TASK 1 REVIEW

C++ String Manipulation Program

Introduction
This is a C++ String Manipulation program that encountered some challenges during development, specifically with the "ChangeToLowerCase" function. After running this function, the program would show the original string but did not provide the lowercase value next to it as it should behave.

Solution
After some research, it was found that the ASCII table was built in such a way that a lowercase letter has the same value of its equivalent uppercase letter plus 32 (Vice-Versa -32). After modifying the lowercase function from -32 to +32 which is used to generate the lowercase value, the program now produces the lowercase value as it should behave.

However, another problem was found with the "ChangeToLowerCase" function, where the program interpreted a user input "space" to be an "@" symbol. This was due to the following line of code: "if(orinigalString[i]<=97" not being specific enough to specify the user's input 'space' in terms of the ASCII table. After changing the code to as follows: "if(orinigalString[i]<=90&&originalString[i]>65)" the program was able to interpret and output the user's input 'space' correctly.

From looking at the ASCII table, the program was debugged by defining the integers which should be greater or less than or equal to within the "ChangeToUpperCase" and "ChangeToLowerCase" functions.

Conclusion
The experience of developing this C++ String Manipulation program has highlighted the importance of thorough research and the use of other resources such as books to assist in understanding strings and how to manipulate them. This will enable further development of programming skills and knowledge to code at a reasonably high level.

Credits
This script was created by Anthony Constant (AC). If you have any questions or suggestions, you can contact him at https://anthonyconstant.co.uk/

License

Task 1: String Manipulation Challenge

https://replit.com/@Ant94x/String-Manipulation-C-Challenge

GitHub
Share Link https://github.com/Anthony-Constant/C-String-Manipulation-Challenge

