Hassan Harajly

Cis 200

9/25/17

**My String Program**

1. **Problem Statement**

Must create a class like the string class. It must hold a maximum of 25 characters. Will have various methods.

1. **Requirements**
   1. **Assumptions**

The user will enter integers, characters and symbols, only 25 of any combination.

Will enter input through command line input/output.

Programmers can also use class for their programs, we want low level of codependence(coupling).

* 1. **Specifications**

The class will contain the capability to store 25 characters, symbols and numbers.

Will have to create a character array to carry the input. Inside of the mystring class.

End array with “/0” in order to know when the end of array is.

Methods of the mystring class:

Size() will determine how many caharacters are in the string.

addstart(mystring)- will add the string parameter to the current string in the front of it.25 char limit.

Addend(mystring)-will add the string in the input parameter to the end of the current string.not above 25.

partString(startPos, length) – returns string from startPos for length given (handle startPos < 0, startPos = size returns null string, handle startPos > size).

replPartString(myString, startPos) – replaces charaters starting at startPos with input string – (what if exceeds 25?).

compareString(myString) – compare current value of string with input parameter string.

initString() – resets/initializes string to null string

setString(string) – assign string to myString data value

getString() – returns string of data from myString data value

printString() – prints myString data value to the monitor

1.The user will enter up to 25 characters.

If the user enters more than 25 characters they will be prompted with an error message to reenter.

When the mystring input is correctly stored, the user will be able to choose from the methods listed above depending upon what the user wants.

Methods will have a status variable to them, meaning if method cant be executed then method will return -1, if executed correctly it will return 0.

1. **Decomposition Diagram** (Used to break program down into components visually. Diagram can have as many components as needed. Defines functionality that will solve the problem – does NOT define a flow of actions)

String program

Input

process

Output

User will enter characters by the command line

Check if input is within limits.

Print out message when string Is input correctly

Display error message when user inputs to much data

Ask user which method they would like to use

User inputs which method they would like to use.

String will become modifiable with different strings.

String will be stored inside of a class and can use any method on string.

1. **Test Strategy**

* Valid Data
* Invalid Data
* files

1. **Test Plan Version 1**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Strategy | Test Number | Description | Input | Expected Output | Actual Output | Pass/Fail |
| Invalid Data | 1 | User enters 26 characters |  |  |  |  |
| Invalid Data | 2 | User enters  No data |  |  |  |  |
| Valid Data | 3 | User enters 24 characters |  |  |  |  |

1. **Initial Algorithm**
2. Create a mystring class, it will accept no more than 25 characters.

Will validate input by limiting the amount of characters inputted.

Once string is stored inside class. Methods will be called to modify string.

Will have appropriate get and set functions in order to set characters to the character array.

**Size** function will step through character array and count how many characters are inside of it. Then pass the size of the string to all other methods that need it.

**While chararray[i]!=”/0”**

**Increment counter variable.**

**Return increment variable**

**Addstart(mystring)** function will have access to the character array by parameter. Will make sure string isn’t already at 25 characters.

**If ((mystring+chararray<=24))**

**Do**

**replace chararray with mystring starting at 0**

**end if**

**Addend(mystring)-** will append string parameter to the current string. Will find out where the string ends and add the characters to it. If string exceeds 25 characters combined will display an error message.

If ((mystring+chararray<=24))

Do

add mystring to chararray

end if

**Partstring(startPos,length)-**returns string. Accepts a startpos integer and length integer in its parameters. Function must take into account the size of both strings being combined.

Replpartstring(mystring,startPos)-will use the starting position of the current string to be replaced with the input string.

for i<startpos

do

replace currentstring with mystring

end for.

Replwholestring(mystring)-rewrites current string. Will start at position 0 and delete current string and enter and a new string.

While(mystring[i]!=/0)

Loop until all characters cleared

End while

Rewrite string

Comparestring(mystring)-will compare current value of string with input parameter string.

Loop and compare each string until they reach \0

Will use the differences in asci values to compare each character in both strings.

Initstring()-resets string to null string.

Will continually loop and delete the string and initialize it to null.

loop and delete until ‘\0’ is found then set the string to nullptr.

Setstring(string)-assign string to myString data value.

Create character array and set userinput equal to it.

Getstring()

Return the character array when this function is called

Printstring()

Prints mystring data value to the monitor.

While(mystring!=/0)

Print each each character in the char array.

1. **Test Plan Version 2**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Strategy | Test Number | Description | Input | Expected Output | Actual Output | Pass/Fail |
| Invalid Data | 1 | User enters 26 characters | Hawurnfjrhejskslnrjdisnwkw | Incorrent input please input 25 characters or less |  |  |
| Invalid Data | 2 | User enters  No data |  | Nothing happens |  |  |
| Valid Data | 3 | User enters 24 characters | Aaaabcdefghiklmnopaaaaa | “Which method would you like to use on inputted string?”  Print menu |  |  |
| Invalid function call | 4 | User enters string to add to current string that exceeds the limit, and vice versa | (using add end function)  Add 2 strings equaling more than 25 characters | Please enter a total of 25 characters to be combined |  |  |
| Invalid part string function call | 5 | User enters invalid starting position for the function | -1 | Error Please enter a number between 0-25 |  |  |
| Invalid reppartstring function call | 6 | User enters invalid starting position for the function | -5 | Error  Please enter a number between 0-25 |  |  |

1. **Code**

**//hassan harajly**

**//cis 200**

**//10/1/2017**

**//this program is an attempt to make a class like the string class used in c++, it basically loads a character array and modifies its values easily through functions**

**#include <iostream>**

**#include <string>**

**using namespace std;**

**//precondition: constructor requires a character array to set to the stringarray**

**//post condition:forming a mystring object allows user to manipulate the character array easily.**

**class myString {**

**private:**

**char stringarray[27] = "jsjsjsjsjssjsjsjsjsjsjj";**

**int i, status = 0, z, startPos;**

**public:**

**myString();**

**int setstring(string);//assigns character array parameter to the character array inside mystring class.**

**int printstring();//prints the mystring object(the character array inside of object)**

**char \*getstring();//returns the character array as a char pointer its still somewhat useable as an array and is used inside other functions as so**

