**FSDS MAY BATCH 2022(Python Basics -1)**

**Submitted by: Shubham Tiwari**

Q1: 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:

\* is **not a valid value or expression.**

'hello' is a **string value**.

-87.8 is a **floating-point value**.

- is a **mathematical operator**.

/ is a **mathematical operator**.

is a **mathematical operator.**

6 is an **integer value**.

Q2. What is the difference between string and variable?

Ans: A string is a data type used to represent text, whereas a variable is a named storage location in a program's memory used to store a value that can be changed during the execution of the program. A string is a specific value, while a variable can hold different values at different times during the execution of the program. For example, the string "Shubham" has a specific value, whereas a variable named "message" could store the value "Shubham" at one point in time and a different value, such as "Goodbye", at another point in time.

Q3. Describe three different data types.

Ans: In Python, there are three main data types:

1. Numeric: Used to store numerical values, including integers (e.g. -1, 0, 1), floating-point numbers (e.g. 1.0, -0.5), and complex numbers.
2. String: Used to store textual information, enclosed in either single or double quotes (e.g. "hello" or 'world').
3. Sequence: Used to store collections of data, including lists (e.g. [1, 2, 3]), tuples (e.g. (1, 2, 3)), and range objects.

Q4. What is an expression made up of? What do all expressions do?

Ans: A expression is made up of values, operators, and/or variables that evaluate to a single value. All expressions in Python return a value, which can be stored in a variable or used as part of another expression. For example, the expression **2 + 3** evaluates to **5**, which can be stored in a variable like this: **result = 2 + 3**. The type of value returned by an expression depends on the types of the values and operators used in the expression. For example, the expression **2 + 3** returns an integer value, while the expression **'hello' + 'world'** returns a string value.

Q5. This assignment statements, like spam = 10. What is the difference between an expression and a statement?

Ans: A statement is a unit of code that the Python interpreter can execute. It performs an action and may produce an output, but does not itself have a value. An expression is a piece of code that can be evaluated to produce a value.

In the example given, **spam = 10** is a statement, because it performs an action (assigning a value to a variable). **10** is an expression, because it can be evaluated to produce a value (the number 10).

So, the difference between an expression and a statement is that an expression has a value, while a statement performs an action.

Q6. After running the following code, what does the variable bacon contain?

bacon = 22

bacon + 1

Ans: After running the code, the variable **bacon** contains 22. The expression **bacon + 1** returns a value of 23, but does not modify the value stored in the **bacon** variable. To update the value of **bacon**, we would need to assign the result of the expression back to the variable: **bacon = bacon + 1**.

Q7. What should the values of the following two terms be?

‘spam’ + ‘spamspam’;

‘spam’ \* 3

Ans**: spam' + 'spamspam' = 'spamspamspam'**

**'spam' \* 3 = 'spamspamspam'**

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

Ans: In Python, variable names can start with an alphabet or underscore, but not with a number. So, "eggs" is a valid variable name because it starts with an alphabet, while "100" is invalid because it starts with a number.

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

Ans: The three functions in Python to get the integer, floating-point number, or string version of a value are:

**int(value)** - returns the integer version of a value.

**float(value)** - returns the floating-point number version of a value.

**str(value)** - returns the string version of a value.

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

‘I have eaten’ + 99 + ‘burritos’

Ans: **This expression causes an error because the number 99 is not within quotes and therefore not interpreted as a string.** To fix it, we can convert the number 99 to a string or add quotes around it: 'I have eaten' + str(99) + ' burritos' or 'I have eaten 99 burritos'.