I. Introduction

1. Purpose

This document specifies the requirements for a software application designed to process strings in a C# Windows-based application. The objective is to transform input strings, encode characters based on their ASCII values, and provide functionalities for sorting and retrieving character codes.

2. Scope

The project will deliver a winform application that operates efficiently within a Windows environment. It will cater to individual users needing string manipulation and encoding functionality, ensuring input validation and security.

II. Description

1. Product

The product will include functionalities to encode input strings based on ASCII value shifts, sort strings alphabetically, and retrieve ASCII codes of input and output strings. The software will ensure accurate data validation and encapsulation.

2. Users

The primary users are developers, students, and individuals needing quick and reliable string manipulation tools.

3. Operational Environment

The application is designed to run on any standard Windows-based C# compiler with .NET Framework or later versions.

III. System Features

1. Input Validation and Data Handling

The system validates input strings to ensure they contain only uppercase letters and limits length to 40 characters. Integer inputs are constrained to the range [-25, 25].

The system needs to:

- Prompt users for correct input if validation fails.
- Display error messages for invalid inputs.

2. Encoding Strings Description

This feature shifts each character in the input string based on a user-provided integer value, wrapping around the alphabet if required.

The system needs to:

- Replace characters accurately based on the ASCII value shift.
- Output a correctly encoded string.

3. Sorting Strings Alphabetically

The system sorts the input string alphabetically.

The system needs to:

- Generate a new, sorted string based on the original input string.

4. ASCII Code Retrieval

Provides arrays of ASCII values for characters in input and output strings.

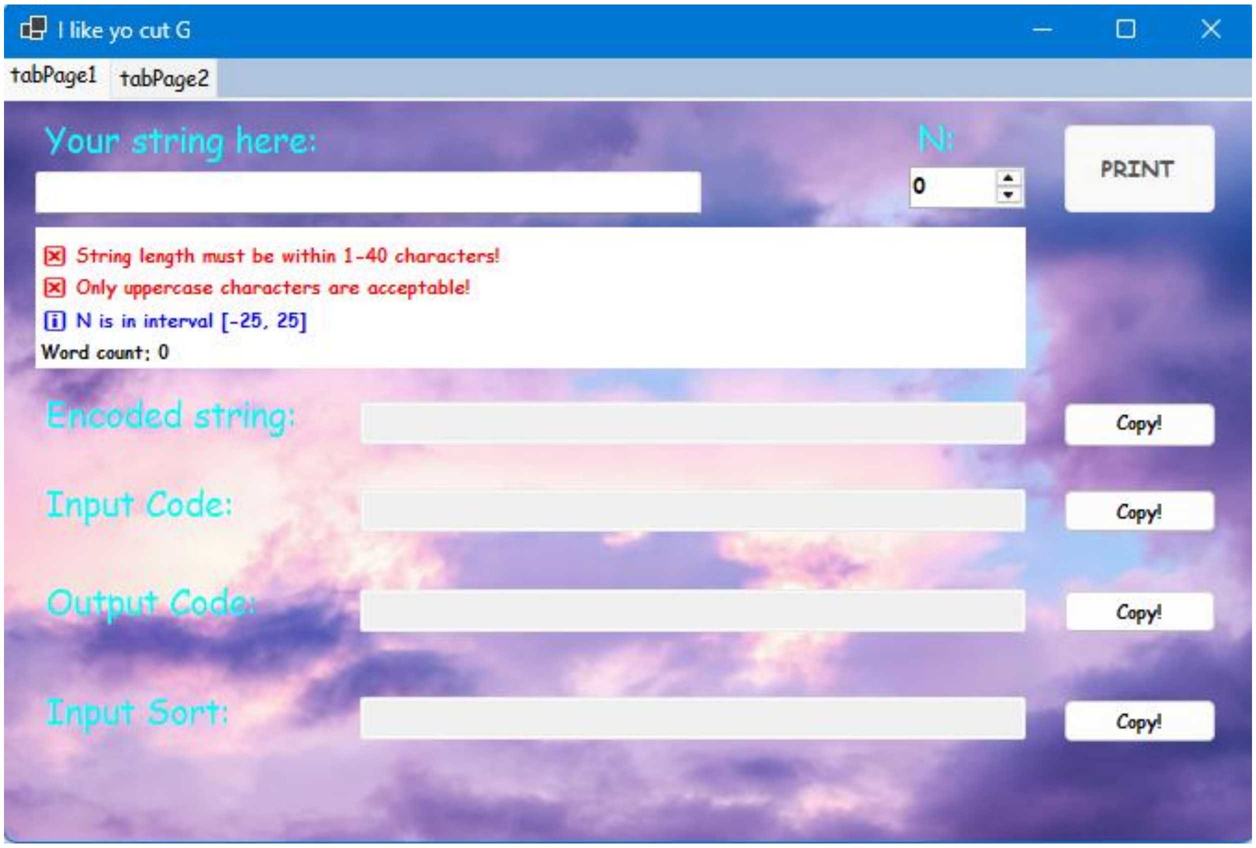
The system needs to:

- Output ASCII values for input and encoded strings as integer arrays.