**int initString();//resets character array to null**

**int compareString(myString);//compares both strings to see if theyre different or the same.**

**int size();//returns the size of the array**

**int replWholeString(myString);//replaces current string with parameter string**

**int replPartString(myString, int);//replaces current string with string given at a specific position**

**myString partString(int, int);// returns a certain range of the current string**

**//int setstring(char[27], int\*);//**

**int addEnd(myString);//adds string to the end of current string**

**int addStart(myString);//adds string to the begining of current string**

**void errorfunction();**

**};**

**void myString::errorfunction()**

**{**

**if (status == -1)**

**{**

**cout << endl << "input error redo the entry you just attempted" << endl << endl;;**

**status = 0;**

**}**

**}**

**//precondition: mystring object with loaded array**

**//post condition: modified mystring object with new parameter starting off the modified array**

**int myString::addStart(myString inputstring)**

**{**

**int z = size() - 1;**

**if ((size() + inputstring.size()) <= 25)**

**{**

**for (int e = 0; e < inputstring.size(); e++)**

**{**

**for (int i = 0; i < size(); i++)//moves string up according to the length of the input string.**

**{**

**stringarray[z + 1] = stringarray[z];//sets the string to a position forward to make space for new incoming string**

**z--;**

**}**

**z = (size() - 1) + 1;**

**}**

**}**

**else**

**{**

**status = -1;**

**errorfunction();**

**return -1;**

**}**

**if ((size() + inputstring.size()) <= 25)**

**{**

**for (int i = 0; i < inputstring.size(); i++)**

**{**

**stringarray[i] = inputstring.getstring()[i];//this sets the old position of the string to a new string**

**}**

**return 0;**

**}**

**else**

**{**

**status = -1;**

**errorfunction();**

**return -1;**

**}**

**}**

**//precondition:mystring object with loaded array**

**//postcondition:currentstring is appended with new string**

**int myString::addEnd(myString inputstring)**

**{**

**if ((size() + inputstring.size()) <= 25)//will not allow user to enter more than a total of 25 characters to be combined**

**{**

**i = size();**

**for (int q = 0; q < inputstring.size(); q++)//this sets the current strings empty positions with a new string**

**{**

**stringarray[i] = inputstring.getstring()[q];**

**i++;**

**}**

**return 0;**

**}**

**else**

**{**

**status = -1;**

**errorfunction();**

**return -1;**

**}**

**}**

**//precondition: starting position of desired return string, and length of the total return value**

**//postcondition: return a snippet of the current string chosen by the input parameter**

**myString myString::partString(int startPos, int length)**

**{**

**char temp[27];**

**if ((length <= 25 && length >= 1) && (startPos >= 1 && startPos <= 24))**

**{**

**int t;**

**for (t = 0; t < length; t++)**

**{**

**temp[t] = stringarray[startPos - 1];//sets new array equal to desired return value**

**startPos++;**

**//cout << t<<endl;**

**}**

**temp[t] = NULL;**

**//cout << temp << endl;**

**myString tempstring;//sets array equal to new string**

**tempstring.setstring(temp);**

**return tempstring;//returns the object to be used**

**}**

**else**

**{**

**status = -1;**

**}**

**errorfunction();**

**}**

**//precondition: mystring object,the position of where the function should start replacing the old string with the newstring.**

**//postcondition: replaced string is returned**

**//desc:replaces a part of the current string**

**int myString::replPartString(myString repstring, int startPos)**

**{**

**int t = 0;**

**if (startPos <= 1 && startPos >= 25)**

**{**

**status = -1;**

**errorfunction();//start position for string must be between 1 and 25**

**return -1;**

**}**

**else**

**for (t = 0; t <repstring.size(); )//loops for how ever big the input string size is**

**{**

**stringarray[startPos - 1] = repstring.getstring()[t];//sets new string equal to input string**

**startPos++;**

**t++;**

**}**

**return 0;**

**}**

**//precondition:loaded mystring object**

**//postcondition:new inputed string**

**//description: empties current string and inputs new string**

**int myString::replWholeString(myString input)**

**{**

**//char inputarray[25] = input.getstring;**

**if (size() <= 25)**

**{**

**i = 0;**

**while (stringarray[i] != NULL)//empties current string**

**{**

**stringarray[i] = 0;**

**i++;**

**}**

**i = 0;**

**for (int y = 0; y < input.size(); y++)//reloads array with new input**

**{**

**stringarray[i] = input.getstring()[i];**

**i++;**

**}**

**return 0;**

**}**

**else**

**{**

**status = -1;**

**errorfunction();**

**return -1;**

**}**

**}**

**//precondition: loaded array**

**//postcondition returns size of the array**

**int myString::size()**

**{**

**i = 0;**

**while (stringarray[i] != '\0')**

**{**

**i++;//increments accoring to number of elements of array**

**}**

**errorfunction();**

**return i;**

**}**

**//precondition: input string**

**//postcondition: returns if the strings are similar or different**

**int myString::compareString(myString inputstring)**

**{**

**bool same = true;**

**if (inputstring.size() > 25 || size() > 25)**

**{**

**status = -1;**

**}**

**if (inputstring.size() == size())//if the arrays are equal it will evaluate the characters**

**{**

**for (int p = 0; p < inputstring.size(); p++)**

**{**

**if ((int(inputstring.getstring()[p])) < (int(getstring()[p])))//will loop until a difference is found among the characters**

**{**

**//inputstring.printstring();**

**same = false;**

**}**

**if ((int(inputstring.getstring()[p])) > int((getstring()[p])))**

**{**

**same = false;**

**//printstring();**

**}**

**}**

**if (same == true)//if same is true that means the strings are the same and vice versa**

**{**

**cout << "strings are the same" << endl;**

**}**

**if (same == false)**

**{**

**cout << "strings are different" << endl;**

**}**

**}**

**else**

**{**

**cout << "strings are different" << endl;**

**}**

**errorfunction();**

**return 0;**

**}**

**//precondition: loaded char array**

**//postcondition:resets exisiting string**

**int myString::initString()**

**{**

**i = 0;**

**if (stringarray[0] == NULL)**

**{**

**status = -1;//if the first position is already null will exit**

**return -1;**

**}**

**while (stringarray[i] != NULL)//resets the string to null**

**{**

**stringarray[i] = NULL;**

**i++;**

**}**

**errorfunction();**

**return 0;**

**}**

**//precondition:loaded array**

**//postcondition: char array**

**//returns the char array as a pointer**

**char \*myString::getstring()//returns a char pointer which is sort of what a char array is because we cant return a array in c++ we do it like this**

