**CS 200: Introduction to Programming**

Dr. Naveed Arshad

Muhammad Kamran Nishat

Assignment 3  
(Due Date: Saturday, October 25, 2014)



This assignment is due on Saturday, 25th October. The usual late submission policy of 10% per day deduction for up to 3 days applies. The assignment needs to be submitted on LMS under the ‘Assignments’ tab.

The course policy about plagiarism is as follows:

1. Students must not share actual program code with other students.
2. Students must be prepared to explain any program code they submit.
3. Students cannot copy code from the Internet.
4. Students must indicate with their submission any assistance they received.
5. All submissions are subject to automated plagiarism detection.

Students are strongly advised that any act of plagiarism will be reported to the Disciplinary Committee.

Welcome to your first C++ programming homework assignment. Please read this handout very thoroughly before starting your assignment. You will be seeing a lot of new terminology along the way. The information in this handout will help you understand any new terms you might encounter. Also, please go through the submission details carefully, so that you don’t lose points due to any inconsistencies in your assignment submission.

* You only need to submit one file called *asg3\_rollnumber*.rar (compressed file), where you should replace *rollnumber* with your actual roll number, e.g. asg3\_15100200.rar. All assignment submissions will be done on LMS.
* All forms of plagiarism, such as the following, will be dealt in the harshest of manner:
  + Typing code for a fellow student,
  + Debugging code for a fellow student,
  + Having a fellow student type or debug code for you,
  + Sharing your code with others via any means,
  + Acquiring a copy of another student's code without his/her knowledge,
  + Deliberately copying off another student's code from his/her monitor,
  + Deliberately letting another student copy off code from your monitor,

Greetings from the people who sent Pacman your way! We sure hope you enjoyed programming the carnivorous monster because we have another bunch of interesting problems coming your way in this assignment. It’s time to really dig deep into C++ and get all the nitty grittys straight and unleash the coder within you! In order to become a great wizard err programmer, you need to learn the basic tricks of the trade. Now without further ado, we present to you CS200 Assignment 3: Of Sorting, Strings and Cryptography. The assignment is geared towards improving your understanding of functions, loops, arrays, indexing, essentially everything that you have studied in class so far! ☺

Now let’s get started!

**Part 1: Sorting the Array**

Sorting is simply taking an array and arranging its elements so that they follow some sort of particular order. There are many ways to sort an array, and you will probably study this well into your sophomore year. So we might as well introduce you to some of the methods early :).

**Task 0:**

for(int x=0; x<n; x++)

{

for(int y=0; y<n-1; y++)

{

if(array[y]>array[y+1])

{

int temp = array[y+1];

array[y+1] = array[y];

array[y] = temp;

}

}

}

This is an basic version of the algorithm for a sorting known as BubbleSort. Look at these nested for loops. Try to see what they are doing if you pass an array[] to them. Understand this code properly. It is very pointless to go on from here if you do not know exactly what is happening. In this task simply implement the algorithm and in comments, explain what this algorithm is doing.

Now that we’ve warmed you up, lets proceed to the actual assignment. The first real task: implementing a simple sorting algorithm called the Gnome Sort. If by any chance you began to think of Enid Blyton novels you read as a kid, you’re gravely mistaken, this has little to do with fairies, pixies and well, even gnomes. Makes you wonder what kind of drug people were on while naming the algorithm, right? The sort basically goes like this, in your array, you iterate through and look at an example

position i.e. **arr[pos]** and **arr[pos-1]**. If **arr[pos]** is lesser than **arr[pos-1]** then you swap. As you iterate through,you’ll end up sorting the whole array.... right? If you think the array is sorted just by doing that , well sorry to say but you are wrong. After the swap you also have to decide if you have to iterate forward or backward in the array! Explain your code in comments!

**Task 1:** Figure out how to iterate back and forth the array and thus implement gnome sort ( in this case, sorting is in ascending order) **[15 points]**

**Task 2:** Edit the program to sort it in descending order now **[10 points]**

**Part 2 : C String library**

Here are the functions that you need to implement: Hopefully by now you have a lucid understanding of what libraries are and have grown accustomed to using them. In this assignment you will be implementing your very own myString library. This library can be seen as a not so fancy twin of the actual c++ string library. The c++ string library contains many functions that allow manipulation of strings such as calculating the length of the string, finding a substring within a string etc. We have handpicked a few such functions and mixed them with some of our own that you will now have to implement.

For the next few tasks, you will be using character arrays that you have done in class. For your convenience, you can use a ‘\0’ character as the last element of your character array to mark the end of the word. This will also come in handy when you deal with other sorts of strings in C++ later on in the course.

**Task 3: [5 points]**

**int StringLen(char stringinput[])**

This function calculates the length of the string that is given to it and returns the total number of characters in it.

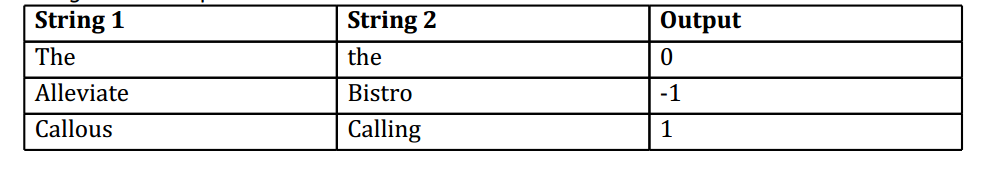
e.g. string input: **IDIOSYNCRASY**

sample **output: 12**

**Task 4: [5 points]**

**int compareString (char string1[], char string2[])**

Your compareString function should take in two character arrays from the main function and return an integer. The function should compare the lowercase version of the two strings character by character. So, if “The” is compared with “the”, they are to be treated as identical. If all characters in the two strings are identical, the function should return a 0. It should return1 if the first character that does not match has a greater value in string1 than in string2; and should return -1 if the first character that does not match has a lesser value in string1 than in string2. It’s time for an example!!

