**1 ans.**

In the given elements, the values are as follows:

* ‘hello’
* -87.8
* 6

Here, **‘hello’** is a string value, **-87.8** is a float value and **6** is an integer value.

On the other hand, expressions are:

* \* -> **Multiplication Operator**
* - -> **Subtraction Operator**
* / -> **Division Operator**
* + -> **Addition Operator**

**2ans.**

String: In programming, a string is a sequence of characters enclosed in quotation marks (either single or double quotes). It is a data type used to represent textual data. For example, 'hello' and "world" are both strings. Strings can contain letters, numbers, symbols, and spaces.

Variable: A variable is a named storage location in a program that holds a value. It is used to store and manipulate data during the execution of a program. Variables can hold different types of data, including strings, numbers, booleans, and more. Unlike a string, which represents a specific piece of textual data, a variable can hold different values of various types throughout the program's execution.

In simple terms, a string is a specific type of data representing textual information, while a variable is a container that can hold different types of data, including strings. Variables allow us to store, retrieve, and modify data dynamically in a program, while strings are one of the many possible types of data that can be stored in a variable.

**3ans.**

There are numerous data types in programming, they are:

1. Integer (int): An integer data type represents whole numbers without any fractional or decimal part. Integers can be positive or negative, including zero. They are typically used to represent counts, indices, and mathematical calculations that involve whole numbers. In many programming languages, integers have a fixed range of values they can represent, such as -2,147,483,648 to 2,147,483,647 for a 32-bit signed integer.
2. Floating-point (float): Floating-point data types are used to represent numbers with fractional parts. They are used to store real numbers, including numbers with decimal points. Floating-point numbers can be positive or negative and can have varying levels of precision. They are useful for representing measurements, calculations involving fractions, and any situation that requires precision beyond whole numbers. In many programming languages, float data types adhere to the IEEE 754 standard for floating-point representation.
3. String (str): A string data type represents a sequence of characters. It is used to store and manipulate textual data. Strings are typically enclosed in quotation marks (either single or double) to differentiate them from other data types. They can contain letters, numbers, symbols, and spaces. String data types are widely used for working with text, such as storing names, addresses, messages, and any other textual information in a program.

**4ans.**

An expression in programming is made up of one or more elements combined using operators, variables, constants, and functions.

Here are the components that make up an expression:

**Values**: These are the fundamental data elements, such as numbers (integers or floating-point), strings, booleans (true/false), or other data types.

**Variables**: These are named storage locations that hold values. Variables can be assigned values, and their values can be used within expressions.

**Operators**: These are symbols or keywords used to perform operations on values or variables. Examples of operators include arithmetic operators (+, -, \*, /), comparison operators (>, <, ==, !=), logical operators (&&, ||), and more.

**Function calls**: Functions are reusable blocks of code that perform specific tasks. They can take inputs (arguments) and return values. Function calls involve using the function name followed by parentheses, which may contain arguments passed to the function.

Expressions in programming are used to compute or derive values based on the combination of the above components. They can perform mathematical calculations, logical evaluations, string manipulations, and more. Expressions can be used in assignments, conditional statements, loops, and various other parts of a program.

**5ans.**

In programming, expressions and statements are two different constructs used to perform different tasks:

**Expression**: An expression is a combination of values, variables, operators, and function calls that evaluates to a single value. It produces a result by performing a computation or operation. Expressions can be as simple as a single value or more complex with multiple components. Examples of expressions include mathematical calculations, string concatenation, function calls that return a value, and more. Expressions can be used within statements to compute values or determine conditions.

**Statement**: A statement is a complete instruction or command that performs an action or controls the flow of a program. It represents an executable unit of code that carries out a specific task. Statements can include expressions, but they go beyond expressions by performing actions or operations that affect the program's state. Examples of statements include assignment statements (e.g., spam = 10), conditional statements (e.g., if-else statements), loop statements (e.g., for and while loops), function declarations, and more. Statements do not necessarily produce a value but rather perform an action or change the program's execution flow.

Summary, expressions are used to compute or evaluate values, while statements are used to perform actions, control program flow, and define the structure of a program. Expressions can be part of statements, such as using an expression to assign a value to a variable in an assignment statement.

**6ans.**

After running the given code, 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 line is an expression that adds 1 to the value of ‘bacon’ (which is 22). However, this expression is not assigned to any variable or used in any way. It is evaluated but the result is not stored or printed.

Therefore, the value of ‘bacon’ remains unchanged at 22. If we want to update the value of ‘bacon’ to the result of ‘bacon + 1’, we need to assign it back to bacon like this: ‘bacon’ = ‘bacon + 1’ or simply ‘bacon += 1’.

**7ans.**

The values of the two given terms would be as follows:

1. 'spam' + 'spamspam':

The term 'spam' + 'spamspam' performs string concatenation, which means joining two strings together. The resulting value would be 'spamspamspam'.

1. 'spam' \* 3:

The term 'spam' \* 3 performs string repetition or replication. It repeats the string 'spam' three times. The resulting value would be 'spamspamspam'.

So, the values of the two terms are:

* 'spam' + 'spamspam' = 'spamspamspam'
* 'spam' \* 3 = 'spamspamspam'

**8ans.**

**Validity of 'eggs'**:

Variable names in Python can contain letters (a-z, A-Z), digits (0-9), and underscores (\_).

They cannot start with a digit.

However, they can start with a letter or an underscore.

In the case of 'eggs', it starts with a letter ('e'), contains only letters, and follows the rules mentioned above. Hence, 'eggs' is a valid variable name.

**Invalidity of '100'**:

Variable names cannot start with a digit. They must start with a letter or an underscore.

'100' starts with a digit ('1'), which violates the rule mentioned above. Thus, '100' is considered an invalid variable name.

**9ans.**

The following three functions to convert a value to an integer, floating-point number, or string version:

1. int(): The int() function is used to convert a value to an integer. It can be used to convert a string representing an integer or a floating-point number to its integer form. It truncates any decimal portion and returns the whole number part. For example:

Python:

int('10') # #Output: 10

int(3.14) # Output: 3

1. float(): The float() function is used to convert a value to a floating-point number. It can be used to convert a string representing a number (integer or decimal) to its floating-point form. For example:

Python:

float('3.14') # Output: 3.14

float(10) # Output: 10.0

1. str(): The str() function is used to convert a value to a string. It can be used to convert any data type to its string representation. For example:

Python:

str(10) # Output: '10'

str(3.14) # Output: '3.14'

These functions are versatile and can be used to convert values between different data types in Python.

**10ans.**

The expression 'I have eaten ' + 99 + ' burritos.' causes an error because we are trying to concatenate a string ('I have eaten ') with an integer (99) directly using the '+' operator. In Python, concatenation with the '+' operator can only be performed between two strings.

To fix the error and successfully concatenate the string and the integer, we need to convert the integer to a string before performing the concatenation. We can achieve this by using the ‘str()’ function to convert the integer to its string representation. Here's the fixed expression:

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

By using str(99), the integer value of 99 is converted to the string '99'.