**{**

**if (size() <= 25)**

**{**

**return stringarray;//returns the string if the size is smaller than 25**

**}**

**else**

**{**

**errorfunction();**

**status = -1;**

**}**

**}**

**//precondition:loaded array**

**//post condition:print array contents to screen**

**int myString::printstring()**

**{**

**//cout << size();**

**i = 0;**

**if (stringarray[0] == NULL)//will not print if the string is empty or null**

**{**

**status = -1;**

**errorfunction();**

**return -1;**

**}**

**while (stringarray[i] != NULL)**

**{**

**cout << stringarray[i];//prints all the elements of string**

**i++;**

**}**

**errorfunction();**

**return 0;**

**}**

**//constructor uses setstring function to initialize the char array**

**myString::myString()**

**{**

**}**

**//precondition: string from main or anywhere else**

**//post condition: loaded and validated array**

**int myString::setstring(string input)**

**{**

**if (input.size() <= 25)//doesnt accept size greater than 25**

**{**

**i = 0;**

**while (stringarray[i] != '\0')**

**{**

**//will reinitialize string to null**

**stringarray[i] = NULL;**

**i++;**

**}**

**i = 0;**

**while (stringarray[i] != '\0')**

**{**

**stringarray[i] = 0;//also reinitilizes array just incase**

**i++;**

**}**

**i = 0;**

**while (input[i] != NULL)**

**{**

**stringarray[i] = input[i];//sets the string here**

**i++;**

**}**

**}**

**else {**

**status = -1;**

**errorfunction();**

**return -1;**

**}**

**//errorfunction();**

**}**

**int main()**

**{**

**cout << "welcome to string master 5000! Click enter to begin" << endl;**

**cin.ignore(100, '\n');**

**myString newstring;**

**int input = 0;**

**string stringinput,st;**

**cout << "enter your new string input" << endl;**

**getline(cin,stringinput);//cannot use cin here because it doesnt pick up white spaces so we use cin.get line**

**newstring.setstring(stringinput);//sets the string to the classes char array**

**for (int i = 0; i < 1;)//will continually loop the menu for the user to select from for the current string**

**{**

**cout << "enter new string enter 1" << endl;**

**cout << "print your custom string enter 2" << endl;**

**cout << "reset your string enter 3" << endl;**

**cout << "to compare a new string and current string enter 4" << endl;**

**cout << "to replace your whole string with a new one enter 5" << endl;**

**cout << "to replace only part of your string enter 6" << endl;**

**cout << "to return only part of your string enter 7" << endl;**

**cout << "to append a string to your current string enter 8" << endl;**

**cout << "to add a string to the front of a existing string enter 9" << endl;**

**cout << "to find the size of your string enter 10" << endl;**

**cout << "enter 0 to exit" << endl;**

**cin >> input;**

**while (cin.fail() || !(input >= 0 && input <= 10))//validates input is 1->10**

**{**

**cout << "error please reenter an integer between 0 and 10";**

**cin.clear();**

**cin.ignore(100, '\n');**

**cin >> input;**

**}**

**if (input == 1)//chained decision structure to determine which function the user wants to use**

**{**

**cin.ignore(100, '\n');**

**cout << endl << "enter your new string input" << endl;**

**getline(cin,stringinput);**

**newstring.setstring(stringinput);**

**}**

**else if (input == 2)//use else if in order to not make computer contiue checking the parameters if one is found to be true**

**{**

**cin.ignore(100, '\n');**

**int num = 0;**

**string h;**

**cout << endl; newstring.printstring(); cout << endl;**

**cout << "enter any input to continue" << endl;**

**cin >> h;**

**}**

**else if (input == 3)**

**{**

**cin.ignore(100, '\n');**

**newstring.initString();**

**cout << "your string has been reset" << endl;**

**}**

**else if (input == 4)**

**{**

**cin.ignore(199, '\n');**

**cout << "enter new string to compare to your current string" << endl;**

**getline(cin, st);**

**myString input;**

**input.setstring(st);**

**newstring.compareString(input);**

**cout << "enter 1 to continue" << endl;**

**}**

**else if (input == 5)**

**{**

**cin.ignore(100, '\n');**

**myString into;**

**string inp;**

**cout << "enter a new string to replace your current one" << endl;**

**getline(cin, inp);**

**into.setstring(inp);**

**newstring.replWholeString(into);**

**}**

**else if (input == 6)**

**{**

**cin.ignore(100, '\n');**

**string sta;**

**int num = 0;**

**myString into;**

**string inp;**

**cout << "enter a new string to replace your current one" << endl;**

**cin >> inp;**

**into.setstring(inp);**

**cout << "enter a starting position to replace your current string with" << endl;**

**cin >> num;**

**newstring.replPartString(into, num);**

**newstring.printstring();**

**cout << "enter any input to continue." << endl;**

**cin >> sta;**

**}**

**else if (input == 7)**

**{**

**cin.ignore(1000, '\n');**

**int start = 0;**

**int length = 0;**

**cout << "Enter the starting position of the current string you want to return" << endl;**

**cin >> start;**

**while (cin.fail() || !(input >= 1 && input <= 25))//validates input is 1->25**

**{**

**cout << "error please reenter an integer between 0 and 25";**

**cin.clear();**

**cin.ignore(100, '\n');**

**cin >> input;**

**}**

**cout << "enter the length of the string you want to return" << endl;**

**cin >> length;**

**while (cin.fail() || !(input >= 1 && input <= 25))//validates input is 1->25**

**{**

**cout << "error please reenter an integer between 0 and 25";**

**cin.clear();**

**cin.ignore(100, '\n');**

**cin >> input;**

**}**

**newstring = newstring.partString(start, length);**

**//newstr.printstring();**

**}**

**else if (input == 8)**

**{**

**cin.ignore(100, '\n');**

**myString strn;**

**string strng;**

**cout << "Enter the new string you would like to add to the end of your current string" << endl;**

**getline(cin, strng);**

**strn.setstring(strng);**

**newstring.addEnd(strn);**

**}**

**else if (input == 9)**

**{**

**cin.ignore(100, '\n');**

**myString strn;**

**string strng;**

**cout << "Enter the string you would like to add to the begining to your current string" << endl;**

**getline(cin,strng);**

**strn.setstring(strng);**

**newstring.addStart(strn);**

**}**

**else if (input == 10)**

**{**

**cin.ignore(100, '\n');**

**cout << endl << "your string size is " << newstring.size() << endl << endl;**

**}**

**else if (input == 0)//if user inputs 0 the program will end**

**{**

**cin.ignore(100, '\n');**

**i = 1;**

**}**

**}**

**return 0;**

**}**

1. **Updated Algorithm**

Create a mystring class, it will accept no more than 25 characters.

