**1. In the below elements which of them are values or an expression? E.g. :- values can be integer or string and expressions will be mathematical operators.**

\*

'hello'

-87.8

-

/

+

6

Ans :  1) \* = operator  
2) 'hello'= value  
3) -87.8 = value  
4) - = operator  
5) / = operator  
6) + = operator  
7) 6 = value

Program : a = '+'

b = 'hello'

c = -87.8

d = '-'

e = '/'

f = '+'

g = 6

print(a)

print(b)

print(c)

print(d)

print(e)

print(f)

print(g)

Output :

\*

'hello'

-87.8

-

/

+

6

**2. What is the difference between string and variable?**

**Python String**

String is a datatype. A string is a data structure in Python that represents a sequence of characters. It is an immutable data type, meaning that once you have created a string, you cannot change it. Strings are used widely in many different applications, such as storing and manipulating text data, representing names, addresses, and other types of data that can be represented as text.

Example:

"Rutuja" or 'Rutuja'

Another Example : print ("A Computer Science")

Output: A Computer Science

**Python Variables**

Python Variable is containers that store values. Python is not “statically typed”. We do not need to declare variables before using them or declare their type. A variable is created the moment we first assign a value to it. A Python variable is a name given to a memory location. It is the basic unit of storage in a program. Variables are containers for storing data values. For example, x= “Rutuja”, Here x is a variable and “Rutuja” is a string.

Another Example of Variable in Python

An Example of a Variable in Python is a representational name that serves as a pointer to an object. Once an object is assigned to a variable, it can be referred to by that name. In layman’s terms, we can say that Variable in Python is containers that store values.

Here we have stored “Rutuja” in a var which is variable, and when we call its name. The stored information will get printed.

Var = "Rutuja"

print (Var)

Output: Rutuja

**3. Describe three different data types.**

1.Numeric Data Type in Python

Integers – This value is represented by int class. It contains positive or negative whole numbers (without fractions or decimals). In Python, there is no limit to how long an integer value can be.

Float – This value is represented by the float class. It is a real number with a floating-point representation. It is specified by a decimal point. Optionally, the character e or E followed by a positive or negative integer may be appended to specify scientific notation.

Complex Numbers – Complex number is represented by a complex class. It is specified as (real part) + (imaginary part) j. For example – 2+3j

# Python program to

# numeric value

a = 5

print ("Type of a: ", type(a))

b = 5.0

print ("\n Type of b: ", type(b))

c = 2 + 4j

print ("\n Type of c: ", type(c))

Output:

Type of a: <class 'int'>

Type of b: <class 'float'>

Type of c: <class 'complex'>

2.Sequence Data Type in Python

The sequence Data Type in Python is the ordered collection of similar or different data types. Sequences allow storing of multiple values in an organized and efficient fashion. There are several sequence types in Python –

* String
* List
* Tuple
* String

Strings in Python are arrays of bytes representing Unicode characters. A string is a collection of one or more characters put in a single quote, double-quote, or triple-quote. In python there is no character data type, a character is a string of length one. It is represented by str class.

# Creating a String

# with single Quotes

String1 = 'Welcome'

print ("String with the use of Single Quotes: ")

print (String1)

Output:

String with the use of Single Quotes:

Welcome

* List

Lists are just like arrays, declared in other languages which is an ordered collection of data. It is very flexible as the items in a list do not need to be of the same type.

# A list with 3 integers

numbers = [1, 2, 5]

print(numbers)

Output: [1, 2, 5]

* Tuple

A tuple is also an ordered collection of Python objects.

tuple = (0, [1, 2, 3], (4, 5, 6), 7.0)

print(tuple)

print(type(tuple))

Output:(0, [1, 2, 3], (4, 5, 6), 7.0)

3.Set Data Type in Python

In Python, a Set is an unordered collection of data types that is iterable, mutable and has no duplicate elements. The order of elements in a set is undefined though it may consist of various elements.

thisset = {"apple", "banana", "cherry"}

print(thisset)

{'apple', 'banana', 'cherry'}

**4. What is an expression made up of? What do all expressions do?**

Expression: An expression is a combination of operators, constants, values and variables. An expression may consist of one or more operands, and zero or more operators to produce a value.

Expressions need to be evaluated. All expressions evaluate to a single value.

Example:

a+b

c

s-1/7\*f

Types of Expressions:

* Expressions may be of the following types:

Constant expressions: Constant Expressions consists of only constant values. A constant value is one that doesn’t change.

Examples:

5, 10 + 5 / 6.0, 'x’

* Integral expressions: Integral Expressions are those which produce integer results after implementing all the automatic and explicit type conversions.

Examples:

x, x \* y, x + int (5.0)

where x and y are integer variables.

* Floating expressions: Float Expressions are which produce floating point results after implementing all the automatic and explicit type conversions.

Examples:

x + y, 10.75

where x and y are floating point variables.

* Relational expressions: Relational Expressions yield results of type bool which takes a value true or false. When arithmetic expressions are used on either side of a relational operator, they will be evaluated first and then the results compared. Relational expressions are also known as Boolean expressions.

Examples:

x <= y, x + y > 2

* Logical expressions: Logical Expressions combine two or more relational expressions and produces bool type results.

Examples:

x > y && x == 10, x == 10 || y == 5

* Pointer expressions: Pointer Expressions produce address values.

Examples:

&x, ptr, ptr++

where x is a variable and ptr is a pointer.

* Bitwise expressions: Bitwise Expressions are used to manipulate data at bit level. They are basically used for testing or shifting bits.

Examples:

x << 3

shifts three bit position to left

y >> 1

shifts one bit position to right.

Shift operators are often used for multiplication and division by powers of two.

**5. This assignment statements, like spam = 10. What is the difference between an expression and a statement?**

Ans : An expression evaluates to a single value. A statement does not.

A statement does something. Statements represent an action or command e.g. print statements, assignment statements. An expression is a combination of operators, variables, values. Expressions need to be evaluated. An expression is something that can be reduced to a value.

**6. After running the following code, what does the variable bacon contain?**

bacon = 22

bacon + 1

print(bacon)

#bacon will contain 22

Output : 22

**7. What should the values of the following two terms be?**

'spam' + 'spamspam'

'spam' \* 3

Ans. Both expressions evaluate to the string 'spamspamspam'

Output : spamspamspam

spamspamspam

**8. Why is eggs a valid variable name while 100 is invalid?**

Ans. Variable names cannot begin with a number, therefore 100 cannot be the variable name .

**9. What three functions can be used to get the integer, floating-point number, or string version of a value?**

Ans. The int(), float(), and str() functions will evaluate to the integer, floating-point number, and string versions of the value passed to them.

# Converting to integer

int\_value = int(10)

print(type(int\_value))

#Converting to float

float\_value = float(10)

print(type(float\_value ))

#Converting to String

string\_value = str(10)

print(type(string\_value))

Output : <class 'int'>

<class 'float'>

<class 'str'>

**10. Why does this expression cause an error? How can you fix it?**

'I have eaten ' + 99 + ' burritos.'

Ans. The expression causes an error because 99 is an integer, and only strings can be concatenated to other strings with the + operator. The correct way is I have eaten ' + str(99) + ' burritos.'.