple lines is known as multi-line strings.

2.numeric literals:

(a)int: numbers(can be both positive and negative) with no fractional part.

Example: 345,7890,-345

(b)floating point numbers:

Real numbers with both integer and fractional part.

Example:

78.345,56.133

(c) complex(): in the form of a+jb where a forms the real part and b forms the imaginary part of complex number

Example:

45+78j

(d) long integers:integers of unlimited size followed by upper case or lower case.

Example:

678632864L

(3)Boolean literals:

It have two values: True and False

(4)special literals:None is used to specify to that field that is not created. It is also used for end of lists in python.

Example:

```
>>> s=90
>>> d=None
>>> print(s)
90
>>> print(d)
None
```

Python comments:

Comments are nothing but giving brief description of a program.

1.single line comments(#)

2.multi-line comments(‘’’…….’’’)

(4)operators:

Operator is an symbol that performs an operation on one or more operands . An operand is a variable or a value on which we perform the operation.

Example:

A+B

Here,A,B are operands

‘+’ is operator

There are 7 types:

1.Arithmetic operators:

(a)Addition:

Add the values on either side of the operator.

Example:

>>> 23+45

68

(b)subtraction: subtracts the value on the right from the one on the left.

Example:

>>> 56-78

-22

>>> 45-78.234

-33.233999999999995

>>>

(c)multiplication: multiplies the values on either side of the operator

Example:

>>> 23*12

276

```
>>> 12*23.45

281.4

>>>
```

(d)division: divides the values on the left by the one on the right. Notice that division results in a floating point value.

Example:

```
>>> 45/12

3.75

>>> 67.89/12

5.6575

>>>
```

(e)modulo(%): divides and return the value of the remainder.

Example:

```
>>> 45%12

9

>>> 90.678%3

0.6779999999999973

>>>
```

(f)exponentiation(**): raises the first number to the power of the second.

Example:

```
>>> 3**4
```

81

>>> 12*4

48

>>> 12**4

20736

>>>

(f)floor division(//): divides and returns the integer value of the quotient. It dumps the digits after the decimal.

Example:

>>> 34//12

2

>>> 45//2

22

>>>

2.realtional operators:

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

Here we carries out the comparison between operands . they tell us whether an operand is greater than the other lesser,equal or combination of those values.

(a)less than(<): it checks if the value on the left of the operator is lesser than the one on right.

Example:

>>> 23<34

True

>>> 23<12

False

>>>

(b)greater than(>): it checks if the value on the left of the operator is greater than the one on the right.

Example:

>>> 23>45

False

>>> 45>12

True

>>> 23.456>67.89

False

>>> 89.123>89.078

True

(c) less than or equal to(<=):it checks if the value on the left of the operator is less than or equal to the one on the right.

Example:

>>> 34<=89

True

>>> 34<=34

True

>>> 34.567<=89.123

True

>>> 34.567<=34.567

True

>>>

(d) greater than or equal to operator(>=): it checks if the value on the left of the operator is greater than or equal to the one on the right.

Example:

>>> 34.89>=34.89

True

>>> 89>=12

True

(e) equal to operator(==): it checks if the value on the left of the operator is equal to the one on the right.

Example:

>>> 3==3.0

True

>>> 1==True

True

>>> 0==False

True

```
>>> 0.4==True

False

>>>
```

(f)not equal to operator(!=): it checks if the value on the left of the operator is not equal to the one on the right. The python operator <> does the same job, but <> it is removed in python 3.

Example:

```
>>> 1!=1.0

False

>>> 1<>2

SyntaxError: invalid syntax

>>>
```

3. Assignment operators:

It is assign a value to a variable.

Example:

```
>>> id=10

>>> name="sushma"

>>> print(id)

10

>>> print(name)

sushma
```

>>>

(a)Add and assign(+=):

Adds the values on either side and assigns it to the expression on the left.

Example:

```
>>> a=10
>>> a+=23
>>> print(a)
33
>>> a=90
>>> a+=123
>>> print(a)
213
>>>
```

(b)sub and assign(-=):

Subtract the value on the right from the value of the left

Example:

```
>>> a=-123
>>> print(a)
-123
```

```
>>> a-=123

>>> print(a)

-246
```

(c) divide and assign(/=)divides the value on the left by the one on the right . then it assign it to the expression on the left.

Example:

```
>>> a=123

>>> a/=23

>>> print(a)

5.3478260869565215

>>>
```

(d) multiply and assign(*=):

Multiplies the values on either sides. Then its assigns it to the expression on the left.

Example:

```
>>> a=12

>>> a*=4

>>> print(a)

48
```

(e)modulo and assign(%=):

Performs modulo on the values on either side then it assigns it to the expression on the left.

Example:

```
>>> a=12

>>> a%=4

>>> print(a)

0
```