IV. Demonstration

This section highlights screenshots and code snippets for each feature, providing visual and technical evidence of program execution, functionality, and underlying implementation logic.

1. Program User Interface



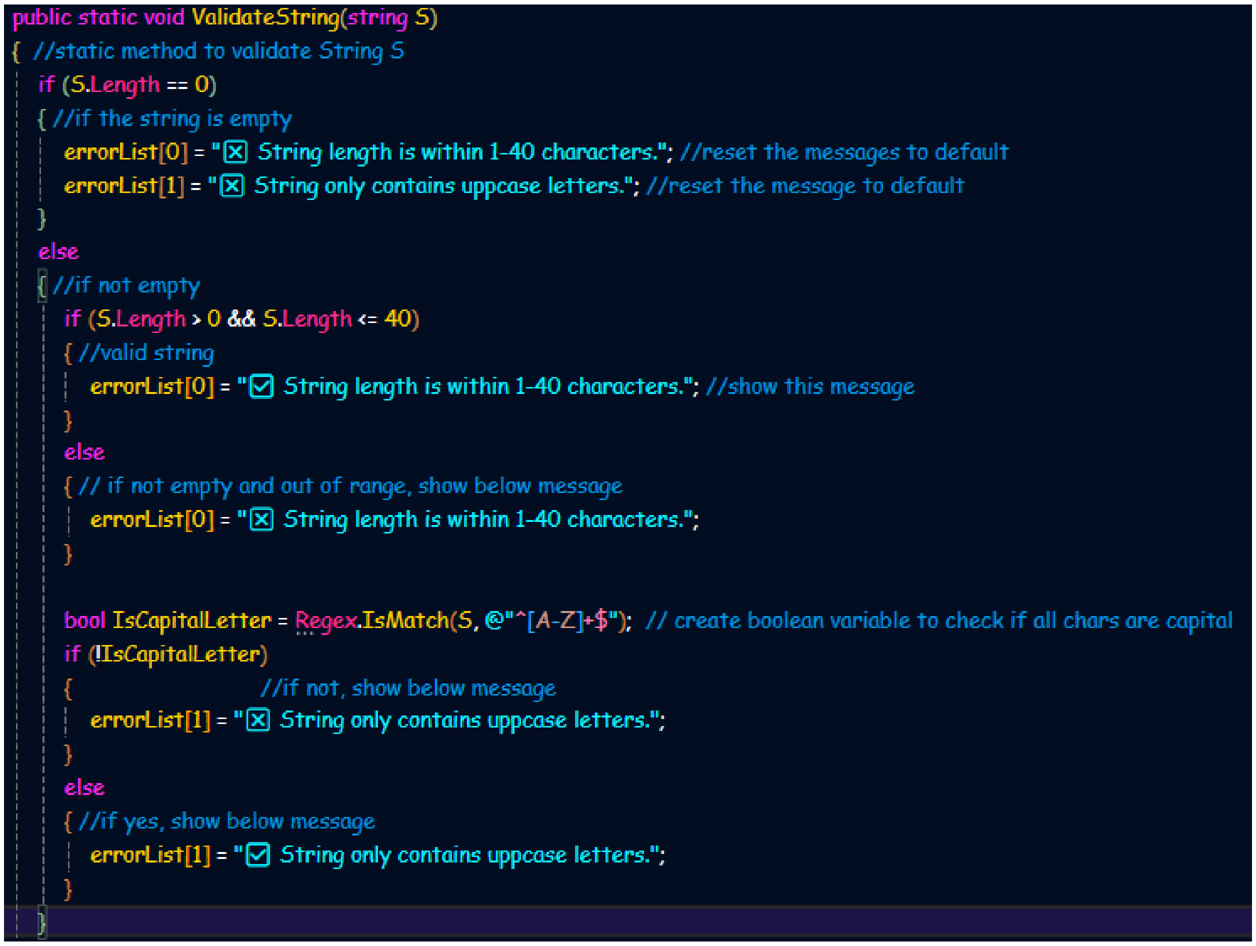
String length must be within 1-40 characters!
Only uppercase characters are acceptable!
N is in interval [-25, 25]
Word count: 0