Will validate input by limiting the amount of characters inputted, and with status variable

Once string is stored inside class. Methods will be called to modify string.

Will have appropriate get and set functions in order to set characters to the character array.

**Size** function will step through character array and count how many characters are inside of it. Then pass the size of the string to all other methods that need it.

**While chararray[i]!=”/0”**

**Increment counter variable.**

**Return increment variable**

**Errorfunction()**

**Addstart(mystring)** function will have access to the character array by parameter. Will make sure string isn’t already at 25 characters.

**If ((mystringsize+chararraysize<=25))**

**Do**

**replace chararray with mystring starting at 0**

**end if**

**errorfunction()**

**Addend(mystring)-** will append string parameter to the current string. Will find out where the string ends and add the characters to it. If string exceeds 25 characters combined will display an error message.

If ((mystring+chararray<=25))

Do

add mystring to chararray

end if

errorfunction()

**Partstring(startPos,length)-**returns string. Accepts a startpos integer and length integer in its parameters. Function must take into account the size of both strings being combined.

Mystring tempclass

**For I to length**

**Newarray[i]=Chararray[startPos]**

**Tempclass.setstring(newarray)**

**Errorfunction()**

Replpartstring(mystring,startPos)-will use the starting position of the current string to be replaced with the input string.

Mystring newobject

for i< newobject.size()

do

currentstring[startpos-1] =newobject.getstring[i];

end for.

Replwholestring(mystring)-rewrites current string. Will start at position 0 and delete current string and enter and a new string.

If (size()<=25)

While(mystring[i]!=/0)

Loop until all characters cleared

End while

For i=0;I to inputstring.size() i=i+1

Mystring[i]=input.getstring[i]

End for

End if

Comparestring(mystring)-will compare current value of string with input parameter string.

bool same=true;

if inputstring.size > 25 or size() > 25

status = -1;

end if

if inputstring.size() == size()

for p = 0 to inputstring.size() p=p+1

if ((int(inputstring.getstring()[p])) < (int(getstring()[p])))

//inputstring.printstring();

same = false;

end if

if inputstring.getstring()[p] > getstring()[p]

same = false;

End if

End for

Errorfunction()

Will use the differences in asci values to compare each character in both strings.

Initstring()-resets string to null string.

i = 0

if stringarray[0] == NULL

status = -1;

return -1;

end if

while stringarray[i] != NULL

stringarray[i] = NULL;

i=i+1

end while

errorfunction()

Setstring(string)-assign string to myString data value.

While newstring!=\0

I=i+1

End while

If i<=25

While mystring[i] !=\0

Mystring[i] = null

End while

I=0

While newstring[i] !=\0

Mystring[i]=inputstring[i]

I=i+1

End while

End if

Else

Status =-1

Error function()

Return -1

Errorfunction()

Getstring()

Return stringarray

Printstring()

While(mystring!=/0)

Print Stringarray[i]

I=i+1

End while

Errorfunction()

{

If status =-1

Print input error redo the entry you just attempted

}

Main function()

Print function menu

Input user choice

If (userchoice >=0 and userchoice<=10)

