**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:** expression: - \*, - , / , +

values 'hello', 6 -87.8

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

Answer:   
One key difference between a string and a variable is that a string is a fixed value, while the value stored in a variable can be changed.

For example, you could define a string with the value "Hello World" and it would always be "Hello World",

but you could define a variable and assign it the value "Hello World" at one point in your program, and then later assign it a different value, such as a number or a different string.

Another difference is that strings are immutable, which means that they cannot be modified once they have been created.

In contrast, the value stored in a variable can be changed as many times as necessary throughout the execution of a program.

Rules for declaring a variable in python

Variable names can only contain letters, digits, and underscores. They cannot start with a digit.

Variable names are case-sensitive. For example, x and X are different variables.

You cannot use any of the Python keywords as variable names.

It is a good practice to give variables descriptive names. This makes your code easier to read and understand

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

Answer:

The three different data types are

1) Integer: These are whole numbers that can be positive, negative, or zero. For example, 10, -5, and 0 are all integers. In Python, you can declare an integer using the int data type.

2) Floating point: These are numbers with a decimal point.

example, 3.14, -0.01, and 4.0 are all floating-point numbers.

In Python, you can declare a floating-point number using the float data type.

3 Strings: These are sequences of characters, such as words, phrases, or symbols.

Strings can be enclosed in single quotes (') or double quotes (").

For example, 'hello', "world", and '#' are all strings. In Python, you can declare a string using the str data type."""

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

Answer: An expression is a combination of values, variables, and operators that represents a single value.

In Python, an expression can be as simple as a single number or as complex as a series of calculations and function calls.

Here are some examples of expressions in Python:

10 + 5

x \* y

len(my\_list)

In the first example, 10 + 5 is an expression that adds the values 10 and 5 and produces the result 15.

In the second example, x \* y is an expression that multiplies the variables x and y and produces a result.

In the third example, len(my\_list) is an expression that calls the len() function and passes it the my\_list variable as an argument.

The function returns the length of the list, which is a single value.

All expressions in Python produce a single value as a result. This value can be used in further calculations or assigned to a variable for later use.""

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

**expression and a statement?**

Answer:

An expression is a combination of values, variables, and operators that represents a single value.

An expression can be as simple as a single number or as complex as a series of calculations and function calls.

A statement is a piece of code that performs a specific action.

In Python, there are several types of statements, including assignment statements, control statements, and function definitions.

An assignment statement is a type of statement that assigns a value to a variable. For example:

spam = 10

In this example, spam = 10 is an assignment statement that assigns the value 10 to the variable spam.

The main difference between an expression and a statement is that an expression produces a value,

while a statement performs an action. An expression can be used as part of a larger statement or as a standalone entity, while a statement cannot be used as an expression.

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

**bacon = 22**

**bacon + 1**

Answer:

After running the code, the variable bacon will still contain the value 22.

The expression bacon + 1 will evaluate to 23, but this value is not assigned to the bacon variable.

In order to update the value of the bacon variable, you would need to use an assignment statement. For example:

bacon = 22

bacon = bacon + 1

This would update the value of bacon to 23. Alternatively, you could use the shorthand assignment operator +=

bacon = 22

bacon += 1

This would also update the value of bacon to 23.

It's important to note that simply using an expression on its own will not change the value of a variable. You need to use an assignment statement to do that

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

**‘spam' + 'spamspam'**

**'spam'\*3""**

**Answer:**

The first expression, 'spam' + 'spamspam', will evaluate to the string 'spamspamspam'. This is because the + operator is used to concatenate (combine) two strings.

The second expression, 'spam' \* 3, will evaluate to the string 'spamspamspam'. This is because the \* operator is used to repeat a string a given number of times.

In both cases, the resulting value is a string containing the word "spam" repeated three times.

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

Answer

In Python, variable names can only contain letters, digits, and underscores. They cannot start with a digit.

This means that eggs is a valid variable name because it consists only of letters.

On the other hand, 100 is not a valid variable name because it starts with a digit.

The reason for this rule is that Python uses certain digits and symbols to represent special meanings (such as the equal sign (=) for assignment).

If variable names were allowed to start with digits, it could cause confusion when reading and writing code.

For example, the following are all valid variable names in Python:

eggs

my\_var

x1

\_private

The following are not valid variable names in Python:

100

1x

$cash

"""

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

**version of a value?**

Answer:

The three function are

Init()

Float()

Str()

By using these functions, you can easily convert a value to the desired data type in Python.

It's important to note that these functions can only be used to convert values to integers, floating-point numbers, or strings.

They cannot be used to convert a value to a different data type, such as a list or a dictionary.

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

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

Answer

expression 'I have eaten ' + 99 + ' burritos.' will cause an error because you cannot concatenate a string and an integer using the + operator.

In Python, the + operator is used to concatenate two strings, or to add two numbers. It cannot be used to combine a string and a number.

To fix the error, you can either convert the integer to a string using the str() function, or you can use a comma to separate the string and the integer, like this:

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

'I have eaten', 99, 'burritos.'

The first version uses the str() function to convert the integer to a string, so that it can be concatenated with the other strings.

The second version uses a comma to separate the string and the integer, which causes Python to automatically convert the integer to a string and insert a space between the values.

Either of these approaches will produce the correct result: 'I have eaten 99 burritos.'