Figure 1 – The user interface of the program

The "String Encoder App" features a user-friendly interface with input fields for strings and numeric shifts, validation guidelines, and output displays for encoded strings, ASCII codes, and sorted strings. Its design ensures clear instructions and efficient functionality, with labeled sections and intuitive controls, offering a seamless experience for string manipulation tasks.

2. Validation

a) Code snippet



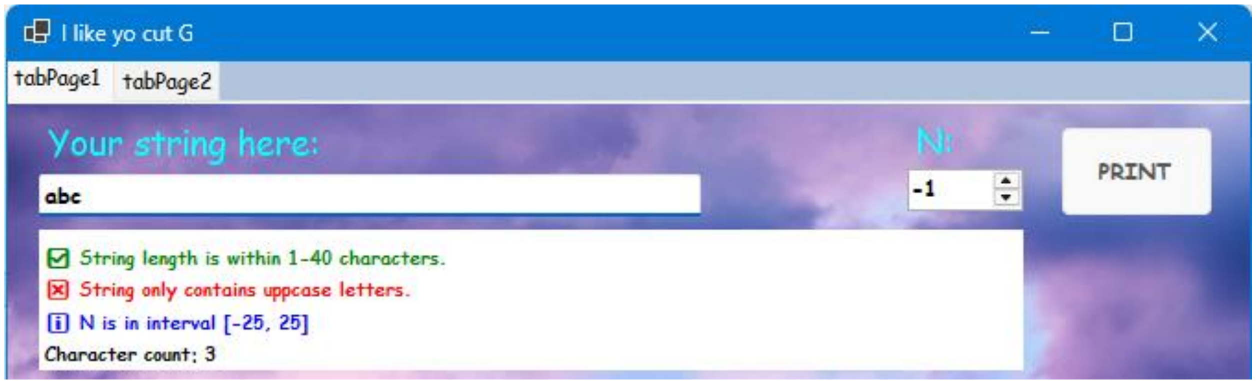
```
public static void ValidateString(string S)
{ //static method to validate String S
    if (S.Length == 0)
    { //if the string is empty
        errorList[0] = "☒ String length is within 1-40 characters."; //reset the messages to default
        errorList[1] = "☒ String only contains uppcase letters."; //reset the message to default
    }
    else
    { //if not empty
        if (S.Length > 0 && S.Length <= 40)
        { //valid string
            errorList[0] = "☑ String length is within 1-40 characters."; //show this message
        }
        else
        { // if not empty and out of range, show below message
            errorList[0] = "☒ String length is within 1-40 characters.";
        }

        bool IsCapitalLetter = Regex.IsMatch(S, @"^[A-Z]+$");  // create boolean variable to check if all chars are capital
        if (!IsCapitalLetter)
        {                           //if not, show below message
            errorList[1] = "☒ String only contains uppcase letters.";
        }
        else
        { //if yes, show below message
            errorList[1] = "☑ String only contains uppcase letters.";
        }
    }
```

Figure 2 - Code snippet for Validation class

This class validates input strings, ensuring they contain only uppercase letters and do not exceed a maximum length of 40 characters.

