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

Ans1) this are values

1)’hello’(string)

2)-87.8(floating-point number)

6(integer)

This are Expression:

\*(multiplication operator)

-(subtraction operator)

/(division operator)

+(addition operator)

2. What is the difference between string and variable?

ans2) String - A string is a data type used to represent text in programming.It is a sequence of characters enclosed within single (‘ ‘) or double (“”) quotation marks.

Example:’hello’, ‘123’, ‘This is a string’

Variable- A variable is a symbolic name that represents or refers to value stored in the computer’s memory.

Variables can hold various types of data,including strings, integers,floating-point numbers, boolean, and etc.

3. Describe three different data types?

ans3)Integer:

An integer is a data type that represents whole numbers, both positive and negative, without any fractional or decimal parts. Integers are used to represent quantities that can be counted, such as the number of items in a shopping cart or the age of a person.

Floating-point:

Floating-point is a data type used to represent numbers that have both an integer and a fractional part. These numbers are stored in a computer's memory using a format that includes a sign bit, a significant or mantissa, and an exponent

String:

A string is a data type used to represent a sequence of characters, such as letters, digits, and symbols. Strings are often used to store textual data in computer programs, such as names, addresses, or messages. In programming languages, strings are typically enclosed within single quotes (' ') or double quotes (" "), and operations such as concatenation (joining strings together) or slicing (extracting substrings) can be performed on them.

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

An expression is made up of operands and operators. Operands are the values or variables on which the operators act, and operators specify the operations to be performed. Expressions can be simple, involving just one operand and one operator (e.g., 5 + 3), or they can be complex, involving multiple operands and operators (e.g., (5 \* 3) - (2 / 4)).

Expressions are fundamental to programming because they allow you to perform calculations, manipulate data, and make decisions based on conditions.

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

Expression:

* An expression is a combination of values, variables, operators, and function calls that evaluates to a single value.
* Expressions are typically used to represent computations or to produce data values.
* Examples of expressions include mathematical operations (e.g., 5 + 3), function calls (e.g., math.sqrt(25)), or variable assignments (e.g., x = 10).

Statement:

A statement is a complete unit of execution that performs some action.

Statements are used to control the flow of a program, define the structure, and perform operations like assignments, loops, conditionals, function definitions, etc.

Examples of statements include variable assignments (e.g., spam = 10), conditional statements (e.g., if-else), loop statements (e.g., for, while), function definitions, and so on.

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

bacon = 22

bacon + 1

Initially, the variable "bacon" is assigned the value 22 with the statement bacon = 22.

Then, the expression bacon + 1 is evaluated, which results in 23. However, this result is not assigned to any variable, nor is it printed or used in any other way. It's simply evaluated and the result is not stored or captured anywhere.

Since the value 23 resulting from the expression bacon + 1 is not stored back into the variable "bacon" or used in any way, the value of "bacon" remains unchanged at 22.

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

'spam' + 'spamspam'

'spam' \* 3

'spam' + 'spamspam':

This expression involves concatenating two strings together. It results in the string 'spamspamspam'. When you concatenate strings using the + operator, it joins the two strings together, forming a longer string.

'spam' \* 3:

This expression involves multiplying a string by an integer. It results in the string 'spam' repeated three times consecutively, like 'spamspamspam'. When you use the \* operator with a string and an integer, it repeats the string multiple times according to the value of the integer. In this case, 'spam' is repeated three times.

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

Validity rules for variable names:

Variable names typically must begin with a letter (a-z, A-Z) or an underscore (\_). They cannot begin with a number.

After the initial character, variable names can contain letters, numbers, or underscores.

Based on these rules:

* 'eggs' is a valid variable name because it starts with a letter and contains only letters, making it conform to the rules.
* '100' is invalid because it starts with a number and does not contain any letters or underscores. Variable names cannot start with a number in most programming languages, including Python.

In essence, while 'eggs' follows the rules for variable names, '100' violates the rule that variable names cannot start with a number.

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 into an integer. If the value is a floating-point number, it will be truncated towards zero (i.e., the decimal part will be discarded). If the value is a string representation of an integer, it will be converted into an integer.

float(): This function can be used to convert a value into a floating-point number. If the value is an integer, it will be converted into a floating-point number. If the value is a string representation of a floating-point number, it will be converted into a floating-point number.

str(): This function can be used to convert a value into a string. Regardless of whether the value is an integer, a floating-point number, or something else, str() will convert it into a string representation.

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

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

This expression causes an error because you're trying to concatenate a string ('I have eaten ') with an integer (99) directly, and then trying to concatenate the result with another string (' burritos.'). In Python, you can only concatenate strings with other strings, not with integers.

To fix it, you need to convert the integer (99) into a string before concatenating it with the other strings. You can do this by using the str() function to convert the integer into a string

.