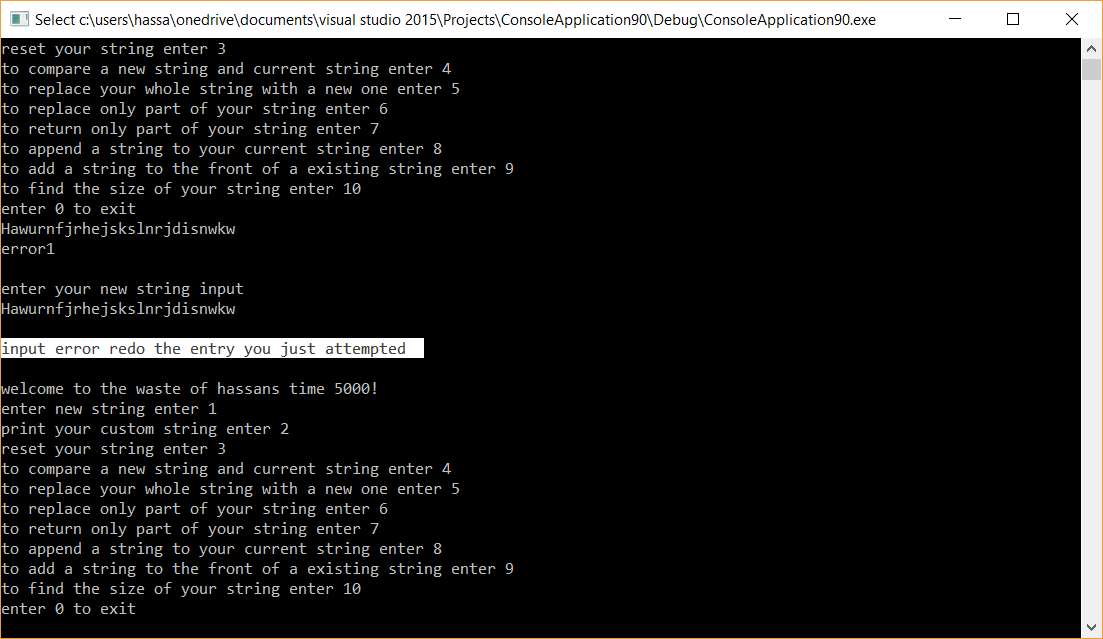
Select function

1. **Test Plan Version 3**

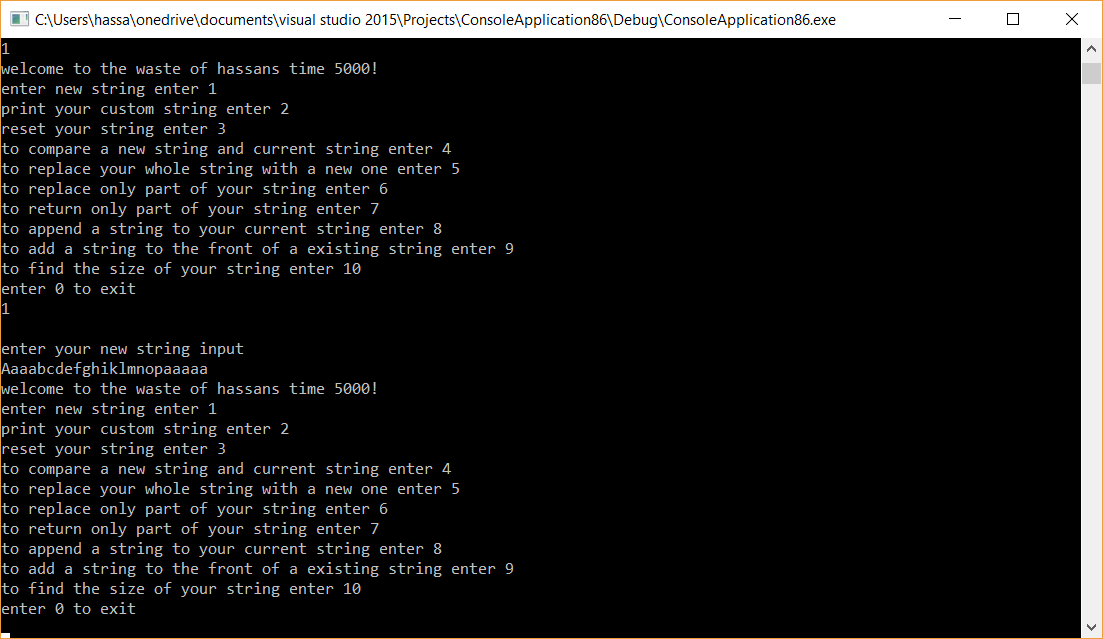
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Strategy | Test Number | Description | Input | Expected Output | Actual Output | Pass/Fail |
| Invalid Data | 1 | User enters 26 characters | Hawurnfjrhejskslnrjdisnwkw | input error redo the entry you just attempted | input error redo the entry you just attempted | pass |
| Invalid Data | 2 | User enters  No data |  | No actions | No actions | Pass |
| Valid Data | 3 | User enters 24 characters | Aaaabcdefghiklmnopaaaaa | “Which method would you like to use on inputted string?”  Print menu | “Which method would you like to use on inputted string?”  Print menu | pass |
| Invalid function call | 4 | User enters string to add to current string that exceeds the limit, and vice versa | (using add end function)  Add 2 strings equaling more than 25 characters | input error redo the entry you just attempted | input error redo the entry you just attempted | Pass |
| Invalid part string function call | 5 | User enters invalid starting position for the function | -1 | Error Please enter a number between 0-25 | input error redo the entry you just attempted | Pass |
| Invalid reppartstring function call | 6 | User enters invalid starting position for the function | -5 | Error  Please enter a number between 0-25 | input error redo the entry you just attempted | pass |
| Addstart function input invalid input | 7 | User enters more than 25 characters in total to add to the current string | asdfhjkadsfasdfasfdsafasfsadfafasfs | input error redo the entry you just attempted | input error redo the entry you just attempted | Pass |
| User prints string after reseting it | 8 | User enters #3(string reset function) then enters #2 to print it | 3, 2 | input error redo the entry you just attempted | input error redo the entry you just attempted | Pass |
| User enters invalid input during menu selection | 9 | User selects a character | P | error please reenter an integer between 0 and 10 | error please reenter an integer between 0 and 10 | pass |
| User enters symbol for menu selection | 10 | User enters symbol | / | error please reenter an integer between 0 and 10 | error please reenter an integer between 0 and 10 | pass |
| Invalid function call | 11 | User tries to call string compare function after reseting string | 3,4 | Strings are different | Strings are different | pass |
| Valid program start | 12 | User enters string successfully when prompted | Helloworld | Menu displays | Menus displays | pass |
| Valid setstring function call | 13 | User enters string from menu number 1 | Heyyyyy | Menu display | Menudisplays | pass |
| Valid print function after setstring function | 14 | User prints their own entered string | 2 | Prints heyyyyy | Prints heyyyyy | pass |
| Valid setfunction after string reset | 15 | After reseting string check to see if user can reinput a new string | 3,1,newstring,2 | Newstring | Newstring | Pass |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