b) Demonstration

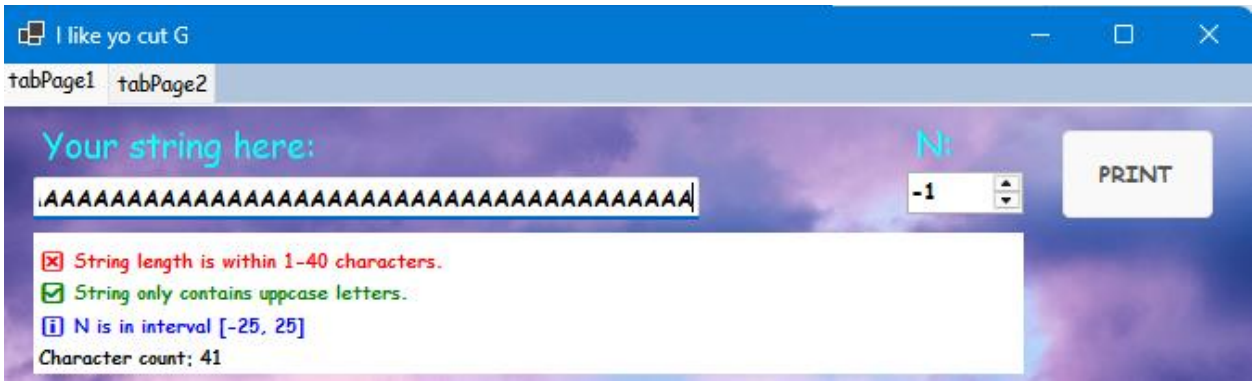


abc

-1

☑ String length is within 1-40 characters.
☒ String only contains uppcase letters.
🛈 N is in interval [-25, 25]
Character count: 3

Figure 3 - Input string contains lowercase letter(s)

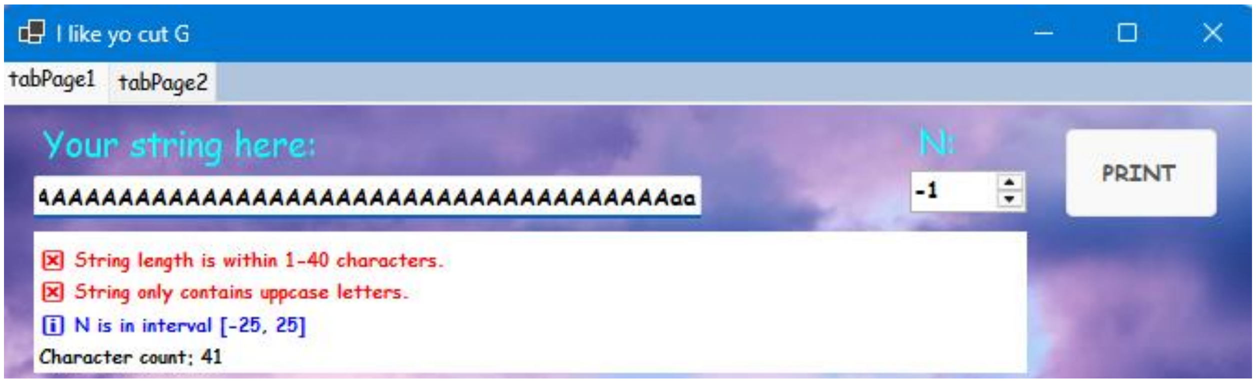


AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

-1

☒ String length is within 1-40 characters.
☑ String only contains uppcase letters.
🛈 N is in interval [-25, 25]
Character count: 41

