CSCE 2004 – Programming Project 3
Midpoint Due Date – Feb 22, 2017 at 11:59pm
Final Due Date – Mar 1, 2017 at 11:59pm

1. Problem Statement:

The goal of this programming assignment is to use iteration and condition statements to create some very simple ASCII art. Your program should prompt the user to enter two pieces of information: (1) the size of the pattern, and (2) the ASCII character to print when making the pattern. Then your program should output the following five patterns on the screen:

Solid Square Pattern

```
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
```

Solid Triangle Pattern

```
*
* *
* * *
* * * *
* * * * *
```

Vertical Stripe Pattern

```
*   *   *
*   *   *
*   *   *
*   *   *
*   *   *
```

Square Outline Pattern

```
* * * * *
*       *
*       *
*       *
* * * * *
```

Letter X Pattern

```
*       *
  *   *
    *
  *   *
*       *
```

The five patterns above were made with size=5 and character='*'. Your program should work for any input size in the range [5..25]. If the user enters a value less than 5, your program should convert it into a 5 and continue. Similarly, if the user enters a value greater than 25, your program should convert it into a 25 and continue. Your program should work with any printable character. If you used spaces or tabs, the patterns above would be invisible, and that would not be very rewarding.

2. Design:

Try to design the code to be as simple as possible. Start by making a solid square of a given size. This can be done by looping over the rows and the columns of the square and printing an asterisk for each (r,c) value. You may also want to print a space character in between output characters so that the width and height of the square look the same.

To create the other output patterns, you need to think about the (r,c) values where you want to print a character, and the (r,c) values where you do not want to print a character, and design your condition statements accordingly. For example, to print out just the first column of the square outline, you want to output a character whenever (r==0) is true. Some patterns are easier to design and implement than others, so you should start with the easy patterns and then work up to the more complex patterns.

3. Implementation:

Since you are starting with a "blank piece of paper" to implement this project, it is very important to develop your code incrementally writing comments, adding code, compiling, debugging, a little bit at a time. This way, you always have a program that "does something" even if it is not complete.

As a first step, start with an empty main function, and add the code to print the initial messages to the user, read the user's input, and then print the input values back out again. Once this part is working, you can start printing the output patterns one at a time to your program.

4. Testing:

Test your program to check that it operates correctly for all of the requirements listed above. To do this, you should try creating patterns with a variety of sizes and a variety of output characters. When you create patterns with even sizes you may notice that they do not look as perfect as patterns with odd sizes. There is not much you can do about this. Just include an example of each in your project report.

You should add some simple error checking in this program to make sure the pattern sizes are in the range [5..25]. You should show examples showing what happens when the user enters a value less than 5 or greater than 25. Don't worry if they type in something silly like "hello mom" instead of an integer size. You should copy/paste your testing results into your project report to document what your program does in these cases.

5. Documentation:

When you have completed your C++ program, write a short report using the "Programming Project Report Template" describing what the objectives were, what you did, and the status of the program. Does it work properly for all test cases? Are there any known problems? Save this project report in a separate document (in docx format or pdf format) to be uploaded into blackboard with your code..

6. Midpoint Project Submission:

To encourage students to get an early start on their programming project, students are required to upload into Blackboard a partial solution to their programming project on the midpoint due date shown above. The program does not need to be complete, but it must compile and perform some of the tasks listed above. This midpoint solution is worth 10% of your final grade on the project.

7. Final Project Submission:

In this class, we will be using Blackboard for all project submissions to make sure that all students hand their programming projects on time, and to perform automatic plagiarism analysis of all programs that are submitted.

When you have completed all of the tasks listed above, go to Blackboard to upload your documentation (a single docx or pdf file), and your C++ program (a single cpp or txt file). Do NOT upload an executable version of your program.

The dates on your electronic submission will be used to verify that you met the due date above. All late projects will receive reduced credit:

10% off if less than 1 day late,
20% off if less than 2 days late,

30% off if less than 3 days late,
no credit if more than 3 days late.

You will receive partial credit for all programs that compile even if they do not meet all program requirements, so handing projects in on time is highly recommended.

8. Academic Honesty Statement:

Students are expected to submit their own work on all programming projects, unless group projects have been explicitly assigned. Students are NOT allowed to distribute code to each other, or copy code from another individual or website. Students ARE allowed to use any materials on the class website, or in the textbook, or ask the instructor and/or GTAs for assistance.

This course will be using highly effective program comparison software to calculate the similarity of all programs to each other, and to homework assignments from previous semesters. Please do not be tempted to plagiarize from another student.

Violations of the policies above will be reported to the Provost's office and may result in a ZERO on the programming project, an F in the class, or suspension from the university, depending on the severity of the violation and any history of prior violations.