1. **Screenshots**

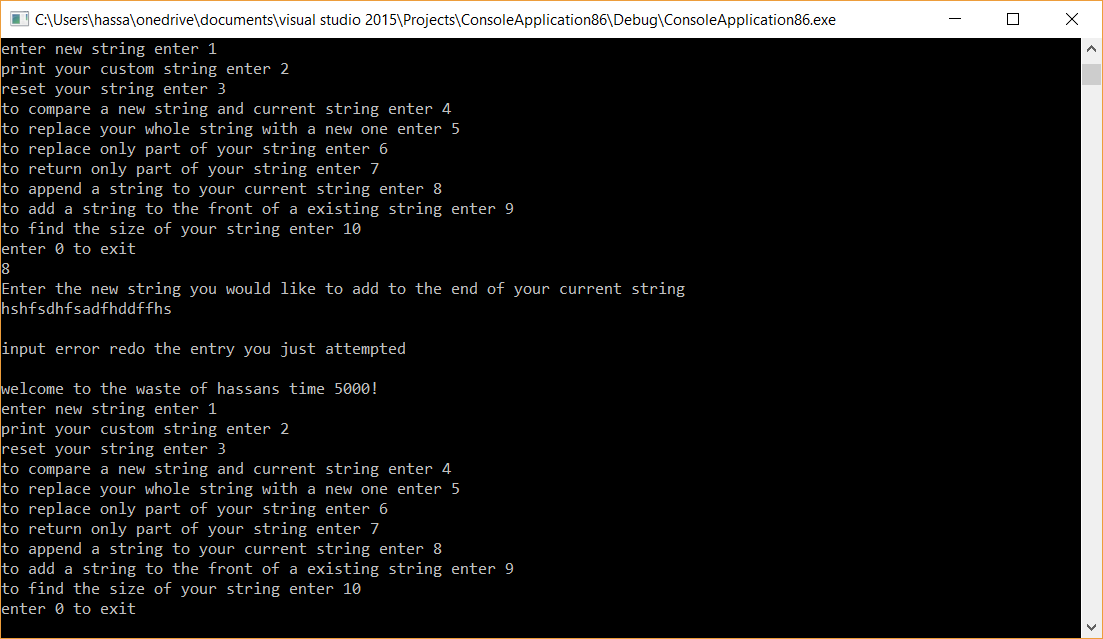
Test case 1



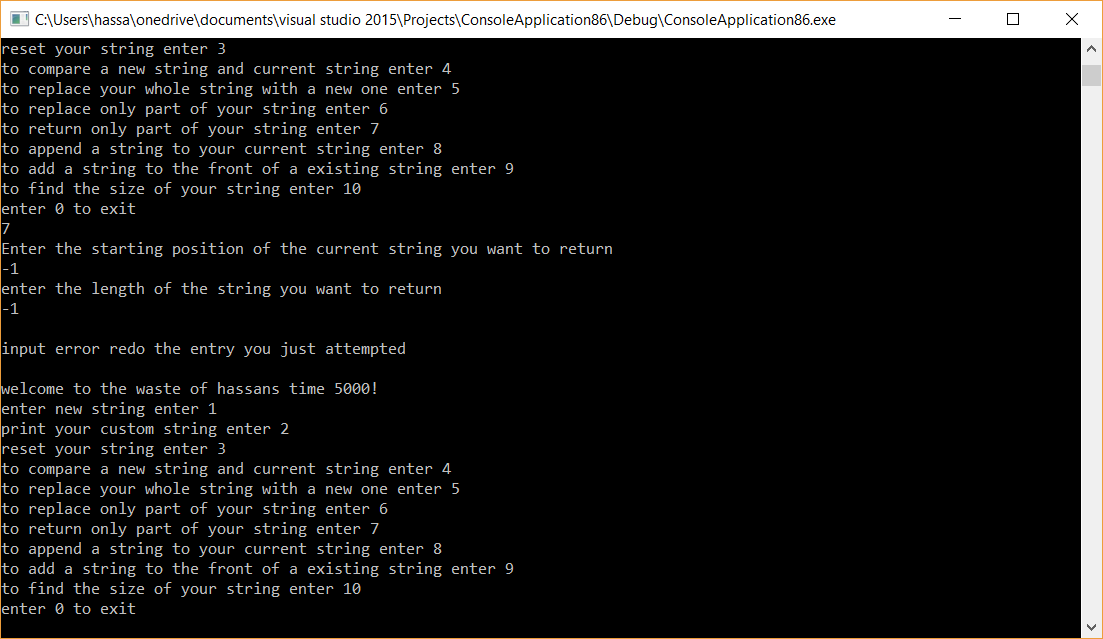
Test case 3:



