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.

\*

'hello'

-87.8

-

/

6

the values are:

'hello' (string value) -87.8 (floating-point value) 6 (integer value)

The expressions are:

* (multiplication operator)
* (subtraction operator) / (division operator)
* (addition operator)

2. What is the difference between string and variable?

String: A string is a data type used to represent a sequence of characters. It is typically used to store and manipulate textual data. In most programming languages, strings are enclosed in single quotes ('') or double quotes (""). For example, 'hello' and "world" are both strings.

Variable: A variable is a named storage location in memory that holds a value. It is used to store and refer to different types of data, including strings. Variables have a name and a data type, and they can be assigned a value, which can be changed during the execution of a program. Variables allow you to store and manipulate data dynamically. For example, you can declare a variable called "name" and assign a string value to it, such as "John".

Describe three different data types.

Integer: An integer is a data type used to represent whole numbers without any fractional or decimal parts. It can be either positive or negative. For example, 1, -5, and 1000 are all integers. Integers are typically used in operations that involve counting, indexing, or representing discrete quantities.

String: A string is a data type used to represent a sequence of characters. It is commonly used to store and manipulate textual data. Strings are typically enclosed in single quotes ('') or double quotes (""). For example, "Hello, World!" is a string. Strings can contain letters, numbers, symbols, and whitespace. They are often used for handling user input, displaying messages, and manipulating text data.

Boolean: A boolean is a data type that represents a logical value, which can be either true or false. Booleans are useful for making decisions and controlling the flow of a program. They are typically used in conditional statements, loops, and boolean expressions. For example, a boolean variable "isSunny" can be true if the weather is sunny or false if it's not. Booleans can be combined using logical operators such as AND, OR, and NOT to perform more complex evaluations.Top of Form

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

An expression is made up of one or more operators, operands, and sometimes function calls or constants. It represents a computation that produces a value.

Operators: Operators are symbols or keywords that perform specific operations on one or more operands. They can be arithmetic operators (such as + for addition or \* for multiplication), comparison operators (such as == for equality or > for greater than), logical operators (such as && for logical AND or || for logical OR), or other types of operators depending on the programming language.

Operands: Operands are the values or variables that the operators act upon. They can be literals (specific values like numbers or strings), variables (representing values stored in memory), or other expressions. For example, in the expression 2 + 3, the numbers 2 and 3 are operands.

Function calls: Expressions can also include function calls, which are used to invoke predefined or user-defined functions. Functions can perform complex operations and return values, and their results can be part of an expression. For example, in the expression sin(x), the function call sin() is an expression that calculates the sine of the value x.

* 1. 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 lies in their purpose and behavior within a programming language:

Expression:

* An expression is a combination of operators, operands, and/or function calls that evaluates to a value.
* It can be as simple as a single value or as complex as a combination of multiple operations.
* Expressions are typically used to compute or calculate values.
* Examples of expressions: 5 + 3, x \* y, "Hello, " + name

Statement:

* A statement is a complete instruction or command that performs an action or controls the flow of a program.
* It doesn't necessarily produce a value or result.
* Statements are used to control the execution of code, define control structures (e.g., loops or conditionals), or interact with the program's environment.
* Examples of statements: assignment statements, control flow statements (if-else, while, for), function declarations, input/output statements.

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

bacon = 22

bacon + 1

The variable bacon will still contain the value 22.

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

'spam' + 'spamspam'

'spam' \* 3

1. 'spam' + 'spamspam': The result of this expression is the string 'spamspamspam'. The + operator, when used with strings, performs concatenation, meaning it combines the two strings together into a single string.
2. 'spam' \* 3: The result of this expression is the string 'spamspamspam'. The \* operator, when used with a string and an integer, repeats the string multiple times. In this case, the string 'spam' is repeated three times, resulting in the string 'spamspamspam'.

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

In programming, variable names need to adhere to certain rules and conventions set by the language. Here's why 'eggs' is a valid variable name while 100 is invalid:

1. Starting Character: In many programming languages, including Python, variable names cannot begin with a digit. They must start with a letter (uppercase or lowercase) or an underscore (\_). Since 'eggs' starts with a letter, it is a valid variable name.
2. Valid Characters: Variable names can typically contain letters (uppercase or lowercase), digits, and underscores. However, they must not contain spaces or special characters (such as punctuation marks). Since 'eggs' only contains valid characters, it is a valid variable name.
3. Conventions: Although 100 is a valid number, it is not a suitable variable name because it does not follow common naming conventions. Variable names should be descriptive, meaningful, and help understand the purpose of the data they represent. Starting a variable name with a number can make it confusing and harder to understand the variable's purpose. It's generally recommended to use more descriptive names that convey the meaning of the data being stored.

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

int(): This function can be used to convert a value to an integer. It takes a parameter and tries to convert it into an integer. If the conversion is successful, it returns the integer value; otherwise, it raises a ValueError

float(): This function can be used to convert a value to a floating-point number (a decimal number). It takes a parameter and attempts to convert it into a float. If the conversion succeeds, it returns the float value; otherwise, it raises a ValueError

str(): This function can be used to convert a value to a string. It takes a parameter and returns a string representation of that value. It is useful when you want to convert a non-string value, such as an integer or a float, to a string

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 it attempts to concatenate a string ('I have eaten ') with an integer (99) directly using the + operator. In Python, the + operator for string concatenation requires both operands to be strings.

To fix this error, you need to ensure that all operands being concatenated are strings.