Figure 4 - Input string length exceeding 40 characters



AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAaa

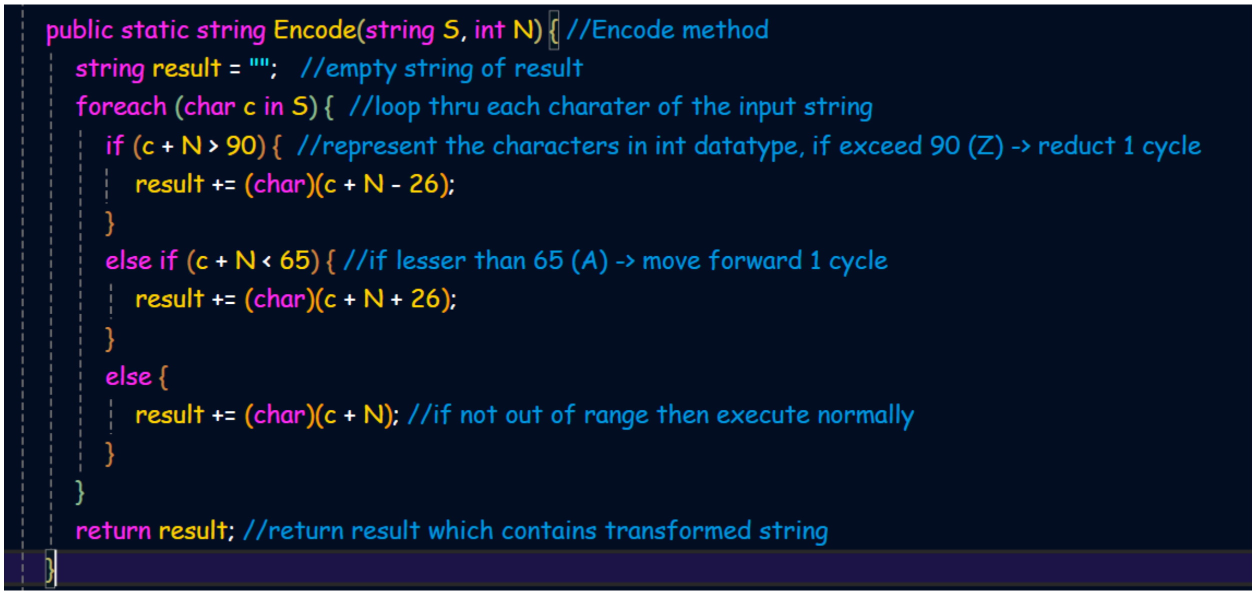
-1

☒ String length is within 1-40 characters.
☒ String only contains uppcase letters.
🛈 N is in interval [-25, 25]
Character count: 41

Figure 5 - Both conditions unsatisfied

3. Encode string feature

a) Code snippet

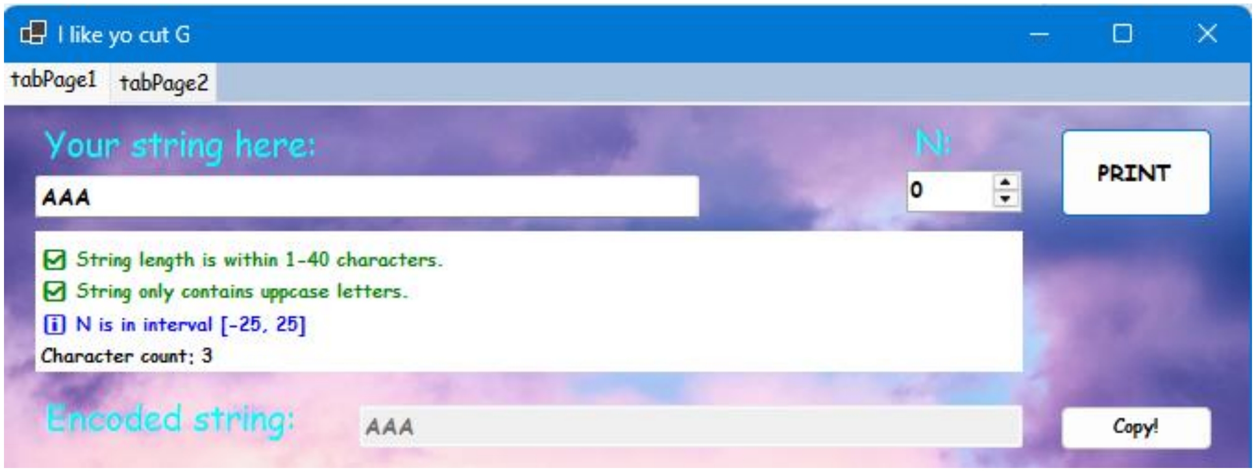


```
public static string Encode(string S, int N) { //Encode method
    string result = "";   //empty string of result
    foreach (char c in S) {  //loop thru each charater of the input string
        if (c + N > 90) {  //represent the characters in int datatype, if exceed 90 (Z) -> reduct 1 cycle
            result += (char)(c + N - 26);
        }
        else if (c + N < 65) { //if lesser than 65 (A) -> move forward 1 cycle
            result += (char)(c + N + 26);
        }
        else {
            result += (char)(c + N); //if not out of range then execute normally
        }
    }
    return result; //return result which contains transformed string
}
```

Figure 6 -Code snippet for Encode feature

When the user inputs a valid string and an "N" value (default: 0), the system processes the data and displays the output string upon clicking "PRINT."

