Name: Akash Patil K M

Email id: akash7760726633@gmail.com

Mobile no: 7483744718

1) Check the 'is' operator on the float and string write your findings.

Ans:- ' is' operator – Evaluates to true if the variables on either side of the operator point to the same object and false otherwise.

```
# Python program to illustrate the use
of 'is' identity operator
x = 5
if (type(x) is int):
    print("true")
else:
    print("false")
```

```
>>> 0.0 is 0.0
True   # as expected
>>> float(0.0) is 0.0
True   # as expected
```

As expected till now, but here is my "problem":

>>> float(0) is 0.0

False

>>> float(0) is float(0)

False

The string comparison operator in python is used to compare two strings. “==” operator returns Boolean True if two strings are the same and return Boolean False if two strings are not the same. ” operator returns Boolean True if two strings are not the same and return Boolean False if two strings are the same.

2) 1>3>4 how to solve

Ans:-

x = 1

x >>= 3>4

print(x)

1

3) demonstrate int(),float(),str() function.

Ans:-

Integer:

In Python , there is effectively no limit to how long an integer value can be. Of course, it is constrained by the amount of memory your system

has, as are all things, but beyond that an integer can be as long as you need it to be:

For example:

```
>>> print(0o10)
8

>>> print(0x10)
16

>>> print(0b10)
2
```

Floating-Point Numbers:

The float type in Python designates a floating-point number. float values are specified with a decimal point. Optionally, the character e or E followed by a positive or negative integer may be appended to specify scientific notation:

```
>>> 4.2
4.2
>>> type(4.2)
<class 'float'>
>>> 4.
```

```
4.0
>>> .2
0.2


>>> .4e7
4000000.0
>>> type(.4e7)
<class 'float'>
>>> 4.2e-4
0.00042
```

Strings:

Strings are sequences of character data. The string type in Python is called str.

String literals may be delimited using either single or double quotes. All the characters between the opening delimiter and matching closing delimiter are part of the string:

```
>>> print("I am a string.")
I am a string.
>>> type("I am a string.")
<class 'str'>
```

```
>>> print('I am too.')
I am too.
>>> type('I am too.')
<class 'str'>
```