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

* ==========➔ expression

'hello' ==========➔ value

-87.8 ==========➔ value

- ==========➔ expression

/ ==========➔ expression

+ ==========➔ expression

6 ==========➔ value

2. What is the difference between string and variable?

Sol: string: string is a datatype in python it could store single or multiple characters. The string in python is enclosed within the double quotes(“ ”) or single quotes(‘ ‘)

Ex: ‘iNeuron’ , “python”

Variable: variable could also be referred as identifier. The fundamental behaviour of the variable is that I could vary over time (in the code). It could also referred as a named container in a lay man terms so that our values can store in variable and we could use those named containers later.

Ex: num1 = 15

Num2 = 25

Sum = num1 + Num2

3. Describe three different data types.

Sol: int, bool, list

int: int is a numeric data type, it can represent negative, positive, zero values without fractional part.

Ex: -20, 0, 15

bool: bool data type having only two different type of values one is True and the other is False. Mostly it is not explicitly written but extensively appears when we perform some logical operations and control flow conditions.

Ex: True, False

list: list is an ordered collection of elements. It most commonly used store the elements of any kind (ex: int, float, bool, str, list, tuple etc)

Ex: l = [1, 3.2, True, 'string', [1,2], (3,4)]

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

Sol: An expression in Python is made up of one or more operands and operators, combined in a way that represents a computation or evaluation. An operand can be a value, a variable, or the result of another expression. Operators act on operands to perform specific operations or calculations.

Expressions in Python can perform various actions, including:

a) **Computations:**

Expressions can perform mathematical calculations, such as addition, subtraction, multiplication, and division. For example, the expression 3 + 5 computes the sum of 3 and 5.

b) **Evaluations:**

Expressions can evaluate conditions and return Boolean values (True or False) based on the evaluation. For example, the expression x > 10 evaluates whether the value of x is greater than 10 and returns either True or False.

c) **Assignments:**

Expressions can assign values to variables. For example, the expression x = 10 assigns the value 10 to the variable x.

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

Sol: The main difference between an expression and a statement in Python is that an expression produces a value, whereas a statement performs an action or a sequence of actions. Here's a breakdown of each:

Expression:

An expression is a combination of operands and operators that, when evaluated, produces a value. Expressions can involve literals, variables, function calls, and operators. They can be

as simple as a single value or as complex as a nested expression. Examples of expressions include:

5 + 3 # Arithmetic expression, evaluates to 8

x * 2 # Variable expression, evaluates to the value of x multiplied by 2

my_func() # Function call expression, evaluates to the return value of the function my_func()

Statement:

A statement, on the other hand, is a complete instruction or command that performs an action or changes the program's state. Statements can involve expressions, control flow structures, or declarations. They are typically composed of keywords, expressions, and punctuation. Examples of statements include:

spam = 10 # Assignment statement, assigns the value 10 to the variable spam

if x > 5: # Conditional statement, executes a block of code conditionally

for item in my_list: # Loop statement, iterates over elements in my_list

def my_func(): # Function definition statement, defines a function named my_func()

In summary, expressions produce values, whereas statements perform actions or control the flow of a program. Statements often include expressions as part of their execution.

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

bacon = 22

bacon + 1

sol: After running the given code:

bacon = 22

bacon + 1

The variable `bacon` will still contain the value `22`.

The expression `bacon + 1` calculates the sum of the value of `bacon` (which is 22) and 1. However, the result of this expression is not assigned to any variable or used in any way.

If you want to update the value of `bacon` to the result of the expression `bacon + 1`, you need to explicitly assign it back to the variable, like this:

bacon = 22

bacon = bacon + 1

Now, `bacon` will contain the value `23` after executing these two lines of code.

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

'spam' + 'spamspam'

'spam' * 3

Sol: 'spam' + 'spamspam' ===➔ ‘spamspamspam’

The + operator is used for string concatenation in Python.

'spam' * 3 ==========➔ ‘spamspamspam’

The * operator, when used with a string and an integer, repeats the string by the specified number of times.

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

Sol: By the naming conventions of python a variable name should only start with alphabets or _. Because of that reason eggs is a valid variable and 100 in not.

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

Sol: 1. `int()`: This function can be used to convert a value to an integer. It takes a value as an argument and returns the integer representation of that value if it is convertible to an integer. For example:

```
value = "42"

integer_value = int(value)  # Converts the string "42" to the integer 42
```

2. `float()`: This function can be used to convert a value to a floating-point number (decimal number). It takes a value as an argument and returns the floating-point representation of that value if it is convertible to a float. For example:

```
value = "3.14"

float_value = float(value)  # Converts the string "3.14" to the float 3.14
```

3. `str()`: This function can be used to convert a value to a string. It takes a value as an argument and returns the string representation of that value. For example:

```
value = 42
```

```
string_value = str(value)  # Converts the integer 42 to the string "42"
```

These functions allow you to convert values between different data types, namely integer, floating-point, and string, as per your requirement.

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

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

Sol: the expression will raise an error because + could be used for addition if we have all numeric values or for concatenation if we have all string value. But here we have combination of string and integer values.