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?

A string and a variable are two different concepts in programming, but they can be related. In programming, a variable is a named container that holds a value. It allows you to store and manipulate data. Variables can hold different types of data, such as numbers, strings, boolean values, or more complex objects. Moreover, a string is a specific type of data that represents a sequence of characters. It is a sequence of characters enclosed in quotation marks (either single or double).

3. Describe three different data types.

Ans. There are three different data types commonly used in programming:

(a). Integer (int): The integer data type represents whole numbers without any decimal points. Integers can be positive or negative. For example -5, 0, 10. In Python, you can perform mathematical operations on integers, such as addition, subtraction, multiplication, and division.

(b). String (str): The string data type represents a sequence of characters. It is used to store and manipulate textual data. Strings are enclosed in quotation marks (either single or double).

(c) Boolean (bool): The boolean data type represents a binary value that can be either true or false. For example: True and False.

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

Ans. An expression is made up of one or more operators and operands. It represents a computation or a value that can be evaluated by the programming language. The operands can be variables, literals (constant values), function calls, or other expressions. The operators define how the operands are combined or manipulated to produce a result. Expressions can perform a variety of operations, such as arithmetic calculations, logical evaluations, string concatenation, and more. They can be used to assign values to variables, pass arguments to functions, make decisions in control structures, or produce values that are used in further computations.

5. What is the difference between an expression and a statement?

Ans. An expression is a combination of one or more operators, operands, and other expressions that produce a value when evaluated. In simpler terms, an expression calculates or resolves to a value. Expressions can be as simple as a single variable or constant, or they can be more complex, involving mathematical or logical operations. Whereas, statement is a complete line of code or a command that performs an action. It represents an instruction or a directive that the program should execute. Statements can include expressions, but they go beyond just producing a value. Statements are used to control the flow of execution, define structures, perform assignments, and interact with the program's environment.

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

bacon = 22

bacon + 1

ans. Bacon contains 23.

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

'spam' + 'spamspam'

Ans: 'spamspamspam'

'spam' \* 3

Ans: 'spamspamspam'

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

In most programming languages, including Python, variable names have certain rules and conventions. Here's an explanation of why `'eggs'` is a valid variable name while `100` is invalid. In this context, `'eggs'` is a valid variable name because it starts with a letter and contains only letters. It adheres to the rules and conventions for variable naming in Python.In the case of `100`, it starts with a digit, violating the rule that a variable name cannot start with a digit. Variable names must begin with a letter or an underscore. However, it's important to note that `100` can still be a valid value or data literal, representing the number 100. It just cannot be used as a variable name.

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

Ans.Following are three functions to obtain the integer, floating-point number, or string version of a value:

(a). integer: This function converts a value into an integer. It takes a number or a string representation of a number as an argument and returns the corresponding integer value. If the argument is a floating-point number, it will be truncated towards zero. If the argument is a string, it should contain a valid integer representation.

(b). float`: This function converts a value into a floating-point number. It takes a number or a string representation of a number as an argument and returns the corresponding floating-point value. If the argument is an integer, it will be converted to a floating-point number.

(c). string: This function converts a value into a string. It takes any value as an argument and returns the string representation of that value. This is useful when you want to concatenate a non-string value with other strings or when you need to format a value as a string.

These functions allow you to convert values between different data types, facilitating various operations and manipulations in your Python programs.

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

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

Ans: The expression `'I have eaten ' + 99 + ' burritos.'` causes an error because you are trying to concatenate a string (`'I have eaten '`) with an integer (`99`) directly. In Python, the `+` operator for strings is used for string concatenation. However, it requires both operands to be strings.

To fix the error, you can convert the integer `99` into a string before concatenating it with the other strings. This can be done using the `str()` function:

'I have eaten ' + str(99) + ' burritos.'

By applying the `str()` function to the integer `99`, it converts it into a string, allowing the concatenation to be performed successfully.