**Object Oriented Programming – Assignment 2**

**Spring 2022**

**Note: There are two questions for this assignment.**

**Q1: Write a program that performs following string manipulation functions:**

**Important Instruction:**

1. Subscript operator and integer iterators are allowed, do not use offset notations and pointers iterators.
2. Pass all the pointers by value unless you explicitly need a pointer to be changed in callee.
3. Make sure that you DO NOT consume any single extra byte.
4. **void StringConcatenate(c-string1, c-string2)**

Write a function that takes two strings inputs and appends str2 at the end of str1. **Don not change the return type Void.** For example,

String 1: “Happy Birthday” (Input String 1 doesn’t have any extra space)

String 2: “ to you !”

After StringConcatenate,

String 1: “Happy Birthday to you !”

String 2: “ to you !”

1. **Char\*\* StringTokens(char\*)**

Write a function which takes a string and returns an array of words in the string. For example:

String: I am a student of OOP in FAST-NU

Function StringTokens returns:

|  |
| --- |
| I  am  a  student  of  OOP  In  FAST-NU |

Hint: words are separated by spaces.

Note: Do not consume space of single extra character. Token printing is not part of this function.

1. **Char\*\* InverseStringTokens(char\*)**

Write a function which takes a string and returns its words in reverse order. Use previous function to accomplish this task. For example:

String: I am a student of OOP in FAST-NU

Function returns Tokens in reverse order:

|  |
| --- |
| FAST-NU  in  OOP  of  student  a  am  I |

Note: Do not consume space of single extra character.

1. **Char\* ReverseSentence(char\*)**

Write a function that takes a sentence and returns its inverse, use previous functions to accomplish this task. For example

String: “I am Pakistani”

After calling ReverseSentence

String: “Pakistani am I” (Return new string. Do not change the original string. Printing is not part of this function.)

1. **Void CompressString(char\*) (Submission not required.)**

Write a function that takes a string and if it finds more than one consecutive occurrences of a character in the string, it removes the extra occurrences. For example:

String: “a”

String after compression: “a”

String: “aaaaaaa”

String after compression: “a”

String: “bbabbbbbcccddddddddddeffffg”

String after Compression: “babcdefg”

Note: Do not use any extra string inside the function.

**Important Note:**

* You cannot change the function prototypes given in the questions.
* You cannot use break or goto statements. Breaks are allowed in switch cases.
* Built-in string functions are not allowed. Use your own string helper functions wherever you need.
* Violation of any of instructions may result in ZERO credit or deduction of marks.
* Submit one running cpp file and your data file. Compressed files are not allowed in submission.

*Create a main program and then test all of these functions. You must dynamically allocate and deallocate memory to all the strings in your program (except the temporary buffer). There should not be any memory leakages and dangling pointers in your program.*

**Sample Run:**

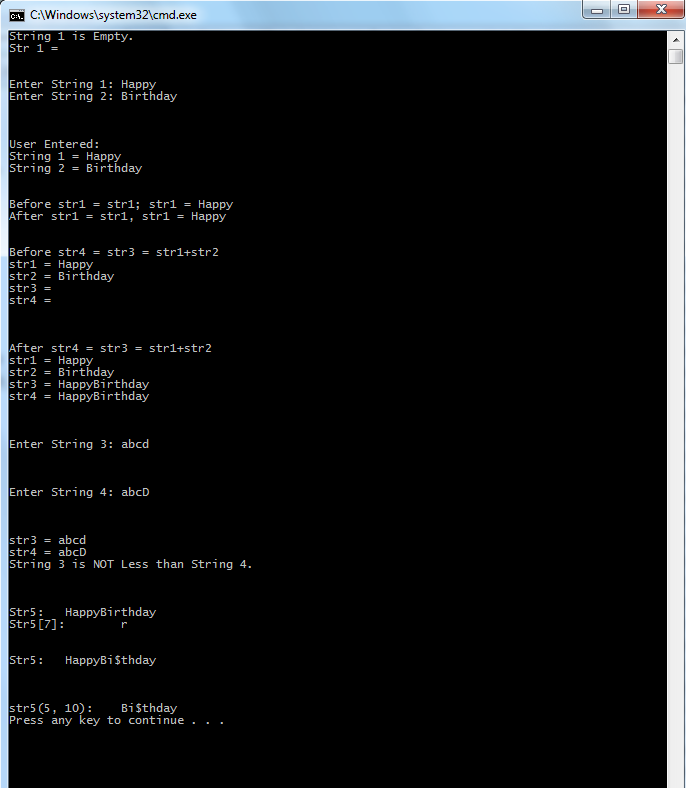
|  |
| --- |
| Testing StringConcatenate:  String 1: “I am a student”  String 2: “ of OOP in FAST-NU”  After Concatenation:  String 1: “I am a student of CP in FAST-NU”  String 2: “of OOP in FAST-NU”  ------------------------------------------------------------------------------------------------------------  Testing StringTokens:  Tokens of String 1 are as follows:  I  am  a  student  of  CP  In  FAST-NU  ------------------------------------------------------------------------------------------------------------  Testing InverseStringTokens:  Tokens of the string in reverse order are as follows:  FAST-NU  in  CP  of  student  a  am  I  ------------------------------------------------------------------------------------------------------------  Testing ReverseSentence  Reverse Sentence of String1 is: “FAST-NU in OOP of student a am I” |