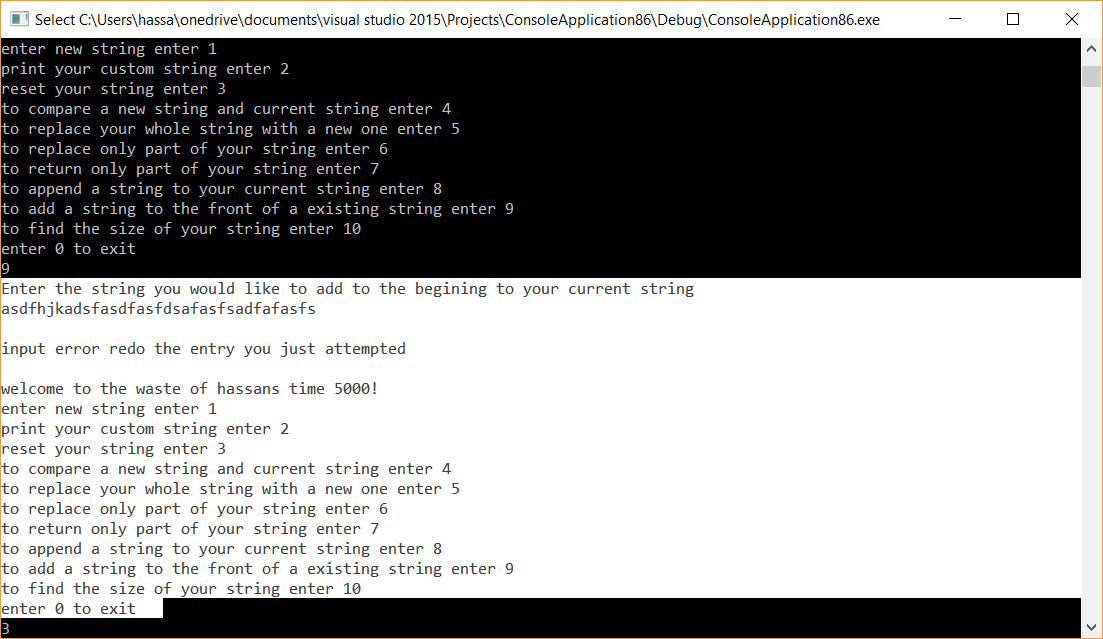
**Test case 4:**



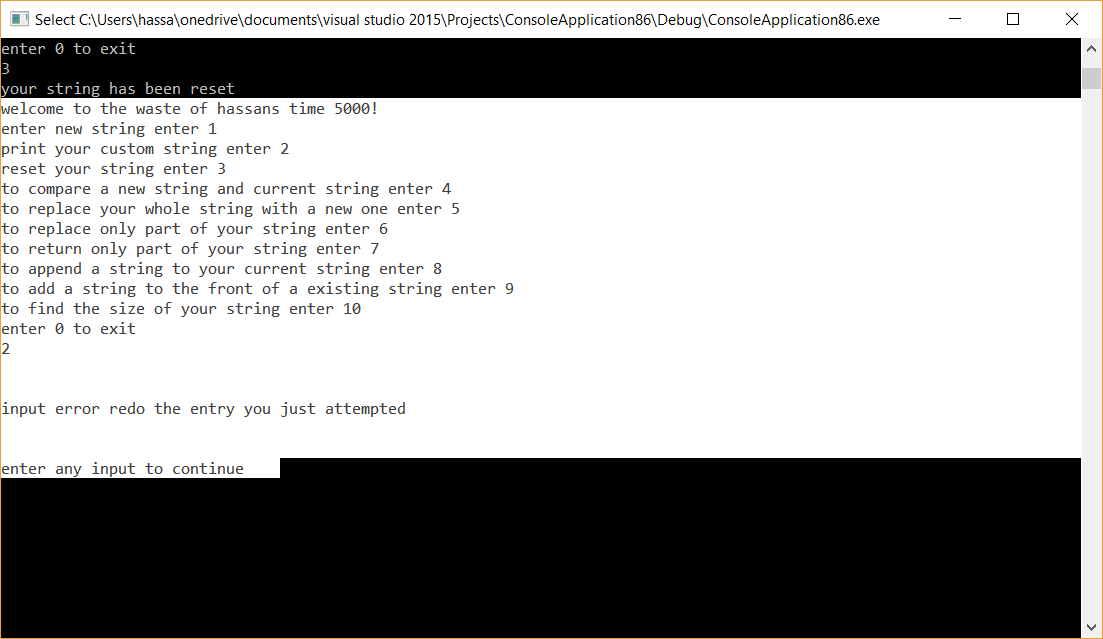
**Test case 5 and 6**



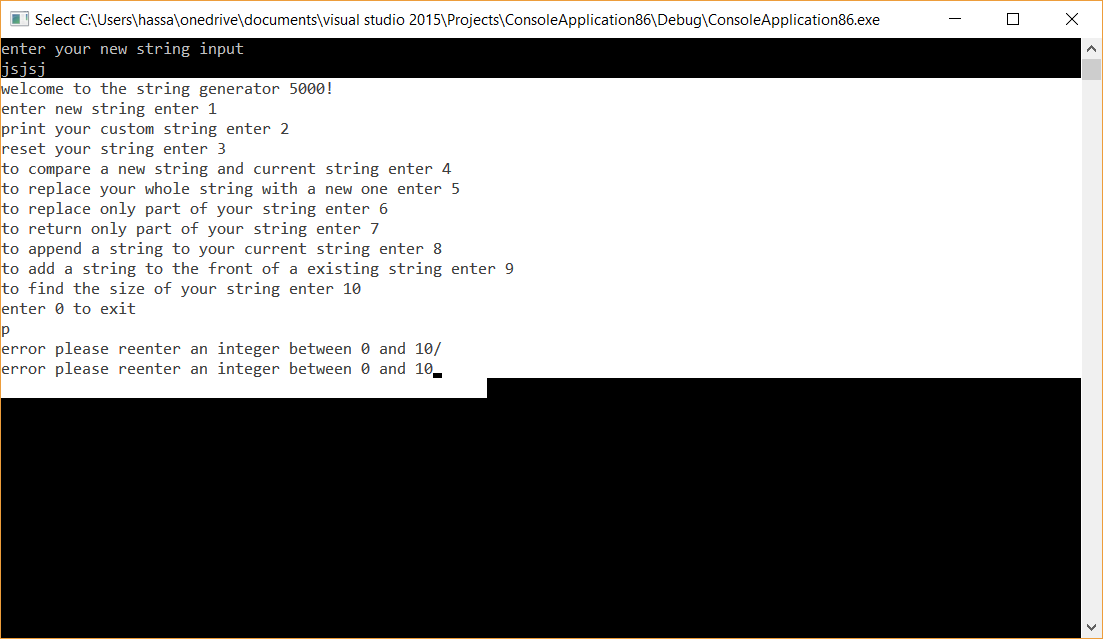
**Test case 7:**



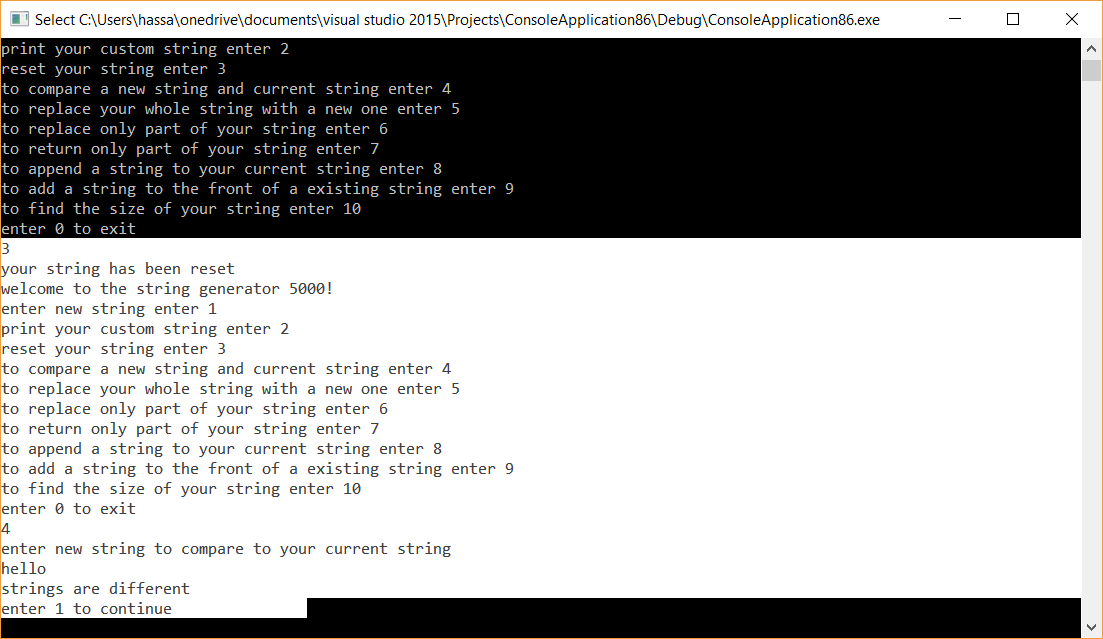
**Test case 8:**



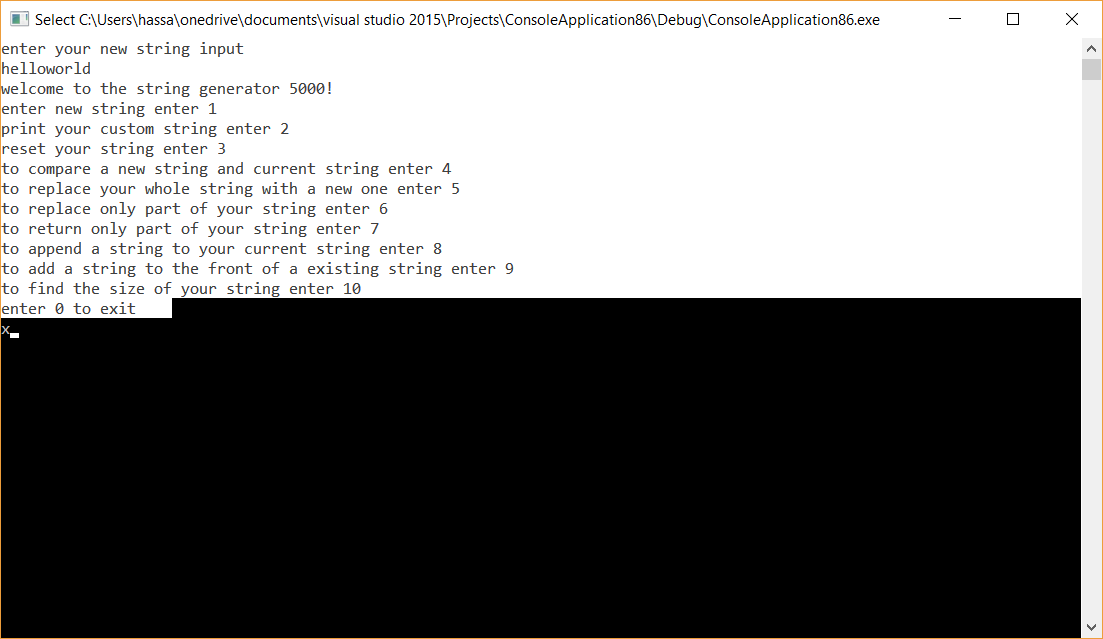
**Test cases 9 and 10:**



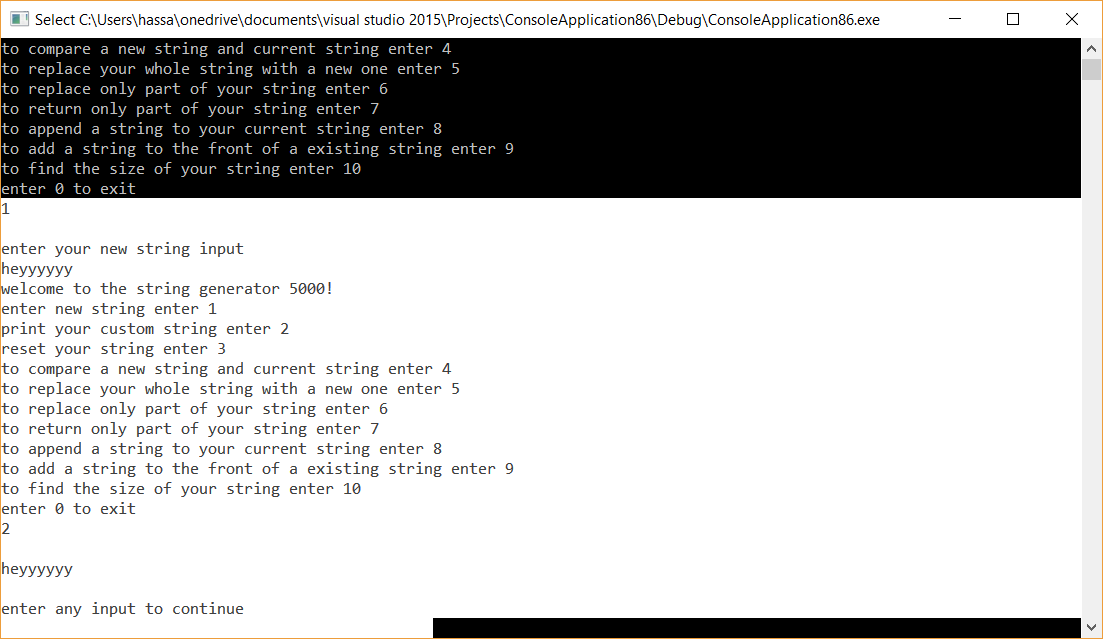
**Test case 11:**



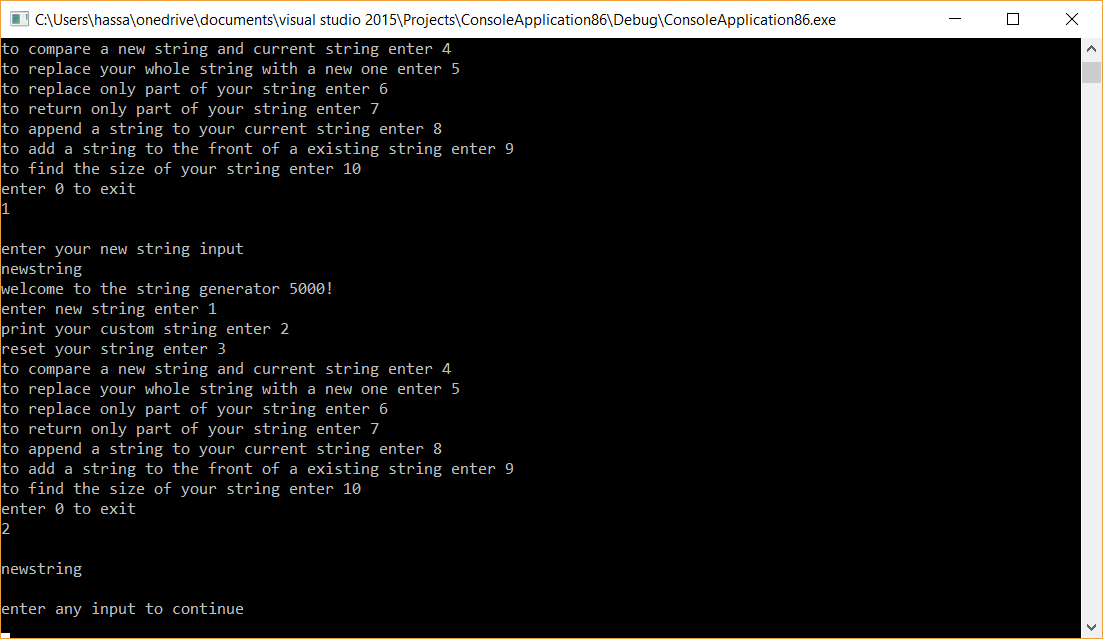
**Test case 12:**



**Test case 13&14:**



**Test case 15:**



1. **Error Log**

|  |  |  |
| --- | --- | --- |
| Error Type | Cause of Error | Solution to Error |
| Part string was not returning correct string input | For loop was not correctly executing due to wrong variable | Used the correct variable inside of for loop |
| String with spaces was dropping anything after the space | Cin doesn’t carry the white space | Changed cin to getline for the array inside of main. |
|  |  |  |

1. **Status**

The program works 100% with assumptions in place

CRC CARD.

**myString class**

**responsibilities**

use class as a variable for storing a string of characters

set a given amount of characters composed of the alphabet, numbers and symbols to the class

print the data onto the screen.

Determine the amount of data given by user input.

Add a new string of characters to the front of previously entered mystring variable

Add a new string of characters to the end of previously entered mystring variable

Return a specific number of characters

Modify the string starting from a certain position until the end

Replace the whole string with a new string

Compare 2 objects(strings) together

Reset strings back to null

Return string

Display the string

Uml:

+setstring(char input[27]) :int

+printstring():int

+getstring():char pointer

+initString(): int

+compareString(myString): int

+size(): int

+ replWholeString(myString) :int

+ replPartString(myString, int): int

+ partString(int , int): mystring

+ addEnd(myString) :int

+ addStart(myString): int

+ errorfunction():void