**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**.

**\* = Mathematical operators**

**'hello' = String Value**

**-87.8 = Floating-point number value**

**- = Mathematical operators**

**/ = Mathematical operators**

* **= Mathematical operators**

**6 = integer value.**

2. **What is the difference between string and variable?**

**.** **A string is a data type in programming that represents a sequence of characters. It is used to store and manipulate text. In most programming languages, strings are enclosed in single quotes ('') or double quotes ("").**

**. A variable, on the other hand, is a named storage location in a computer's memory that holds a value. It is used to store data that can be referenced and manipulated in a program. Variables can hold different types of data, including strings, numbers, booleans, and more.**

**3. Describe three different data types.**

**Integer (int): The integer data type represents whole numbers without any fractional or decimal parts. It can be both positive and negative numbers, including zero. Examples of integers are -5, 0, 42, and 100. Integer data types are often used for counting, indexing, and performing arithmetic operations.**

**Floating-point (float): The floating-point data type represents numbers with fractional or decimal parts. It is used to store real numbers, such as 3.14 or -0.5. Floating-point numbers can have a varying degree of precision depending on the programming language and the specific implementation. They are commonly used in mathematical calculations, scientific computations, and when dealing with measurements or values that can have a fractional component.**

**String (str): The string data type represents a sequence of characters, such as letters, numbers, symbols, and whitespace. Strings are used to store textual information, like names, sentences, or any other collection of characters. They are typically enclosed in single quotes ('') or double quotes ("") in most programming languages. Examples of strings include "Hello, world!", "OpenAI", and "12345". String data types are widely used in various applications for text processing, user input/output, and manipulation of textual data.**

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

**An expression is made up of one or more operands and operators.**

**Operands: These are the values or variables that the expression operates on. For example, in the expression 2 + 3, the operands are the numbers 2 and 3.**

**Operators: These are symbols or keywords that represent specific operations to be performed on the operands. They define how the operands should be combined or manipulated. In the example expression 2 + 3, the operator is the addition symbol (+), which adds the two operands together.**

**Expressions can also include other elements such as parentheses, function calls, and constants. They can be as simple as a single value or as complex as a combination of multiple operations.**

**The purpose of an expression is to compute or evaluate a value. When an expression is evaluated, it produces a result or a value based on the operations performed on the operands. The result of an expression can be assigned to a variable, used in further computations, or used to control the flow of a program.**

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

**The key difference between an expression and a statement lies in their functionality and purpose within a programming language:**

**Expression:**

**An expression is a combination of operands, operators, and other elements that evaluates to a value.**

**It can be as simple as a single value or complex, involving multiple operations and function calls.**

**Expressions can be used within statements, as part of larger expressions, or as standalone entities.**

**The primary purpose of an expression is to compute or calculate a value.**

**Example: In the expression 2 + 3, the addition operator combines the operands 2 and 3 to evaluate to the value 5.**

**Statement:**

**A statement, on the other hand, is a complete instruction or action that performs a specific task.**

**It is a unit of code that carries out an operation, controls the flow of the program, or manipulates data.**

**Statements are typically composed of expressions, keywords, and control flow structures.**

**Unlike expressions, statements do not necessarily produce a value or have a return value.**

**Statements can include assignments, loops, conditionals, function declarations, and more.**

**Example: The assignment statement spam = 10 assigns the value 10 to the variable spam.**

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

**bacon = 22**

**bacon + 1**

**OUTPUT:**

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

**In the code snippet provided, the first line bacon = 22 assigns the value 22 to the variable bacon. However, in the second line bacon + 1, the expression bacon + 1 evaluates to 23, but the result is not assigned to any variable. Therefore, the value of bacon remains unchanged at 22.**

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

**'spam' + 'spamspam'**

**'spam' \* 3**

**Output:**

**'spam' + 'spamspam':**

**The result would be the concatenation of the two strings: 'spamspamspam'. The + operator, when used with strings, performs string concatenation, meaning it combines the two strings together.**

**'spam' \* 3:**

**The result would be the repetition of the string 'spam' three times: 'spamspamspam'. The \* operator, when used with a string and an integer, repeats the string by the specified number of times.**

**In both cases, the final value is the same, which is 'spamspamspam'.**

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

**Start with a letter or underscore: Variable names should begin with a letter (a-z or A-Z) or an underscore (\_). They cannot start with a digit.**

**Since 'eggs' starts with the letter 'e', it satisfies this rule and is a valid variable name.**

**Can contain letters, digits, and underscores: After the initial character, variable names can include letters, digits (0-9), and underscores (\_). They cannot contain spaces or special characters.**

**Since 'eggs' only contains letters, it satisfies this rule and is a valid variable name.**

**On the other hand, '100' starts with a digit, which violates the first rule for variable names. Variable names cannot begin with a digit in most programming languages, as it would cause ambiguity with numeric literals. This is because the interpreter/compiler needs to distinguish between a variable name and a literal value.**

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

**Integer: int()**

**The int() function is used to convert a value to an integer. It takes a numeric value or a string that represents an integer and returns the corresponding integer value. If the value is a floating-point number, it will be truncated (rounded towards zero) to the nearest whole number.**

**Ex: x = int(3.14)**

**print(x)**

**Floating-Point Number: float()**

**The float() function is used to convert a value to a floating-point number. It takes a numeric value or a string that represents a number (integer or floating-point) and returns the corresponding floating-point value.**

**Ex: x = float(5)**

**print(x)**

**String: str()**

**The str() function is used to convert a value to a string. It takes any value and returns its string representation.**

**Ex: x = str(42)**

**print(x)**

**These functions allow you to convert values between different data types, enabling you to manipulate and work with them in various ways based on your programming needs**

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

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

**The given expression 'I have eaten ' + 99 + ' burritos.' causes an error because it attempts to concatenate a string with an integer directly. In many programming languages, including Python, string concatenation can only be performed between strings.**

**To fix the error, you can convert the integer 99 to a string using the str() function before concatenating it with the other strings.**

**Here's an updated version of the expression with the integer converted to a string:**

**By using str(99), we convert the integer 99 to the string '99'. Now, all the operands in the expression are strings, and the concatenation will work correctly without causing an error.**