Assignment 1

Question 1:-

In the below elements which of them are values or an expression? eg:- values can be integers or strings and expressions will be mathematical operators.

*****
'hello'
-87.8
-
/
+
6

Ans:-

* = Expression
'hello' = Values
-87.8 = Values
\- = Expression
/ = Expression
\+ = Expression
6 = Values

Question 2:-

What is the difference between string and variable?

Ans:-

Strings	Variables
A ‘string’ is simply a list of characters in order.	A variable is something that holds a value that may change.
In simplest terms, a variable is just a box in that you can put stuff.	A character is anything you can type on the keyboard in one keystroke, like a letter, a number, or a backslash.
Strings are data, so we can use them to fill up a variable.	Variables are symbols that you can use to store data in a program.
Types of string 1. Single line string 2. Multiline String	In Python variables is of four different types: 1. Integer 2. Long Integer 3. Float 4. String
Single-line strings are enclosed in single or double quotes and terminate in one line. Multiline strings store multiple lines of text and are enclosed in triple quotes.	Independent variables are the input for a process that is being analyzed.
Examples:- “hello world”, “LKJH019283”, etc…	Examples:- Height, age, income, province or country of birth, grades obtained at school, and type of housing, etc…

Question 3:-

Describe three different data types.

Ans:-

Data types are the classification or categorization of data items. It represents the kind of value that tells what operations can be performed on a particular data.

1. **Numeric:-**

In Python, numeric data type represents the data that has a numeric value. Numeric values can be integers, floating numbers, or even complex numbers. These values are defined as int, float, and complex classes in Python.

- **Integer -** This value is represented by the int class. It contains positive or negative whole numbers (without fractions or decimals). In python, there is no limit to how long an integer value can be.
  For example :- a = 5.

- **Float -** This value is represented by the float class. It is a real number with a floating-point representation. It is specified by a decimal point. Optionally, the character e or E followed by a positive or negative integer may be appended to specify scientific notation.
  For example :- b = 5.0

- **Complex Numbers -** Complex number is represented by complex class. It is specified as (real part) + (imaginary part) j .
  For example - 2 + 3j .

2. **Sequence Type:-**

In Python, the sequence is the ordered collection of similar or different data types. Sequences allow you to store multiple values in an organized and efficient fashion. There are several sequence types in Python —

- **String:-** In Python, Strings are arrays of bytes representing Unicode characters. A string is a collection of one or more characters put in a single quote, double-quote, or triple-quote. In python there is no character data type, a character is a string of length one. It is represented by the “str” class.

  Creating String:- Strings in Python can be created using single quotes or double quotes or even triple quotes.

For Example:-
String = “Welcome to the team”
print(string)

Output:- Welcome to the team

- **List:-** Lists are just like the arrays, declared in other languages which is an ordered collection of data. It is very flexible as the items in a list do not need to be of the same type.

  Creating List:-
  Lists in Python can be created by just placing the sequence inside the square brackets [ ].

  For example:-
  List = [“Hii”, “I”, “am”, “Pooja”, 007]
  print(List)
  print(List[0])
  print(List[1])
  print(List[2])
  print(List[3])
  print(List[4])

  Output:-
  [‘Hii’, ‘i’, ‘am’, ‘Pooja’, 007]
  Hii
  I
  am
  Pooja
  007

- **Tuple:-** Just like a list, a tuple is also an ordered collection of Python objects. The only difference between a tuple and a list is that tuples are immutable i.e. tuples cannot be modified after it is created. It is represented by a tuple class.

  Creating Tuple:-
  In Python, tuples are created by placing a sequence of values separated by ‘comma’ with or without the use of parentheses for grouping the data sequence. Tuples can contain any number of elements and of any datatype (like strings, integers, lists, etc.).

