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' => Values

-87.8 => Values

- => Expression

/ => Expression

* => Expression

6 => Values

2. What is the difference between string and variable?

A string is a data type that represents a sequence of characters enclosed in either single quotes (' ') or double quotes (" "). It can contain letters, numbers, symbols, and whitespace. Strings are used to store and manipulate textual data.

A variable, on the other hand, is a name that represents a value in memory. It is used to store data that can be accessed and manipulated throughout a program. A variable can hold various types of data, including strings, numbers, Booleans, and more.

In simple terms, a string is a specific type of data, while a variable is a placeholder that can hold different types of data, including strings.

3. Describe three different data types.

1. Integer (int): The integer data type represents whole numbers without any fractional or decimal part. It can be positive or negative. For example, 5, -10, and 0 are all integers. Integers are often used for counting, indexing, and performing mathematical operations that don't require decimal precision.

2. String (str): The string data type represents a sequence of characters enclosed within quotes (' ' or " "). Strings are used to store and manipulate textual data. They can contain letters, numbers, symbols, and whitespace. For example, "Hello, World!" and "42" are both strings. Strings support various operations such as concatenation, slicing, and formatting.

3. Boolean (bool): The boolean data type represents a binary value indicating either true or false. It is used to evaluate conditions and make logical decisions in programming. Boolean values are typically the result of comparison or logical operations. For example, 5 > 3 evaluates to True, and 10 == 2 evaluates to False. Booleans are essential for flow control and decision-making in programs.

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

An expression is made up of one or more values, variables, operators, and function calls combined in a way that produces a new value. It can represent a computation or operation that is evaluated to obtain a result.

Expressions can consist of:

1. Values: These can be literals such as numbers (e.g., 5, 3.14) or strings (e.g., "Hello").

2. Variables: These are names that represent values stored in memory. They can be used in expressions to refer to and manipulate the associated values.

3. Operators: These are symbols or keywords that perform specific operations on values. Examples of operators include arithmetic operators (+, -, \*, /), comparison operators (==, <, >), and logical operators (and, or, not).

4. Function calls: These involve invoking predefined or user-defined functions to perform specific tasks or computations. Functions can take arguments (values passed to the function) and return a value as a result.

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

The main difference between an expression and a statement in Python lies in their purpose and the actions they perform:

1. Expression: An expression is a combination of values, variables, operators, and function calls that evaluates to a value. It produces a result when it is executed or evaluated. Expressions can be used as part of larger expressions, as arguments for functions, or as the right side of an assignment statement.

Examples of expressions:

- 5 + 3

- x \* y

- len("Hello")

- a > b

2. Statement: A statement, on the other hand, is a complete unit of code that performs an action. It is a standalone instruction or a command that carries out a specific operation or controls the flow of the program. Statements do not necessarily produce a value. They can involve assignments, loops, conditionals, function definitions, and more.

Examples of statements:

- Assignment statement: spam = 10

- If statement: if x > 0:

- For loop: for item in list:

- Function definition: def greet(name):

Statements are typically used to control the execution flow and perform actions, such as modifying variables, iterating over data, or executing specific blocks of code conditionally. They may contain expressions within them, but they are not limited to just expressions.

In summary, expressions are evaluated to produce a value, while statements perform actions or control the flow of a program. Expressions can be part of a statement or used within statements to calculate values or determine conditions.

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

bacon = 22

bacon + 1

**23**

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

'spam' + 'spamspam'

'spam' \* 3

**'spamspamspam'**

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

In Python, variable names need to follow certain rules and conventions. Here are a few reasons why "eggs" is a valid variable name while "100" is invalid:

1. Starting with a letter: Python variable names must start with a letter (a-z or A-Z) or an underscore (\_). In this case, "eggs" starts with a letter ('e'), making it a valid variable name. However, "100" starts with a number ('1'), violating this rule.

2. Consisting of letters, numbers, or underscores: Valid variable names can consist of letters, numbers, and underscores, but they cannot start with a number. "eggs" contains only letters, which is allowed. However, "100" consists entirely of numbers, which makes it an invalid variable name.

3. Avoiding reserved keywords: Python reserves certain keywords for its syntax and built-in functions, and these cannot be used as variable names. "eggs" is not a reserved keyword, so it can be used as a variable name. On the other hand, "100" is not a valid variable name because it consists only of numbers and is not a valid identifier.

It's worth noting that although "100" is not a valid variable name, it can be used as a value or within expressions. For example, you can assign it to a variable like this: `number = 100`. However, it cannot be used as a standalone variable name.

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

To get the integer, floating-point number, or string version of a value in Python, you can use the following built-in functions:

1. Integer: `int()`

The `int()` function can be used to convert a value into an integer. It takes a numeric string or a floating-point number and returns the equivalent integer value.

2. Floating-point number: `float()`

The `float()` function converts a value into a floating-point number. It can convert an integer, numeric string, or even scientific notation into a floating-point representation

3. String: `str()`

The `str()` function is used to convert a value into a string representation. It can convert integers, floating-point numbers, Booleans, and other types into their string equivalents.

These functions allow you to convert values from one data type to another, enabling you to perform specific operations or format data as needed in your Python programs.

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

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

The expression 'I have eaten ' + 99 + ' burritos.' causes an error because you are attempting to concatenate a string ('I have eaten ') with an integer (99) directly. In Python, concatenation of strings can only be done between two strings, not between a string and an integer. To fix this error, you need to convert the integer 99 into a string before concatenating it.