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

Ans:

Values: 'hello', -87.8, 6

Expressions: -, /, +

2. What is the difference between string and variable?

String is a value (text), while a Variable is a name used to hold a value.

3. Describe three different data types.

Integer is whole numbers without decimals.

Ex: 42, -7

String is a sequence of characters (text).

Ex: "Hello", "1234"

Boolean represents True or False values.

Ex: True, False

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

An expression in programming is a combination of variables, values, operators, and sometimes function calls that are evaluated to produce a result. Essentially, it is a construct that the program can "evaluate" to compute a value.

An expression can be made up of:

Operands (e.g., values or variables):

These are the entities on which operations are performed. They can be constants (like numbers or strings) or variables that hold data.

Examples: 5, x, "hello", True

Operators (e.g., mathematical or logical operators):

These define the operations to be performed on the operands. Operators can be arithmetic, comparison, logical, etc.

Examples:

Arithmetic operators: +, -, \*, /

Comparison operators: ==, !=, <, >

Logical operators: and, or, not

Functions (optional in more complex expressions):

Functions can be part of an expression, where the function returns a value that is then used in the rest of the expression.

Example: abs(-5), len("hello")

Parentheses (optional for controlling order of operations):

Parentheses can group parts of an expression to ensure the correct order of operations (just like in mathematical expressions).

Example: (x + y) \* z

What Do All Expressions Do?

All expressions in programming are evaluated to produce a value or a result. When the expression is evaluated, it goes through the following steps:

Evaluation:

The program computes the result by applying the operators on the operands (values or variables).

If there are multiple operators, the program follows the order of precedence (like in mathematics) to determine how the operands are combined.

Result:

The output of the evaluation of an expression is a value (which could be a number, string, boolean, etc.).

The result is typically used in the program for further operations, stored in variables, or returned by functions.

Types of Expressions:

Arithmetic Expressions: Involve mathematical calculations.

Example: x + 5, a \* b - c

Relational Expressions: Used for comparisons, producing boolean results (True or False).

Example: x > y, a == b

Logical Expressions: Combine boolean values using logical operators.

Example: x and y, not isTrue

String Expressions: Combine strings (concatenate them) or manipulate them.

Example: "Hello" + " " + "World!"

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

Expression:

An expression always evaluates to a value.

Can be used inside other statements (e.g., in conditionals or assignments).

Example: x + 5 is an expression that will evaluate to a value (a number).

Statement:

A statement performs an action.

It does not return a value (it may modify the state or control the program flow).

Example: spam = 10 is a statement that assigns the value 10 to the variable spam.

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

bacon = 22

bacon + 1

After running the code, the variable bacon still contains 22.

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

'spam' + 'spamspam'

'spam' \* 3

Ans: 'spam' + 'spamspam': This concatenates the two strings, so the result is 'spamspamspam'.

'spam' \* 3: This repeats the string 'spam' three times, so the result is also 'spamspamspam'.

Both expressions yield the same result: 'spamspamspam'.

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

Why eggs is valid:

Starts with a letter: Variable names must begin with a letter (a-z, A-Z) or an underscore (\_).

eggs starts with a letter.

Contains valid characters: After the first character, variable names can only include letters, numbers, or underscores.

eggs meets this requirement, as it contains only letters.

Not a reserved keyword: It is not a Python keyword like if, else, or for.

Why 100 is invalid:

Cannot start with a digit: Variable names cannot start with a number.

100 starts with a digit, so it violates this rule.

Does not adhere to identifier rules: Numbers alone are not valid variable names in Python.

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

Ans: int(): Converts a value to an integer (if possible).

Example: int(3.7) → 3

Example: int("42") → 42

float(): Converts a value to a floating-point number (if possible).

Example: float(3) → 3.0

Example: float("4.2") → 4.2

str(): Converts a value to a string.

Example: str(42) → "42"

Example: str(3.14) → "3.14"

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

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

Ans: Why it causes an error:

Python cannot concatenate a string ('I have eaten ') with an integer (99) directly.

The + operator expects both operands to be of the same type.

How to fix it:

You need to convert the integer to a string using the str() function. Here's the corrected