TASK 1 C++ COPY & PASTED LOCAL SOURCE CODE

```
//*****************************************************************************************
//4COM1037 – C++  String Manipulation Challenge by AC
// *****************************************************************************************

#include <iostream> //std::cout
 //defines the standard devices cin, cout, clog, cerr; for more info see http://www.cplusplus.com/reference/iostream/

#include <string> //defines string types and conventions including begin & end iterator; for more info see
http://www.cplusplus.com/reference/string/


#include <algorithm> //std::shuffle
//defines a collection of ranged functions; for more info see http://www.cplusplus.com/reference/algorithm/
```

```
#include <random> //std::default_random_engine
//defines random number generation facilities; for more info see http://www.cplusplus.com/reference/random/

using namespace std; //tell the compiler that by default to use the "std" – this means that we don't need to keep saying
"std::cout" we can just use: "cout"

//CHANGELOG
// Template released 20/02/19 – User menu – working; Enter word/phrase – working; stores users input into variable and stores
it for later use; reverse word partially complete; Quit StringWorld – working; Show user their current word – working;

// 21.02.19 Fixed Reverse word/phrase. Reverses the users current word so it reads backwards to the user – tested – works;
// 21.02.19 Fixed some minor bugs; Changed order of user menu; Changed syntax within the user menu;
```

```
//21.02.19 Added variable 'AsciiWord'  - tested - works; Now generates ASCII value for each character within the string

// 22.02.19 Added varaible 'ChangeToUpperCase'; Changes users string input to ALL UpperCase characters
// 22.02.19 Added variable 'ChangeToLowerCase'; Changes users string input to ALL LowerCase characters

// 23.02.19 Fixed some minor bugs; Rephrased menu according to brief; SRN added to (current word/phrase is: ); Changed menu title according to brief: 'C++ String Manipulation Challenge'; Renamed Menu functions according to brief: 'GetWord', 'ReverseWord', "ToUpper", "ToLower" "RandomiseWord", "OrderWord", "QuitNow".
// 23.02.19 Added 'Default:' to Switch menu - If the user enters any non-valid option, the program will issue the following error message: Unfortunately, 'XX' is not a valid option, please try again

// 24.02.19 Fixing QuitNow and EXIT_SUCCESS - If the user enters 0 to quit it will ask the user "are you sure (Y/N?)' - If user enters "Y' it should terminate the program, If the user enters 'N' it should return the user back to the main menu.; CURRENTLY NOT WORKING

// 25.02.19 Latest update: attempting to return last word/phrase from each function to main menu

// Partially fixed - 'Unfortunately, won is not a valid option, please try again.' - it is taking the users currentString rather than their incorrect option.

// 05.03.19 Fixed QuitNow and EXIT_SUCCESS - When user enters 'O' at the menu option, the program executes 'QuitNow' function and asks user 'Are you sure you want to Quit? ( Say Yes or No)' if 'Yes' then terminate program if 'No' use return back to main menu. - invalid user input provides error message to user and returns back to main menu partially working;

// 05.03.19 Fixed return last word/phrase from each function to main menu - Now returns each word/phrase from each function i.e. user current word is 'hello' after the user selects menu option 3 the last word/phrase returned was: HELLO;

// 18.03.19 Fixed 'Enter a word or phrase' - user can now enter special characters, space between words etc. - fixed using getline (); and cin.ignore(); for user to enter any characters;

// 18.03.19 Added 'RandomiseWord'; if the user enters their word or phrase and chooses menu option '5' the program will randomise their current word stored in the localString;

//18.03.19 Fixed 'ReverseWord' - ReverseWord before would add the currentString with modifiedString and produce the reverse word backwards plus the original word - Now it reverses the word and only shows the user the current reversed word;
```

```
//19.03.19 - Added 'OrderWord' - sorts the current word in ascending alphabetic order;

//19.03.19 Fixed Invalid user input within menu option - if the user enters any non-valid option, the program will issue the following error message: Unfortunately, 'XX' is not a valid option, please try again. ( XX is replaced by user input);

//19.03.19 Fixed Invalid user input within QuitNow option - If the user enters any non-valid option, the program will issue the following error message: Invalid Input, Please try again.;

//26.03.19 Fixed some bugs - ToLower/ToUpper now prints to the user the following "the LowerCase for h == h" "the UpperCase for h == H"


    //MODIFY USER INTERACTION FOR 'ENTER A WORD OR PHRASE'
 //MENU OPTION = 1
string GetWord(void) { //this function should allow the user to enter a new word that can then be later manipulated

  string localString = ""; //declare and initialise a local string, used to store the new string

  cout << "Enter a new word or phrase: "; //output this message when user goes into menuOption1
  cin.ignore(); // ignore amount of characters specified
  getline (cin, localString); //declare string object that stores users word or phrase

  return localString; //pass the value stored in 'localstring' back to the calling instruction

}

   //MODIFY USER MENU QUIT OPTION HERE

void QuitNow(void) { //declare the void variable 'QuitNow'
string input; //declare string input for user to enter their option
cout << " Are you sure? (Type "Y", "Yes" or "y" or "N", "No", "n")"; //output this message to user
cin >> input; //initialise input
if (input == "Yes" ) { //if the user enters 'Yes'
  cout << "Thank you for using StringWorld - please come back soon." << endl; //send this message to the user

}
```

```
else if (input == "Y" ) { //if the user enters 'Y'
  cout << "Thank you for using StringWorld - please come back soon." << endl; //send this message to the user

}

else if (input == "y" ) { //if the user enters 'y'
  cout << "Thank you for using StringWorld - please come back soon." << endl; //send this message to the user

}

 else if (input == "No") { //if the user enters "No"
  cout << "\n You entered No, Return back to menuOption. \n";   //send this message to the user
 return; //return user back to menuOption
}

else if (input == "N") { //if the user enters 'N'
  cout << "\n You entered No, Return back to menuOption. \n";   //send this message to the user
 return; //return user back to menuOption
}

else if (input == "n") { //if the user enters 'n'
  cout << "\n You entered No, Return back to menuOption. \n";   //send this message to the user
 return; //return user back to menuOption
}

else {
  cout << "\n Invalid Input, Please try again. \n"; //if the user enters non-valid input send this message to user and return
back to menuOption
 return;
}

  exit(EXIT_SUCCESS); //execute the c++ exit function and pass it c++ contant value called EXIT_SUCCESS; EXIT_SUCCESS is
actually 0 - for more  info see: http://www.cplusplus.com/reference/cstdlib/exit/
}
```

```
    //MODIFY USER INTERACTION FOR 'ReverseWord'
//MENU OPTION = 2
string ReverseWord(string originalString) { //this function is designed to expect one aurgument (called originalString) and returns a string

  string localString; //declate a local variable called localstring


  int len; //declare a local variable called len of type integer (whole numbers)


  //some debug code - used to check the values
  cout << "Original String: " << originalString << " \n\n";
  len = originalString.length(); //execute a special function (part of the "string" datatype) that returns the number of character contined in the variable
  cout << "Total Length: " << len << " \n\n";

  //this block of code loops forward through the "originalString" variable

  for (int i = len; i >= 0; i--) {
    //this executes a loop that uses and decreases a local and temporary variable (i) staring with the value contained in 'len' while "i" is great than/equal to 0
