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

ANSWER:

\* - / + are expressions

‘hello’ -87.8 6 are values

2. What is the difference between string and variable?

ANSWER: A string is a specific type of data representing text, while a variable is a symbolic name used to store and manipulate various types of data within a program. Strings are a type of data, and variables are placeholders for storing different types of data. In programming, variables are often used to store strings and other types of data for processing and manipulation.

3. Describe three different data types.

ANSWER:

**Integer (int)**:

* Integers are a fundamental data type used to represent whole numbers, both positive and negative, without any fractional or decimal part.
* Examples of integers include -10, 0, 42, 100, etc.

**Floating-point (float)**:

* Floating-point numbers, often referred to as floats, are used to represent real numbers that include both integer and fractional parts.
* Examples of floating-point numbers include 3.14, -0.5, 2.71828, etc.

**String (str)**:

* Strings are a sequence of characters, such as letters, digits, symbols, or whitespace, enclosed within single (' ') or double (" ") quotation marks.
* Examples of strings include "Hello, World!", 'Python', "12345", etc.

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

ANSWER: An expression in programming is made up of operands and operators. Here's what each of these terms means:

1. **Operands**:
   * Operands are the values or data on which operations are performed within an expression.
   * Operands can take various forms depending on the programming language and context. They can be constants, variables, function calls, or even other expressions.
   * Examples of operands:
     + Constants: 5, 3.14, 'Hello'
     + Variables: x, y, result
     + Function calls: sum(3, 5), len('Hello')
     + Other expressions: x + y, (3 \* a) - (2 \* b)
2. **Operators**:
   * Operators are symbols or keywords that specify the type of operation to be performed on the operands.
   * Operators define how the operands are manipulated to produce a result.
   * There are different types of operators, such as arithmetic operators, comparison operators, logical operators, assignment operators, etc.
   * Examples of operators:
     + Arithmetic operators: +, -, \*, /, %
     + Comparison operators: ==, !=, <, >, <=, >=
     + Logical operators: and, or, not
     + Assignment operators: =, +=, -=, \*=, /=

An expression can consist of a single operand (e.g., 5, x, len('Hello')) or multiple operands combined with operators (e.g., x + y, 2 \* a - 3 \* b, x == y). Expressions can also be nested within each other, forming more complex expressions.

Now, what do all expressions do?

Expressions in programming languages generally evaluate to a value. When an expression is evaluated, the operands are processed according to the operators, and a resulting value is produced. This value can be of any data type depending on the expression and the operations performed. For example:

* In the expression 3 + 5, the operator + adds the operands 3 and 5, resulting in the value 8.
* In the expression x == y, the operator == checks if the values of x and y are equal and evaluates to either True or False depending on the comparison.

In summary, expressions are fundamental constructs in programming that combine operands and operators to produce values. They can be simple or complex, but their primary purpose is to evaluate to a result.

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

ANSWER: In programming, expressions and statements are both fundamental concepts, but they serve different purposes and have different characteristics:

**Expression**:

* + An expression is a combination of values, variables, operators, and function calls that evaluates to a single value.
  + Expressions are used to perform computations, produce results, or represent values.
  + Examples of expressions:
  + Arithmetic expressions: 2 + 3, x \* y, a / (b - c)
  + Comparison expressions: x == y, a < b, len(s) > 10
  + Function call expressions: print('Hello'), math.sqrt(16)

**Statement**:

* + A statement is a complete line of code that performs an action, defines a variable, controls program flow, or does some other operation.
  + Statements are used to express the logic and structure of a program.
  + Unlike expressions, statements do not necessarily evaluate to a value.
  + Examples of statements:
  + Assignment statements: spam = 10, x = 5 + 3
  + Control flow statements: if, else, while, for
  + Function definition statements: def my\_function():, class MyClass:
  + Input/output statements: print('Hello'), input('Enter a number: ')

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

bacon = 22

bacon + 1

ANSWER: bacon contains 23

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

'spam' + 'spamspam'

'spam' \* 3

ANSWER: The result for both the operations are same ‘spamspamspam’

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

ANSWER: In most programming languages, variable names must adhere to certain rules and conventions. These rules typically dictate what characters are allowed in variable names and what conditions they must satisfy.

Eggs starts with a letter which is allowed as per convention. 100 starts with a number which is not allowed.

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

ANSWER:

**Integer conversion**:

* Use the int() function to convert a value to an integer.
* If the value is a floating-point number, it will be truncated towards zero.
* If the value is a string representing an integer, it will be converted to an integer.

**Floating-point conversion**:

* Use the float() function to convert a value to a floating-point number.
* If the value is an integer, it will be converted to a floating-point number.
* If the value is a string representing a floating-point number, it will be converted to a floating-point number.

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

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

ANSWER: 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. In Python, concatenation using the + operator is only allowed between strings, not between strings and integers.

To fix this error, you can convert the integer 99 to a string using the str() function so that it can be concatenated with the other strings.