ASSIGNMENT 1

WEEK 1 & WEEK 2

NAME: Sanjitha P Naik

COURSE NAME: Python for Data Science

SRN: PES2UG22EC117

SEM: 7 B

MCQ BASED QUESTIONS:

1. Which of the following options does not require to process the raw data?
   a) Inspecting
   b) Transforming
   c) Modeling
   d) Interpreting
   e) Storing

ANS: e) Storing.

2. The method to clear all the element from a set is
   a) remove()
   b) discard()
   c) clear()
   d) delete()

ANS: c) clear()

3. Variable m is defined as “MooN”, command to convert ‘m’ as ‘moon’ is
   a) m.upper()
   b) m.lower()
   c) m.string()
   d) m.title()

ANS: b) m.lower

4. Assignment operator used in Python is:
   a) ==
   b) =
   c) <=
   d) >>

Ans: b) =

Descriptive Based Questions:

1. Explain the difference between the assignment operator = and the comparison operator == in Python with the help of an example. Why is it important not to confuse them while writing programs?

ANS: In Python, the ‘*assignment operator*’ **=** is used to assign a value to a variable, while the ‘*comparison operator’* **==** is used to check whether two values are equal.

For example:

```
x = 10

y = 10

print(x == y)   # comparison: checks if x and y are equal
```

expected output: True

Example-2: (For assignment operator)

```
x = 50

print("Value of x:", x)

y = x + 25

print("Value of y:", y)

name = "NPTEL"

print("Course Name:", name)

x = 100   # previous value (50) is overwritten

print("New value of x:", x)
```

OUTPUT:

```
 Value of x: 50

Value of y: 75

Course Name: NPTEL

New value of x: 100
```

It is important to not to confuse between these two as using == instead of = ; variable will never be updated, and the program logic may fail. Similarly, using = instead of == inside a condition will throw an error.

2. In Python, both strings and lists are sequence data types, but their methods behave differently because strings are immutable while lists are mutable. Explain this difference in detail with appropriate examples. Write code snippets to demonstrate:

a) An operation that works differently on a string and a list (e.g., append/concatenation).

b) How slicing can be used to overcome immutability in strings.

ANS: In Python, strings and lists are both sequence types, and they support indexing, slicing, and iteration. However, the key difference is that strings are immutable, while lists are mutable.

Mutable (List): We can directly modify elements.

Immutable (String): Any operation creates a new string; the original cannot be changed.

Example-a:

```
# list is mutable

numbers = [1, 2, 3]

numbers.append(4)

print(numbers)
```

Output: [1, 2, 3, 4]

```
# String is immutable

text = "NPTEL"

text.append("X")
```

Output: Error: 'str' object has no attribute 'append'

Example-b:

```
text = "Data"

new_text = "M" + text[1:]    # Reconstruct a new string

print(new_text)
```

Output: “Mata”