

## Report

### String Manipulation

This report provides an overview of a Python program designed for string manipulation. The program takes a user input string and performs various operations to demonstrate fundamental text processing capabilities. These operations include calculating the length of the string, converting it to uppercase and lowercase, checking if it contains only alphabetic characters, replacing a specific substring, and splitting the string into a list of words.

The program begins by soliciting user input. Users are prompted to enter a string of their choice.

- **Operation 1: Length of the String:** The program calculates and prints the length of the input string using the ``len()`` function. This operation provides a simple but essential metric for understanding the size of the text.
- **Operation 2: Uppercase and Lowercase Conversion:** Next, the program converts the input string to both uppercase and lowercase using the ``upper()`` and ``lower()`` string methods, respectively. It then prints both versions. These operations are valuable for standardizing text or performing case-insensitive comparisons.
- **Operation 3: Alphabetic Character Check:** The program checks if the input string contains only alphabetic characters using the ``isalpha()`` method. It prints the result, indicating whether the string consists solely of letters. This operation is useful for validating input data for specific text-based applications.
- **Operation 4: Substring Replacement:** Users are prompted to enter a specific substring they wish to replace and the replacement substring. The program utilizes the ``replace()`` method to substitute the old substring with the new one. It then prints the modified string. This operation is beneficial for text editing and manipulation tasks.
- **Operation 5: Splitting into Words:** Finally, the program splits the input string into a list of words using the ``split()`` method. It defaults to splitting on whitespace but can be customized to use a different delimiter. The list of words is then printed, which is valuable for text analysis, tokenization, and parsing.

In conclusion, the Python program presented in this report offers a practical set of text manipulation operations. It enables users to interactively explore and modify textual data, making it a valuable tool for a wide range of applications in data science, natural language processing, and general text processing tasks. Its ease of use and clear outputs make it accessible to users with varying levels of programming experience.

The screenshot displays a Visual Studio Code (VS Code) editor window titled "Python Labs". The Explorer sidebar on the left shows a project structure with folders for ".venv", "Module-1", "Module-2", "Lab-1", "Lab-2", "Lab-3", "Task-1", "Assignment-1", "Assignment-2", "Report-InputCat...", "Screenshot-Inp...", "Task-1\_InputCat...", "Task-2", and "Lab-4". The file "Assignment-1\_StringManipulation.py" is open in the editor.

The code in the editor is as follows:

```

1  ## Student Name: Harsh Siddhapura
2  ## Student ID: 1230169813
3  ## Date: 09/01/2023
4
5  # Input from the user
6  user_input = input("Enter a string: ")
7
8  # Print the length of the string
9  length = len(user_input)
10 print("Length of the string:", length)
11
12 # Convert the string to uppercase and lowercase
13 uppercase = user_input.upper()
14 lowercase = user_input.lower()
15 print("Uppercase:", uppercase)
16 print("Lowercase:", lowercase)
17
18 # Check if the string contains only alphabetic characters
19 is_alphabetic = user_input.isalpha()
20 print("Contains only alphabetic characters:", is_alphabetic)
21
22 # Replace a specific substring with another substring
23 old_substring = input("Enter the substring to replace: ")
24 new_substring = input("Enter the replacement substring: ")
25 replaced_string = user_input.replace(old_substring, new_substring)
26 print("String after replacement:", replaced_string)
27
28 # Split the string into a list of words
29 words = user_input.split()
30 print("List of words:", words)

```

The TERMINAL panel at the bottom shows the execution of the script. It displays three separate runs of the program with user input and the corresponding output:

```

(.venv) harshsiddhapura@Harshs-MacBook-Air Assignment-1 % python3 Assignment-1_StringManipulation.py
Enter a string: abc DEF ghi 123
Length of the string: 15
Uppercase: ABC DEF GHI 123
Lowercase: abc def ghi 123
Contains only alphabetic characters: False
Enter the substring to replace: ghi
Enter the replacement substring: jkl
String after replacement: abc DEF jkl 123
List of words: ['abc', 'DEF', 'ghi', '123']

(.venv) harshsiddhapura@Harshs-MacBook-Air Assignment-1 % python3 Assignment-1_StringManipulation.py
Enter a string: def xYz
Length of the string: 7
Uppercase: DEF XYZ
Lowercase: def xyz
Contains only alphabetic characters: False
Enter the substring to replace: def
Enter the replacement substring: abc
String after replacement: abc xYz
List of words: ['def', 'xYz']

(.venv) harshsiddhapura@Harshs-MacBook-Air Assignment-1 % python3 Assignment-1_StringManipulation.py
Enter a string: 123 jkl *()
Length of the string: 11
Uppercase: 123 JKL *()
Lowercase: 123 jkl *()
Contains only alphabetic characters: False
Enter the substring to replace: *()
Enter the replacement substring: +)
String after replacement: 123 jkl +)

```

The status bar at the bottom indicates the current file is "main\*", the cursor is at "Ln 30, Col 31", and the editor is using "Spaces: 4", "UTF-8", "LF" encoding, "Python" language, and "3.9.6 (.venv: venv)" interpreter. The "Prettier" formatter is also active.