b) Demonstration



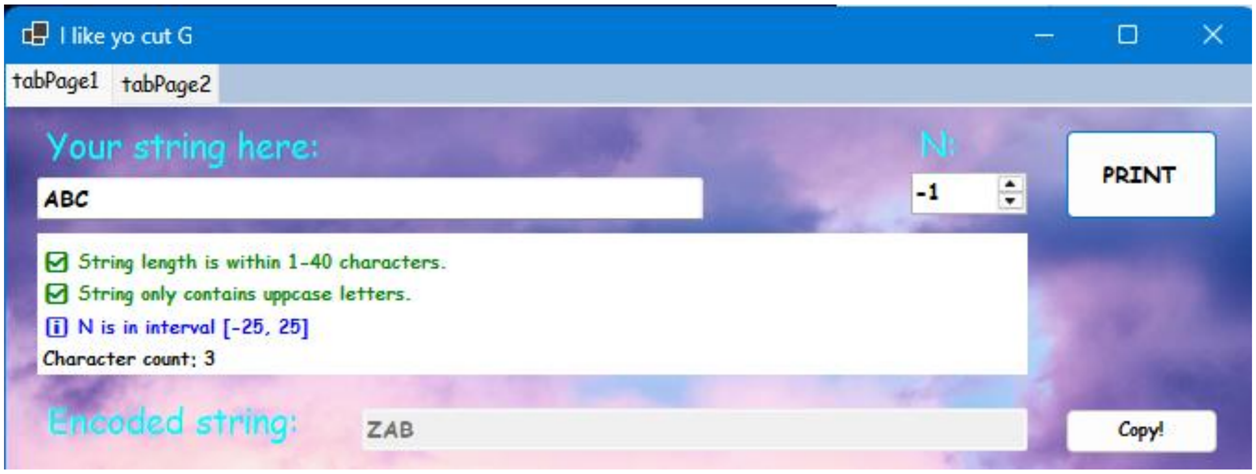
Your string here: AAA

N: 0

✓ String length is within 1-40 characters.
✓ String only contains uppcase letters.
ⓘ N is in interval [-25, 25]
Character count: 3

Encoded string: AAA

Figure 7 - Input string is "AAA" and N is 0



ABC

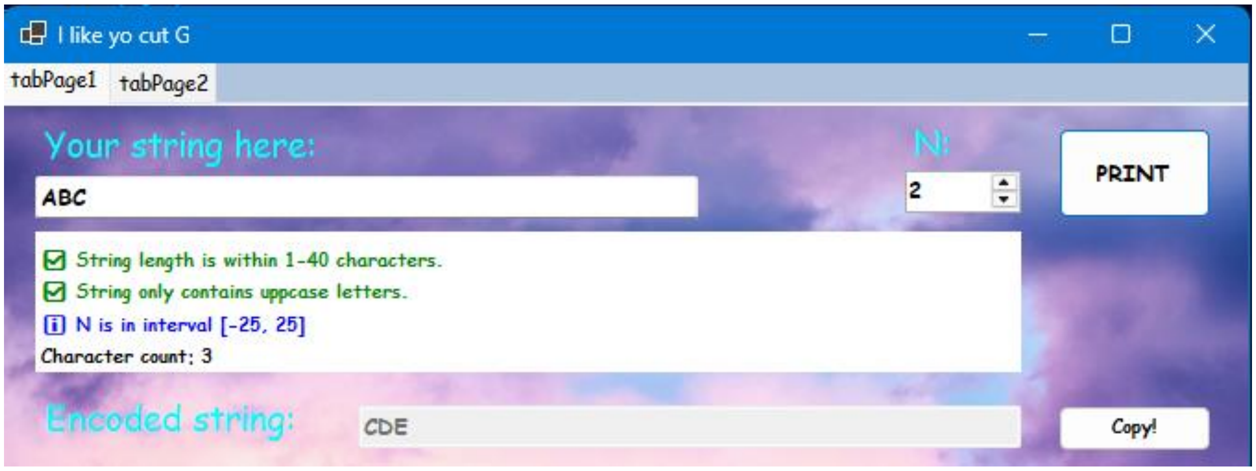
-1

☑ String length is within 1-40 characters.
☑ String only contains uppcase letters.
ⓘ N is in interval [-25, 25]
Character count: 3

ZAB

Figure 8 - Input string is "ABC" and N is -1

The system wraps alphabetically during encoding if characters exceed 'Z' or 'A,' ensuring proper cyclical character transformation in outputs.



ABC

2

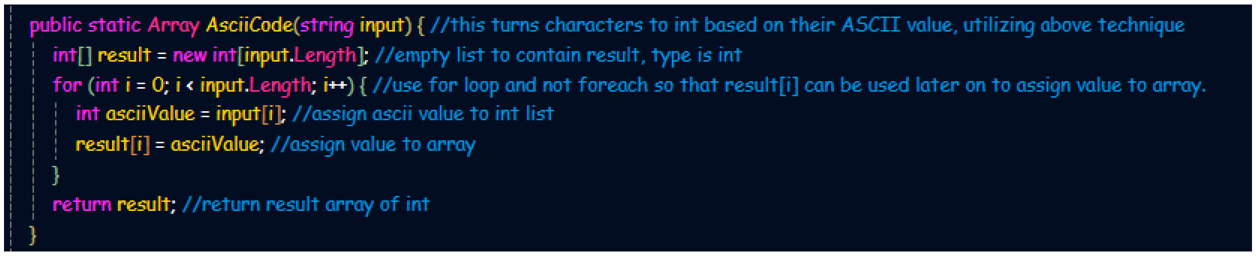
☑ String length is within 1-40 characters.
☑ String only contains uppcase letters.
ⓘ N is in interval [-25, 25]
Character count: 3

