***ASSIGNMENT\_1***

**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

***ANS:***

Values: (‘hello’, ‘-87.8’, ‘6’)

Expressions: (\*, -, /, +)

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

***ANS:***

|  |  |
| --- | --- |
| **STRING** | **VARIABLE** |
| Is a sequence of characters enclosed in either in single quotes (‘’) or double quotes (“”).it represents textual data and can include letters, numbers,symbols,and, whitespace.  strings are immutable, cannot be changed once they created.  can perform various operations on strings, such as concatenation, slicing, and formatting. | A variable is a name that refers to a value or an object. It acts as a container for storing data of any type, including strings.  Variables are dynamically typed, no need to specify the type explicitly.  Variables can be reassigned to new values and their values can be changed during the execution of a program. |

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

***ANS:***

**Integer (int):** The integer data type represents whole numbers without any fractional or decimal parts.

Integers can be positive or negative. They are used to perform mathematical operations such as addition, subtraction, multiplication, and division.

Integer can represent very large or small numbers.

Eg: **x = 5**

**Y=-10**

**String (str):** The string data type represents a sequence of characters. It is used to store and manipulate textual data.

Strings are enclosed in either single quotes (' ') or double quotes (" "). They can contain letters, numbers, symbols, and whitespace.

Strings are immutable, meaning their values cannot be changed once they are created. However, we can perform various operations on strings, such as concatenation, slicing, and formatting**.**

Eg : **message = “hello world”**

**Boolean (bool**): The Boolean data type represents a logical value that can be either True or False.

Booleans are used in conditional statements and logical operations. They are often the result of comparisons or logical operations.

True and False are the only valid Boolean values in Python. Booleans are helpful for decision-making in programs.

Eg: **it\_is\_dark = True**

**It\_is\_white = False**

**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 can also include function calls and other constructs.

**Operand**: An operand represents a value or a variable. It can be a literal value, such as a number or a string, or it can be a variable that holds a value.

For example, in the expression **2 + x**, **2** and **x** are operands.

**Operator:** An operator performs an operation on one or more operands to produce a result. Operators can be mathematical **(e.g., +, -, \*, /),** logical (e.g., **and, or, not**), or relational (**e.g., >, <, ==).** Operators define the type of operation to be performed.

For example, in the expression **2 + x, +** is the operator.

**Function Calls:** Expressions can include function calls, where a function is invoked with arguments to perform a specific task or calculation. Functions can return a value that becomes part of the expression.

For example, in the expression **math. sqrt (16), math. sqrt ()** is a function call.

**5. What is the difference between an expression and a statement?**

***ANS:***

**Expression:** An expression is a combination of values, variables, operators, and/or function calls that produces a single value when evaluated.

Expressions can be as simple as a single constant value or as complex as a combination of multiple operations.

They can be used within statements or other expressions to compute a result.

**Eg: 6\*9**

**X\*y**

**Math.sqrt(15)**

**Statement:** A statement is a complete instruction or action that performs a specific task. It typically consists of keywords, expressions, and other programming constructs.

Statements are used to control the flow of a program, define actions, or change the state of variables. Unlike expressions, statements do not produce a value when evaluated**.**

**Eg:** **spam = 10**

**if x > 5:**

**print("x is greater than 5")**

**for i in range(5):**

**print(i)**

In the assignment statement spam = 10, it is a statement that assigns the value 10 to the variable spam. The statement doesn't produce a value but changes the state of the variable.

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

**bacon = 22**

**bacon + 1**

***ANS:***

bacon = 22

bacon+1

The variable bacon will still contain the value 22.

Here's why:

In the first line, the value 22 is assigned to the variable bacon.

In the second line, bacon + 1 is an expression that evaluates to 23. However, this expression is not assigned to any variable or used in any way, so the result is not stored or updated in the variable bacon.

In Python, an expression like bacon + 1 by itself does not modify the value of bacon unless you explicitly assign the result back to the variable, like bacon = bacon + 1 or bacon += 1.

So, the value of bacon remains unchanged at 22.

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

**'spam' + 'spamspam'**

**'spam' \* 3**

***ANS:***

spamspamspam

spamspamspam

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

***ANS:***

* eggs starts with a letter ('e'), which satisfies the requirement for the first character of a variable name.

It doesn't contain any special characters or spaces.

Thus, eggs adheres to the valid variable name rules and is considered valid**.**

* 100 starts with a digit ('1'), which violates the rule that a variable name must start with a letter or an underscore.

Variable names cannot begin with a digit in Python.

Therefore, 100 is not a valid variable name.

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

***ANS:***

**int():** This function can be used to convert a value to an integer. It takes a numeric string or a floating-point number and returns the corresponding integer value. If the value cannot be converted to an integer, it will raise a ValueError.

**Eg: value = "68"**

**integer value = int(value)**

**print(integer\_value) #output = 68**

**float ():** This function is used to convert a value to a floating-point number. It takes a numeric string or an integer and returns the corresponding floating-point value. If the value cannot be converted to a float, it will raise a ValueError.

**Eg: value = "3.14"**

**float\_value = float(value)**

**print(float\_value) #output = 3.14**

**str():** This function converts a value to its string representation. It takes any value, such as an integer, float, or object, and returns its string representation.

Eg:

**value = 42**

**string\_value = str(value)**

**print(string\_value) # Output: "42"**

**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 attempts to concatenate a string ('I have eaten ') with an integer (99) directly, without converting the integer to a string.

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

Here's the corrected expression:

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