    localString = localString + originalString[i]; //essentially this instruction adds the specific letter at position "i" in the string variable "originalString" to the 'localstring' variable
  }

    //at this point the 'localString' variable should now contain the same information as originalString
  cout << "now lets see what localString contains: " << localString << " \n\n";

  return localString; //return the value contained in the localstring local variable

}

//MODIFY USER INTERACTION FOR 'ChangeToUpperCase'
//MENU OPTION = 3
string ToUpper(string originalString) {
```

```
  int len; //create local variable for len
  len = originalString.length();
  string localString; //declare local variable 'localString'
  string modifiedString("Nothing"); //declare and initialise the variable to hold the last modified word

for (int i=0;i < len; i++) { //if i is less than len then increment ++
if(originalString[i]<=122&&originalString[i]>=90){ //if the orignialString[i] is less than or equal to 122 and the
orignialString[i] is more than or equal to 90
  originalString[i]=originalString[i]-32; // the program will subtract 32 from the current ASCII value which will output the
UpperCase version of the user's word stored in localString (it will -32 from each character in the array string)
  localString = originalString;} //at this point localString is equal to orignialString
  else //if the user's current word is already UpperCase execute this part of the program
  originalString[i]=originalString[i]; // if the user has UpperCase as their current word already, do nothing and output the
users word.
  localString = originalString; // the orignialString is stored into the localString

  cout << "the UpperCase for " <<originalString[i]<< " == "<<localString[i]<<"\n"; //the program will output this message to
show the user their UpperCase version of their current word stored in the localString
 }
return localString; // return the "ChangeToUpperCase" word to menuOption
}

//MODIFY USER INTERACTION FOR 'ChangeToLowerCase'
//MENU OPTION = 4

string ToLower (string originalString) {
  int len; //create local variable for len (integer)
  len = originalString.length();
  string localString; //declare local variable 'localString'
  string modifiedString("Nothing"); //declare and initialise the variable to hold the last modified word

for (int i=0;i < len; i++) { // if i is less than len then increment i (++)
  if(originalString[i]<=90&&originalString[i]>=65){ // if the orinialString[i] is less than or equal to 90 and the
originalString[i] is greater than or equal to 65
  originalString[i]=originalString[i]+32; // then add 32 to the ASCII value to get Lowercase value for users current word
stored in localString (it will +32 to each character in the array string)
```

```
  localString = originalString;} // the localString is equal to originalString here
  else //if the user's current word is already lowercase execute this part of the program
  originalString[i]=originalString[i]; //the users originalString[i] will stay as originalString[i]
  localString = originalString; //the originalString is stored into the localString

  cout << "the LowerCase for "<< originalString[i]<<" ==  "<<localString[i]<<"\n"; //the program will output this message to
show the user their LowerCase version of their current word currently stored in the localString
}
return localString; //return "ChangeToLowerCase" word to menuOption

}

//MODIFY USER INTERACTION FOR 'RandomiseWord'
//MENU OPTION = 5

string RandomiseWord (string orignialString) { //declare the variable RandomiseWord
string localString; //declare the variable localString
random_shuffle(orignialString.begin(), orignialString.end()); //rearrange the string elements in the range first to last
randomly
localString=orignialString; //the orignialString is stored into the localString here

return localString; //return "RandomiseWord" word back to menuOption

}

//MODIFY USER INTERACTION FOR 'OrderWord'
//MENU OPTION = 6

string OrderWord (string originalString) { //declare the string variable OrderWord
  string localString; //declare the local variable localString
originalString = ToLower(originalString); // store ToLower(originalString) in the originalString
sort (originalString.begin(), originalString.end()); //sort the elements in range first to last into ascending order
std::sort

localString = originalString; //store the originalString into the localString variable
```

```
return localString; //return "SortWord" back to menuOption
}


 // MODIFY USER MENU HERE
void Menu(void) { //this function displays the menu to the user

  string currentString("Hello"); //declare and initialise the variable to hold the current word ready to be manipulated

    string localString;

  string modifiedString("Nothing"); //declare and initialise the variable to hold the last modified word

  int menuOption; //declare a variable to hold the user's menu choice

  do { //start an endless loop - this is so the associated code block repeats forever

  //Modify menuOption here
    cout << "\n\n Welcome to C++ String Manipulation Challenge      \n\n"  << endl; //display the title for the menu
    cout << "1. Enter a word or phrase (current word/phrase is: " << currentString << "). \n" << endl; //display "case 1"
      cout << "2. Reverse the current word. \n" << endl; //display "case 2"
     cout << "3. Convert the current word to uppercase. \n" << endl; //display "case 3"
    cout << "4. Convert the current word to lowercase. \n" << endl; //display 'case 4'
    cout << "5. Randomise the letters in the current word. \n" <<
    endl; //display "case 5"
   cout << "6. Sort the letters in the current word in ascending alphabetic order. \n" << endl; //display "case 6"
    cout << "0. Quit the program. \n" << endl; //display 'QuitNow'
    cout << "\nThe last word/phrase returned was: " <<   modifiedString << endl; //used for returning the localString
    cout << "\nPlease enter a valid option (1 - 6 or 0 to quit): ";

    cin >> menuOption; //store the choice made by the user in the variable menuOption

    switch(menuOption) { //a 'switch' statement is similar to multiple if's; see
https://en.cppreference.com/w/cpp/language/switch

      case 1: //this basically "if the menuOption = 1", option 1 should call the function GetWord
```

```
        currentString = GetWord(); //call the function and store any returned value in the local variable currentString
        break; //this keyword is optional, however, if not used all the following "cases" are automatically true and the
instructions will all be executed

        case 2: //this basically "if the menuOption = 2", option 2 should call the function ReverseWord
        modifiedString = ReverseWord(currentString);
        break;

            case 3: //this basically "if the menuOption = 3", option 3 should call the function ToUpper
        modifiedString = ToUpper(currentString);
        break;

              case 4: //this basically "if the menuOption = 4", option 4 should call the function ToLower
        modifiedString = ToLower(currentString);
        break;

          case 5: //this basically "if the menuOption = 4", option 4 should call the function ToLower
        modifiedString = RandomiseWord(currentString);
        break;

        case 6: //this basically "if the menuOption = 4", option 4 should call the function ToLower
        modifiedString = OrderWord(currentString);
        break;



        break; //this keyword is optional, however, if not used all the following "cases" are automatically true and the
instructions will all be executed

      case 0: //this basically "if the menuOption = 0", option 0 should call the function QuitNow
        QuitNow();
        break;

      default:
      cout << " \n Unfortunately, " << menuOption << " is not a valid option please try again. \n "; //this will produce an
alert if their is invalid user input.
```

```
    }
  } while(true); //this tells the computer the conditions to break the "do" loop - while true is always true and therefore
will continue forever; in essence and endless loop
}
//END OF USER MENU

int main() { //this is the required function that will automatically execute when the program is run
  Menu(); //start the program menu function
}
```