CDE

Figure 9 - Input string is "ABC" and N is 2

4. InputCode and OutputCode feature

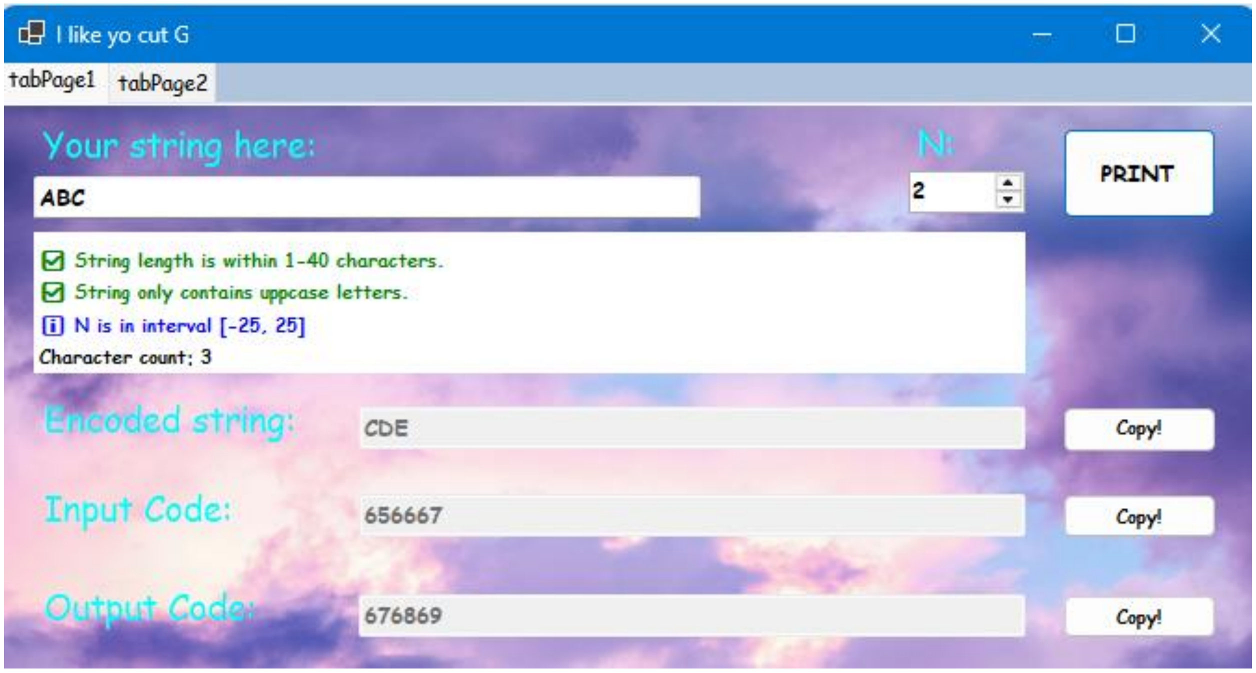
a) Code snippet



```
public static Array AsciiCode(string input) { //this turns characters to int based on their ASCII value, utilizing above technique
    int[] result = new int[input.Length]; //empty list to contain result, type is int
    for (int i = 0; i < input.Length; i++) { //use for loop and not foreach so that result[i] can be used later on to assign value to array.
        int asciiValue = input[i]; //assign ascii value to int list
        result[i] = asciiValue; //assign value to array
    }
    return result; //return result array of int
}
```

Figure 10 - Code snippet for InputCode and OutputCode

b) Demonstration



ABC

2

☑ String length is within 1-40 characters.
☑ String only contains uppcase letters.
[i] N is in interval [-25, 25]
Character count: 3

