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.

* →Expression

'hello'→Values

-87.8→Values

- →Expression

/ →Expression

+ → Expression

6 →Values

2. What is the difference between string and variable?

Ans: A string is a sequence of characters that is treated as text data in a program. It is a data type used to represent textual data, such as a name, address, or message. In most programming languages, strings are enclosed in quotation marks (" " or ' '). For example, "Hello, World!" is a string.

On the other hand, a variable is a named storage location in a program that holds a value, which can be a string, a number, or any other data type. Variables are used to store data that can change during the execution of a program. They allow programmers to create reusable code by assigning values to a variable and then using that variable throughout the program. For example, a variable named "name" can store a string value, such as "John".

In summary, a string is a data type that represents textual data, while a variable is a named storage location that can hold any type of data, including strings. A variable can store a string, but a string cannot store a variable.

3. Describe three different data types.

Ans: In programming, data types define the type of data that can be stored and manipulated in a program. Here are three common data types:

Integer: An integer is a whole number that can be positive, negative, or zero. It is typically used to represent countable quantities, such as the number of items in a list or the index of an element in an array.

Floating-point: A floating-point number is a number with a decimal point. It is typically used to represent real-world quantities, such as temperature or distance. Floating-point numbers are stored in a format that allows for a certain amount of precision, but they may not be able to represent every possible real number.

Boolean: A boolean is a data type that can have only two possible values: true or false. It is often used to represent conditions or logical values in a program. For example, a boolean variable could be used to represent whether a user is logged in or not, or whether a certain condition has been met.

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

Ans: In programming, an expression is a combination of variables, values, operators, and function calls that can be evaluated to produce a result. Expressions can be used to represent calculations, comparisons, or any other operation that can be performed on data.

An expression can be made up of one or more operands and operators. An operand can be a variable, a value, or the result of a function call. An operator is a symbol or keyword that performs a specific operation on one or more operands.

All expressions in programming have one thing in common: they produce a value. The value that an expression produces can be assigned to a variable, used as part of a conditional statement, or passed as an argument to a function, among other things.

For example, consider the following expression:

5+3*2

This expression consists of the values 5 and 3, the operators + and *, and the operands 2 and the result of 3 * 2. When this expression is evaluated, it produces the value 11.

Expressions are an essential part of programming and are used extensively to represent various computations and operations.

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

Ans: In programming, an expression is a combination of values, variables, operators, and function calls that can be evaluated to produce a result, whereas a statement is a complete line of code that performs an action or sets a value.

An assignment statement, like spam = 10, is an example of a statement. It sets the value of the variable spam to 10.

On the other hand, an expression can be part of a statement or a standalone statement. For example, the following are expressions:

```
3 + 4

spam * 5

max(2, 6)
```

Expressions can also be assigned to variables or used in larger expressions. For example:

```
eggs = spam + 5

result = (3 + 4) * eggs
```

In summary, an expression is a combination of values, variables, operators, and function calls that can be evaluated to produce a result, while a statement is a complete line of code that performs an action or sets a value. An expression can be part of a statement, but a statement is not necessarily an expression.

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

bacon = 22

bacon + 1

Ans: After running the following code:

```
bacon = 22

bacon + 1
```

The variable bacon still contains the value 22 because the code has not assigned the result of bacon + 1 back to the bacon variable.

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

'spam' + 'spamspam'

'spam' * 3

Ans: The values of the following two terms would be the same:

'spam' + 'spamspam'   # Result is 'spamspamspam'

'spam' * 3   # Result is 'spamspamspam'

Both expressions result in a string that contains the characters 'spam' repeated three times.

In the first expression, the + operator concatenates the two strings 'spam' and 'spamspam' together to form a new string 'spamspamspam'.

In the second expression, the * operator repeats the string 'spam' three times to form a new string 'spamspamspam'. Therefore, both expressions result in the same string: 'spamspamspam'.

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

Ans: In programming, variable names must follow certain rules and conventions.

One of the rules for variable names is that they must start with a letter or an underscore (_), and can be followed by letters, numbers, or underscores.

Since 'eggs' starts with a letter, it is a valid variable name. However, '100' starts with a number, which violates the rule, and hence is not a valid variable name.

Variable names should also be chosen in a way that is meaningful and descriptive, making the code more readable and easier to understand.

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

Ans: In Python, there are three built-in functions that can be used to get the integer, floating-point number, or string version of a value:

int() function: This function is used to convert a value to an integer. It can convert a string or a floating-point number to an integer. For example:

int('123')   # Returns 123

int(3.14)   # Returns 3

float() function: This function is used to convert a value to a floating-point number. It can convert a string or an integer to a floating-point number. For example:

```
float('3.14')  # Returns 3.14

float(123)     # Returns 123.0
```

str() function: This function is used to convert a value to a string. It can convert an integer, a floating-point number, or any other value to a string. For example:

```
str(123)   # Returns '123'

str(3.14)  # Returns '3.14'
```

These functions are commonly used in programming to convert values from one type to another.

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 it is attempting to concatenate a string ('I have eaten ') with an integer (99) without first converting the integer to a string.

To fix this error, you can convert the integer to a string using the str() function, like this:

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

This will convert the integer 99 to a string before concatenating it with the other strings, resulting in the string 'I have eaten 99 burritos.'.