// TEST LOG - Test Date, Use & Operatins No (1-6), Description, What did you test, Results, Reflections

// 21.02.19 Adding "AsciiWord" function; Generates the ASCII value for each character within a string; tested; Results: Shows user their current word and total word length. Generates the ASCII value for each character in the users word/phrase which is exposed to the user. Also by generating the Ascii value for characters&strings assisted in debugging the majority of functions such as ChangeToUpperCase, ChangeToLowerCase to get specific results and match it with the Ascii table; PASS;

// 22.02.19 Adding 'ChangeToUpperCase" function; Converts a string which is LowerCase to UpperCase otherwise, it stays the same. tested; Results: Shows the user their original string equal to their Generated UpperCase character and returns user back to main menu; PASS;

// 25.02.19 Adding "ChangeToLowerCase" function; The programme should Convert a string which is UpperCase to LowerCase otherwise, it stays the same. tested; Results: Shows the user their original string equal to their supposed Generated LowerCase character however, it shows nothing; FAIL; test(1); test(2) Changed -32 from the ASCII library to +32 which is used to generate the LowerCase value; tested; PASS; test (3) the programme considers when the user enters 'space' to be '@' symbol. FAIL; before line '185' had the following code: "if(originalString[i]<=97)" which means if the Ascii character is <= to 97 +32 to the Ascii value to generate the lowercase character. I have modified the code as follows: 'if(originalString[i]<=90&&originalString[i]>=65)' which means if the Ascii value is <= 90 and >= 65 then +32 to the Ascii value (including symbols). The programme now shows the user their original string equal to their generated lowercase word

including 'space' and the programme now correctly reads it as 'space' then stores and returns the word back to localString. PASS;

// 10.03.19 Adding 'RandomiseWord' function; this function randomises the current user's word. for example if provided with a string such as "Hello World" it will generate a word such as "odorH leWll" tested; Results: the programme prints to the user their 'RandomiseWord'and return it back to localString to be viewed in the void 'menuOption'. PASS;

//13.03.29 Adding 'OrderWord' function; this function sorts the current word in ascending alphabetic order; tested; Results: the programme prints to the user their 'OrderWord' for example, the user's current world is "Hello World", once the OrderWord function has been executed it will return back to the localString "dllHo eoWlr" PASS;

//17.02.19 Adding QuitNow() function; this function once selected will allow the user to change their mind using "are you sure? " option. if the user is sure then the program will quit otherwise, the user will be returned back to the menuOption and select another option from here. tested; Results: once in the QuitNow function it will ask the user Are you Sure? (Type 'Y', 'yes' or 'y' or 'N', 'No', 'n') if the user enters either Y, Yes or y it will terminate the programme and print the following message: "Thank you for using StringWorld - please come back soon." If the user enters N, No or n it will return the user back to the menuOption. However, if the user enters anything else other than these options it will print out the following message "Invalid Input, Please try again." and return the user back to the menuOption.