CDE

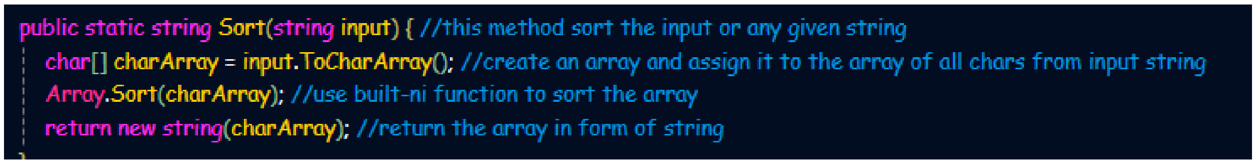
656667

676869

Figure 11 – Demonstration

5. InputSort feature

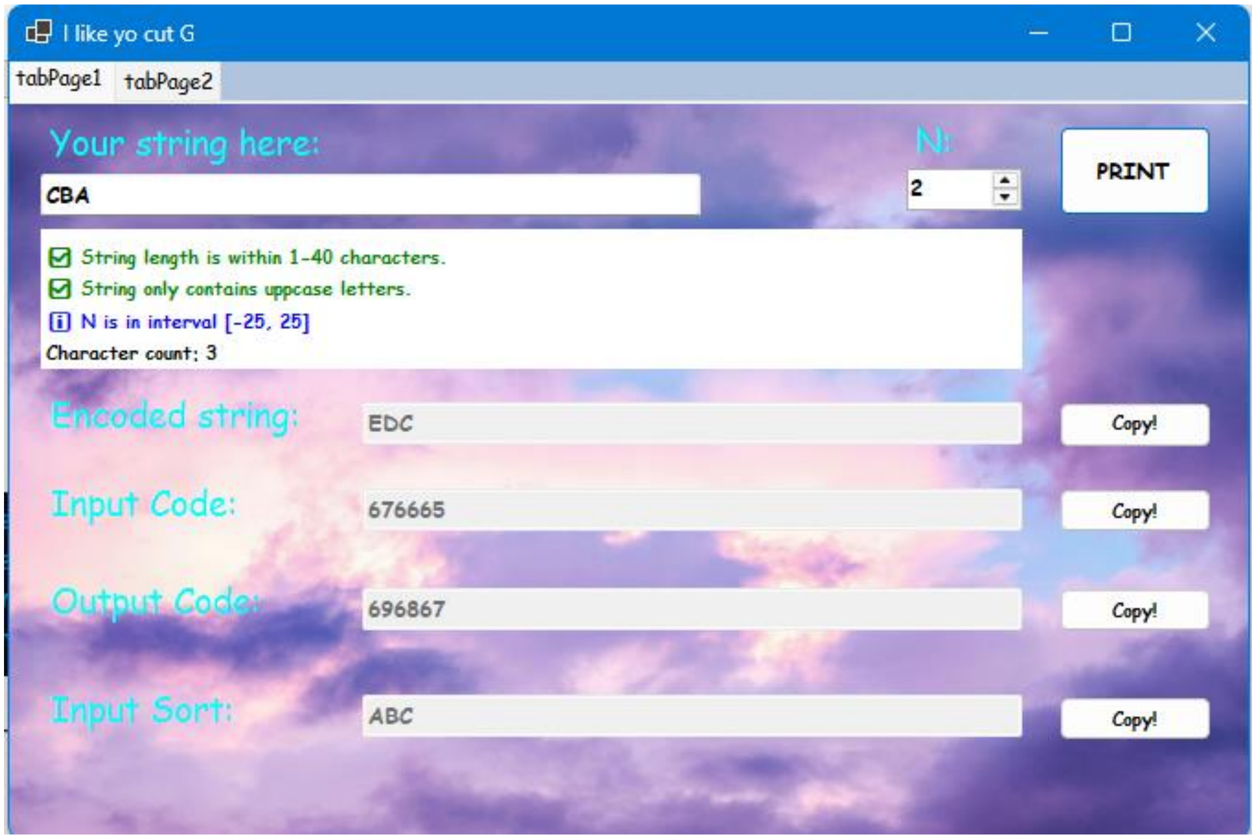
a) Code snippet



```
public static string Sort(string input) { //this method sort the input or any given string
    char[] charArray = input.ToCharArray(); //create an array and assign it to the array of all chars from input string
    Array.Sort(charArray); //use built-ni function to sort the array
    return new string(charArray); //return the array in form of string
```

Figure 12 - Code snippet for Sorting feature

b) Demonstration



Your string here: CBA

N: 2

☑ String length is within 1-40 characters.
☑ String only contains uppcase letters.
[i] N is in interval [-25, 25]
Character count: 3

Encoded string: EDC

Input Code: 676665

Output Code: 696867

Input Sort: ABC

Figure 13 - Input string is "CBA" and when sorted, being "ABC"