  For Examples:-
  tuple = (‘Pooja’, ‘Rahamatkar’, 45, 56, 85)
  print(tuple)

Output:-
('Pooja', 'Rahamatkar', 45, 56, 85)

3. **Boolean:-**

Data type with one of the two built-in values, True or False. Boolean objects that are equal to True are truthy (true), and those equal to False are falsy (false). But non-Boolean objects can be evaluated in a Boolean context as well and determined to be true or false. It is denoted by the class bool.

Note - True and False with capital 'T' and 'F' are valid booleans otherwise python will throw an error.

For Example:-

```
print(type(True))
print(type(False))
```

Output:-

```
<class 'bool'>
<class 'boo'>
```

Question 4 :-

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

Ans:-

An expression is a combination of values and operators. All expressions evaluate (that is, reduce) to a single value.

An expression is a combination of operators and operands that is interpreted to produce some other value. In any programming language, an expression is evaluated as per the precedence of its operators. So that if there is more than one operator in an expression, their precedence decides which operation will be performed first. We have many different types of expressions in Python.

An expression is a construct made up of variables, operators, and method invocations, which are constructed according to the syntax of the language, and that evaluate to a single value. You're already seen examples of expressions, illustrated in bold below:

For Examples:-
X = 5
X < 10
Output:- True

X = 8
X < 10 and callable(x)
Output:- True

Expressions are representations of value. They are different from statements in the fact that statements do something while expressions are a representation of value. For example, any string is also an expression since it represents the value of the string as well.

Python has some advanced constructs through which you can represent values and hence these constructs are also called expressions.

Creating an Expressions:-

Python expressions only contain identifiers, literals, and operators. So, What are these?

Identifiers:- Any name that is used to define a class, function, variable module, or object is an identifier.

Literals:- These are language-independent terms in Python and should exist independently in any programming language. In Python, there are string literals, byte literals, integer literals, floating point literals, and imaginary literals.

Operators:- In Python you can implement the following operations using the corresponding tokens.

Operator	Token
add	+
subtract	-
multiply	*
power	**
Integer Division	/
remainder	%
decorator	@
Binary Left Shift	<<
Binary right shift	>>
and	&
ir	\
Binary Xor	^
Binary ones complement	~
Less than	<
Greater than	>
Less than or equal to	<=
Greater than or equal to	>=
Check equality	==
Check not equal	!=

Question 5:-

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

Ans:-

An expression evaluates to a single value. A statement does not.

Expression	Statement
An expression evaluates to a value	A statement executes something
The evaluation of a statement does not change the state	The execution of a statement changes the state
Evaluation of an expression always Produces or returns a result value.	Execution of a statement may or may not produce or display a result value; it only does whatever the statement says.
Every expression can't be a statement	Every statement can be an expression
Example:- a + 16 20	Example:- x = 3 print(x) Output:- 3

Question 6:-

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

```
bacon = 22
bacon + 1
```

Ans :-

23

The bacon variable is set to 20. The bacon + 1 expression does not reassign the value in bacon (that would need an assignment statement: bacon = bacon + 1)

Question 7:-

What should the values of the following two terms be?

Ans:-

Both expressions evaluate the string
spamspamspam

Question 8:-

Why are eggs a valid variable name while 100 is invalid?

Ans:-

Variable names cannot begin with a number.
And eggs are the string/character.

Question 9:-

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

Ans:-

The int(), float(), and str() functions will evaluate the integer, floating-point number, and string versions of the value passed to them.

Question 10:-

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

‘I have eaten’ + 99 + ‘burritos’.

Ans :-

Error :-

```
File "<ipython-input-1-d3928340a68a>", line 1
    ‘I have eaten’ + 99 + ‘burritos’
     ^
SyntaxError: invalid character in identifier
```

Stack Overflow

Solution :- `'I have eaten' + str(99) + 'burritos'`

Output :- I have eaten99burritos

The expression causes an error because 99 is an integer, and only strings can be concatenated to other strings with the + operator. The correct way is ‘I have eaten’ + str(99) + ‘burritos’.