**Note: Read Strings 1, 2 and 3 from Data.txt (copy the data given below in your data file). Strings given in Data.txt are just samples. User can give any string in file (of 80 characters at max). Submit your data file along with the cpp file.**

**Data.txt**

|  |
| --- |
| //String 1  I am a student  //String 2  of OOP in FAST-NU |

**Q2: (Operator Overloading) You are given a cpp file “YourRollNumber.cpp” which contains partial definition of class “MyString” and a driver program Main. Your task is to complete the definition of MyString such that it gives following output:**



**Important:**

**You are not allowed to change main program.**

**YourRollNo.cpp**

|  |
| --- |
| #include<iostream>  using namespace std;  class MyString  {  //You can add your code here  private:  char\* str;  int lenght; //including the null char here  //Add GetStringFromBuffer as private member (helper)  //Add Concatenate as private member (helper)  //You can add your class members here  public:  //-------------DO\_NOT\_CHANGE REGION starts below---------------------  //Do not change the prototypes given below  MyString operator+(const MyString); //concatenate str  MyString& operator=(const MyString&); //Deep copy  MyString& operator++(); //Append a character ‘a’ at the end of str and increment length by 1.  MyString& operator--(); //remove last character of str and decrement length by 1.    MyString& operator-(const MyString&); //if (a – b) and length of a is greater than b then remove the string in a that is substring of b and also subtract length of a from b. let a.str = “hello” and b.str = “el” than (a-b).str = hlo, (b-a) would result in no change as b is smaller in length than a.    bool operator<(MyString); //Comparison on the basis of ascii values  //-------------End of DO\_NOT\_CHANGE REGION---------------------  //Add your class members here  };  //--------------------------Add your code here----------------------  //-------------DO\_NOT\_CHANGE REGION starts below---------------------  void main()  {  MyString str1, str2, str3, str4, str5; //Default constructor will make a string of lenght 1 having null character only i.e. empty string    if(!str1)  {  cout<<"String 1 is Empty.\n";  cout<<"Str 1 = "<<str1<<endl<<endl<<endl;  }    cout << "Enter String 1:\t";  cin >> str1;  cout << "Enter String 2:\t";  cin >> str2;    cout << "\n\n\nUser Entered:\n";  cout << "String 1 = " << str1 << endl;  cout << "String 2 = " << str2 << endl<<endl<<endl;  //What is following code testing?  cout<<"Before str1 = str1; str1 = "<<str1<<endl;  str1 = str1;  cout<<"After str1 = str1, str1 = "<<str1<<endl<<endl<<endl;  cout<<"Before str4 = str3 = str1+str2\n";  cout<<"str1 = "<<str1<<endl;  cout<<"str2 = "<<str2<<endl;  cout<<"str3 = "<<str3<<endl;  cout<<"str4 = "<<str4<<endl;  str4 = str3 = str1+str2;  cout<<"\n\n\nAfter str4 = str3 = str1+str2\n";  cout<<"str1 = "<<str1<<endl;  cout<<"str2 = "<<str2<<endl;  cout<<"str3 = "<<str3<<endl;  cout<<"str4 = "<<str4<<endl;  cout<<"\n\n\nEnter String 3:\t";  cin >> str3;  cout<<"\n\n\nEnter String 4:\t";  cin >> str4;    cout<<"\n\n\nstr3 = "<<str3<<endl;  cout<<"str4 = "<<str4<<endl;  if(str3 < str4)  cout<<"String 3 is Less than String 4.\n";  else  cout<<"String 3 is NOT Less than String 4.\n";  MyString str5 = str1 + str2;  cout << "\n\n\nStr5:\t" << str5 << endl;  cout << "Str5[7]:\t" << str5[7] << endl; //Function Call: str5.operator[](7).  str5[7] = '$';  cout << "\n\nStr5:\t" << str5 << endl;  cout << "\n\n\nstr5(5, 10):\t" << str5(5, 10) << endl;// Substring of lenght 10 starting from index 5 . Function Call str5.operator()(5,10) Let the returned MyString or char\* leak  // Output for below statements is going to be shown below the sample output given above  // Values right now -> str3 = abcd, str4 = abcD  cout << "\n\n ++Str3 :\t" << ++str3 << endl;  // ++Str3: abcda  str5 = str4 – str3;  cout << "\n\n Str4 – Str3 :\t" << str5 << endl;  // Str4 – Str3: // Nothing printed because str4 is less than str3  cout << "\n\n --Str3 :\t" << --str3 << endl;  // Str3 : abcd  cout << "\n\n --Str3 :\t" << --str3 << endl;  // Str3 : abc  str5 = str4 – str3;  cout << "\n\n Str4 – str3 :\t" << str5 << endl;  // Str4 – Str3: D ( abcD – abc = D )    }  //-------------End of DO\_NOT\_CHANGE REGION--------------------- |
|  |