The following statement is an example of how comparisons can be made to determine whether a character is lesser than the other:

if (character1<character2)

{

cout<< “true”;

}

This code will compare the ASCII values of the two characters and will print “true” if the ASCII value corresponding to character1 is lesser than the ASCII value corresponding to character2, otherwise do nothing.

**Task 5: [5 points]**

**void shiftString(int n, char stringinput[])**

This tasks takes as input a word and an integer shift amount. Assume that the integer amount is less than the length of the word. Your task is to take each letter in the word and replace it by the letter that comes after n hops in the word.

e.g. n = -3

input given = Whale

new string = leWha

The integer can be negative or positive. If negative, you perform a shift leftwards and vice versa if integer is positive. Do notice that this is a void function so it will not return anything. Therefore, you need to put your knowledge of passing by reference to use and change the original string to the shifted string.

**Task 6:** **[5 points]**

**void stringCase(char stringinput[], bool choice)**

This task involves taking a word as input and a boolean variable: ‘choice’. The function should convert the word to uppercase if the choice is 1 and to lowercase if the choice is 0.

e.g.

word = Apples

choice = 1

new word = APPLES

word = LOyALTy

choice = 0

new word = loyalty

**Part 3 Crossword Puzzles**

Write a program to search for words in the given puzzle: a two-dimensional array of letters read from a file (e.g. puzzle.txt). For example, you might be given the following array:

m s j l i x a p e

j h b x e e n p p

h o t t h b s w y

r o a i n u y z h

p t f x r d z k q

t p n l q o y j y

a n h a p f g b g

h x m s h w y l y

u j f j h r s o a

Your program should ask the user for a word he wants to search. If the word is found, display the location of the word and the direction the word from that point. You have to search for the words in horizontal and vertical direction only. **[40 points]**

**An example run of the program:**

Please enter the word you want to find: shoot

Result: Word found at location 0,1 in downward direction.

Do you want to continue searching: y

Please enter the word you want to find: syzygy

Result: Word found at location 2,6 in downward direction.

Do you want to continue searching: y

Please enter the word you want to find: hello

Result: Word not found.

Do you want to continue searching: n

Good Bye.

**Part 4 Tic Tac More**

You may have played tic-tac-toe, where the object is to get 3 in a row.  This is a slight extension, where the object is to get 4 in a row OR to get 3 in a row along one of the edges.  The first part of the game looks like:

Welcome to TicTacMore

Author: Adil Ahmed

CS 200, program #4. Written in C++ on Mars Server.

The object of the game is to get either 4 in a row or

3 in a row on any edge. Player X starts and play alternates

until someone wins or the board is full. You may only move

in a square adjacent to a piece already played.

1 2 3 4 5

A . . . . . A

B . . . . . B

C . . . . . C

D . . . . . D

E . . . . . E

1 2 3 4 5

1. Enter row and column for move (e.g. b3) -> c3

1 2 3 4 5

A . . . . . A

B . . . . . B

C . . X . . C

D . . . . . D

E . . . . . E

1 2 3 4 5

2. Enter row and column for move (e.g. b3) -> b2

1 2 3 4 5

A . . . . . A

B . O . . . B

C . . X . . C

D . . . . . D

E . . . . . E

1 2 3 4 5

3. Enter row and column for move (e.g. b3) -> x

Exiting program...

Thanks for playing. Exiting...

Consider writing the program a step at a time, using the following order. The number of points for each item are shown: **[35 points]**

1. Get the instructions to display. Declare and initialize the variables used to represent the board. Write the method to display the board **[5 points]**
2. Write the method to get user input, keeping track of the move number. Then write the method that modifies the board, based on the user input. On odd numbered moves it is the turn of 'X' to play. On even numbered moves it is the turn of 'O' to play. Put this code in a loop so multiple moves can be made. When the user input is 'x' then give a message and exit the game. **[10 points]**
3. Write the method that verifies each move is adjacent to at least one existing piece on the board. The one exception to this is that the first move of 'X' can be anywhere on the board. **[10 points]**
4. Write the method to check for a win of 3 in a row along one of the edges. Give a message, display the final board configuration and exit the program if found. **[5 points]**
5. Write the method to check for a win of 4 in a row anywhere else on the board. Give a message, display the final board configuration and exit the program if found. **[5 points]**

**Tips and Resources:**

When faced with a problem, never be afraid to go to Google. You will soon learn soon (if you haven’t already!) that Google will be your best friend for most of your LUMS life. A few good tutorials and examples can be found at:

<http://www.cplusplus.com/>

<http://www.fredosaurus.com/notes-cpp/>

When Googling, do look at books as well. Sometimes a good book can be way more helpful than a collection of online pages.

Use the CS200 Forum on Piazza, helping others can really improve your own concepts. Visit the tutorials and TA office hours and start the assignment on time for smooth sailing!

When debugging, it can always be helpful to print out the various variables and parameters that you are using onto the screen. Indent your code beautifully, it can help save you from making various blunders while programming.

Note: Although you are urged to Google, you should never copy any code. However, if you do we’ll have no other option but to forward you to the guillotine or more formally, the Disciplinary Committee.

Have a Happy Eid with a Coding add-on!! ☺