**Python 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.

\* - expression

‘Hello’ - value

-87.8 - value

- - expression

/ - expression

+ - expression

6 - value

2. What is the difference between string and variable?

String:

A string is a data type used to represent text or sequence of characters. It is enclosed within single (' ') or double (" ") quotes. For example:

Here, "Hello, World!" is a string.

Variable:

A variable is a symbolic name that represents a value in memory. It is used to store data, including strings, numbers, and other types. Variables are essential for storing and manipulating data in a program. For example:

my\_variable = 42

Here, my\_variable is a variable that stores the value 42.

a string is a specific type of data (text) that can be stored in a variable, while a variable is a named container that holds a value, which can be a string or any other data type.

3. Describe three different data types.

Here are descriptions of three different data types commonly used in programming:

Integer (int):

An integer data type represents whole numbers, both positive and negative, without any fractional or decimal part. Integers are used for counting, indexing, and performing arithmetic operations. In Python, you can declare an integer using a simple assignment:

age = 25

Float (float):

A float data type represents numbers with a fractional or decimal part. Floats are used to represent real numbers and are crucial for calculations involving precision. They are declared using a decimal point or scientific notation:

pi = 3.14159 , distance = 2.5e-3 # Scientific notation: 2.5 \* 10^-3

String (str):

A string data type represents a sequence of characters, typically used to represent text. Strings are enclosed within single (' ') or double (" ") quotes. They are versatile and used for various purposes, including text manipulation, formatting, and displaying information:

name = "Alice"

message = 'Hello, ' + name + '!'

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

An expression in programming is a combination of values, variables, operators, and function calls that can be evaluated to produce a result. Expressions are a fundamental concept in programming and are used to perform various computations and operations. They can represent calculations, comparisons, and other actions.

An expression is made up of the following components:

Values: These are constants like numbers or strings.

Variables: These are symbols that represent values stored in memory.

Operators: These are symbols that perform operations on values or variables. Examples include addition (+), subtraction (-), multiplication (\*), division (/), etc.

Function Calls: These are invocations of functions that may take some input values (arguments) and produce an output value.

Expressions can vary in complexity from simple arithmetic calculations to more intricate combinations involving multiple operators and function calls. The primary purpose of expressions is to compute or derive a value.

All expressions, when evaluated, produce a value. The value can be of various types, such as numbers, strings, booleans, or even more complex data structures. Expressions are used extensively in programming to manipulate data, make decisions, control program flow, and perform various operations. Here are a few examples of expressions:

# Arithmetic expression

result = 2 \* (3 + 5)

# String concatenation expression

greeting = "Hello, " + "world!"

# Comparison expression

is\_equal = (x == y)

# Function call expression

length = len("example")

# Logical expression

is\_valid = (x > 0) and (y < 10)

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

expression and a statement?

The distinction between expressions and statements is a fundamental concept in programming languages. Let's break down the differences between the two:

Expression:

An expression is a combination of values, variables, operators, and function calls that can be evaluated to produce a value. It always results in a value and can be used as part of a larger expression or assigned to a variable. Expressions can be simple, like a single value or variable, or complex, involving multiple operations. Expressions are used to compute values.

Examples of expressions:

2 + 3

x \* 5

"Hello, " + name

len("example")

Statement:

A statement is a complete line of code that performs an action. It can be a command to do something or a directive that changes the program's state. Statements don't necessarily produce a value. They can include variable assignments, control flow structures (like if statements or loops), function definitions, and more. Statements are used to control the program's behavior and structure.

Examples of statements:

spam = 10 # Assignment statement

if x > 0: # If statement

print("Positive")

while i < 10: # While loop statement

print(i)

i += 1

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

bacon = 22

bacon + 1

Ans : 22

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

'spam' + 'spamspam' o/p - spamspamspam

'spam' \* 3 o/p - spamspamspam

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

In most programming languages, including Python, variable names must adhere to certain rules and conventions. These rules are in place to ensure clarity, consistency, and to prevent ambiguity. Let's address the specific examples you provided:

eggs:

The variable name "eggs" is considered valid because it follows the rules for naming variables. In Python, variable names must start with a letter (a-z, A-Z) or an underscore (\_), followed by letters, underscores, or digits (0-9). So, "eggs" starts with a letter ('e') and consists only of letters, which makes it a valid variable name.

100:

The variable name "100" is considered invalid because it does not follow the rules for naming variables. In Python and many other programming languages, variable names cannot start with a digit. They must start with a letter or an underscore. Since "100" starts with a digit, it is not a valid variable name.

Here are a few additional points to consider:

Variable names are case-sensitive. For example, "eggs" and "Eggs" would be treated as two different variables.

It's a good practice to use descriptive names for variables that indicate their purpose or content. This helps improve code readability and maintainability.

Certain words, called keywords, are reserved for specific purposes in the language and cannot be used as variable names. For example, you cannot use "if," "while," "for," or "print" as variable names.

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

version of a value?

In Python, you can use the following three built-in functions to convert values to different types:

int(): This function is used to convert a value to an integer. It can take various inputs, including other integers, floating-point numbers, or even strings that represent valid integers. For example:

x = int(5.7) # Converts the float 5.7 to the integer 5

y = int("10") # Converts the string "10" to the integer 10

float(): This function converts a value to a floating-point number. Like int(), it can handle integers, floating-point numbers, and strings representing valid numeric values. For example:

a = float(7) # Converts the integer 7 to the float 7.0

b = float("3.5") # Converts the string "3.5" to the float 3.5

str(): The str() function is used to convert a value to its string representation. This is often useful when you want to concatenate values with strings or display them as text. For example:

value = 42

text = "The value is: " + str(value) # Converts the integer 42 to the string "42"

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

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

The expression 'I have eaten' + 99 + 'burritos.' causes an error because we are trying to concatenate a string with an integer directly, and Python does not implicitly convert the integer to a string in this context.

To fix this error, we need to explicitly convert the integer 99 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.'