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.

\*

&#39;hello&#39;

-87.8

-

/

+

6

Ans: In the given list,the values are ‘hello’,-87.8,and 6.the expressions are \*,-,/,and +.

2. What is the difference between string and variable?

Ans: Strings: A string is a data type used to represent text or a sequence of characters. In most programming languages, strings are enclosed in single quotes (' '), double quotes (" "), or triple quotes (""" """). Examples of strings:

"Hello, World!"

'Programming is fun!'

"12345"

Strings can contain letters, numbers, symbols, and spaces, and they are often used to store and manipulate textual data.

Variables: A variable is a named container or storage location in a program that holds a value. Variables allow you to store and manipulate data during program execution. Each variable has a unique name and a data type associated with it. The data type defines the kind of data the variable can hold, such as integers, floating-point numbers, characters, or strings.

Here's an example of using a variable to store a string:

message= “Hello world”

print(message)

we declare a variable called ‘message’ and assign it the string value ”Hello world”.we print the value of the variable using ‘print()’ function,which outputs the string ‘”Hello world”’.

Top of Form

Regenerate response

Top of Form

3. . Describe three different data types

Ans: Integer (int): An integer data type represents whole numbers without any fractional or decimal parts. Integers can be either positive or negative numbers or zero. They are typically used for counting, indexing, and performing arithmetic operations that do not require fractional precision.

Floating-Point (float): The floating-point data type represents numbers with decimal points or fractional parts. Floating-point numbers are used when precision is needed in calculations, such as in scientific and financial applications. Floating-point numbers can represent a wide range of values, including very large and very small numbers.

It's important to note that floating-point numbers may not always be represented precisely due to the limited precision of floating-point representations in computer systems, leading to rounding errors in certain calculations

String (str): As mentioned earlier, the string data type represents a sequence of characters. Strings are used to store and manipulate textual data, such as names, sentences, or any sequence of letters, numbers, and symbols. Strings are typically enclosed in single quotes (' '), double quotes (" "), or triple quotes (""" """) depending on the programming language.

Top of Form

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

Ans: An expression in programming is made up of one or more operands and operators. It represents a computation or a value that can be evaluated to produce a result.

Operand: An operand is a value or a variable that is used in an expression. It can be a literal value (e.g., numbers or strings) or a variable (e.g., x, y). Operands are the basic building blocks of an expression and serve as the data that the expression manipulates.

Examples of operands:

5 (a numeric literal)

"Hello" (a string literal)

x (a variable)

Operator: Operators are symbols or special keywords that perform operations on the operands. They define how the operands should be combined, compared, or modified within the expression. Operators can be arithmetic (e.g., + for addition, - for subtraction), logical (e.g., && for logical AND, || for logical OR), relational (e.g., < for less than, == for equal to), and more, depending on the programming language.

Examples of operators:

+ (addition)

- (subtraction)

\* (multiplication)

/ (division)

== (equality check)

> (greater than)

Top of Form

5. This assignment statements, like spam = 10. What is the difference between an

expression and a statement?

Ans: The main difference between an expression and a statement lies in their purpose and behavior:

the program's state. Unlike expressions, statements do not produce values as their primary purpose is to execute a specific task or control the program's flow.

Expression: An expression is a combination of operands and operators that evaluates to a value. It represents a computation and can be used to perform calculations or produce results. Expressions do not change the state of the program; they only produce a value.

Examples of expressions:

**2 + 3** (evaluates to **5**)

**x \* y** (evaluates to the product of variables **x** and **y**)

**"Hello, " + name** (evaluates to a concatenated string)

Statement: A statement, on the other hand, is a complete instruction or command that performs an action or changes.

Examples of statements:

Assignment statement: **spam = 10** (assigns the value **10** to the variable **spam**)

Conditional statement (if-else):

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

bacon = 22

bacon + 1

Ans: After running the provided code:

bacon = 22 bacon + 1

The variable **bacon** will still contain the value **22**.

Let's break down the code:

**bacon = 22**: This line assigns the value **22** to the variable **bacon**.

**bacon + 1**: This expression computes the value of **bacon + 1**, which is **23**. However, this expression is not used in any assignment or output statement. It's essentially a standalone expression that does not change the value of **bacon** or have any side effects.

Since there is no assignment of the result of **bacon + 1** back to the variable **bacon**, the value of **bacon** remains unchanged, and it will still be **22** after the code execution. If you want to update the value of **bacon**, you need to assign the result back to the variable explicitly, like this:

bacon = 22 bacon = bacon + 1 # Or you can use the shorthand: bacon += 1

Now bacon will be 23 after this code snippet execute .

Top of ForBottom of Form

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

‘spam’ + ‘spamspam’

‘spam’ \* 3

Ans: Let's evaluate the two terms:

**'spam' + 'spamspam'**: When you use the **+** operator with two strings, it performs string concatenation, meaning it combines the two strings into one.

Result: **'spamspamspam'**

**'spam' \* 3**: When you use the **\*** operator with a string and a number, it performs string replication, which repeats the string a specified number of times.

Result: **'spamspamspam'**

So, the values of both terms will be **'spamspamspam'**. The first term uses the **+** operator to concatenate two strings, and the second term uses the **\*** operator to replicate the string **'spam'** three times, resulting in the same output for both.

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

Ans: Valid Variable Name (eggs): Variable names can start with a letter or an underscore, followed by letters, numbers, or underscores. Since "eggs" starts with a letter 'e', it is a valid variable name.

Invalid Variable Name (100): Variable names cannot start with a number. "100" starts with a number '1', so it is an invalid variable name.

9. What three functions can be used to get the integer, floating-point number, or string

version of a value?

Ans:To get the integer version of a value, you can use int(). It's like taking a number and cutting off any decimal parts, so you have a whole number.

Example:

python code

x = int(5.7) print(x) # Output: 5

To get the floating-point number version of a value, you can use float(). It's like adding a dot to the number, even if it didn't have one before.

Example:

python code

a = float(10) print(a) # Output: 10.0

To get the string version of a value, you can use str(). It's like putting the value inside quotes so that it becomes text.

Example:

python code

name = "Alice" print(str(name)) # Output: "Alice"

Top of Form

Bottom of Form

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

‘I have eaten’ + 99 + ‘burritos’.

Ans: The expression 'I have eaten' + 99 + 'burritos' causes an error because you cannot directly combine words with a number.

To fix it, convert the number 99 into a word (string) using str():

pythonCopy code

'I have eaten ' + str(99) + ' burritos'

The output will be: 'I have eaten 99 burritos'.