Objective

The purpose of this assignment is to test your familiarity with C++ control structures (selection and repetition). You’ll also need to do fundamental algorithmic error checking. These programs have to be tested on linprog before they are turned in.

Email your 2 files letters.cpp and primes.cpp to jayarama@cs.fsu.edu

Program 1 - Printing Letters

ZAN, the great overlord of all things overlordy has a penchant for labeling things that belong to him. However, he believes his time would be better spent in acquiring things to label, and decides to delegate the menial task of inscribing his name, to you. In the same vein, you decide to write a program to perform this repetitive task. Write a C++ program to print the letters Z, A and N in ASCII art. Make sure your program conforms to the following requirements:

1. Accept the size of the letter (in the number of lines) from the user. This number should be an odd number greater than or equal to 5. If the value entered is invalid, tell the user so, and ask for another one. Repeat until you get a valid size. (6 points)

2. Accept the letter to be printed from the user. If the letter is Z, A or N, go to the next step. If not, tell the user that the letter is invalid, and ask for another one. Repeat until you get a valid letter. (6 points)

3. Print the letter in ASCII art form. The sample run gives examples for each letter. (30 points, 10 per letter)

4. Repeat the entire process if the user indicates they wish to continue. (4 points)

5. Make sure you add comments to explain the logic. (4 points)

Sample Run

Welcome to the letter printer.
Enter the size: 3
Invalid size. Enter the size again: -4
Invalid size. Enter the size again: 7

Enter the letter: D
Invalid letter: Enter the letter again: #
Invalid letter: Enter the letter again: N
Would you like to continue? (Y or N): Y

Enter the size: 8
Invalid size. Enter the size again: 5

Enter the letter: Z

*****
* *
* *
*****

Would you like to continue? (Y or N): Y

Enter the size: 9

Enter the letter: A

* *
* *
* *
********
* *
* *
* *

Would you like to continue? (Y or N): N

3 Prime Numbers

Zan is very impressed with your work. He has another task for you, upon completion of which he will reward you with more money than you can imagine (and you can imagine a lot). Zan has now, in his infinite wisdom, decided that all under his purview will only count in prime numbers. He will randomly yell out a number (which need not be prime. Note: Zan is not under Zan’s purview) and you need to show him all the prime numbers that come before it.

Write a C++ program to print all the prime numbers below a certain given number. A prime number is defined as a number that can only be divided by 1 and itself. For example, 2, 3, 17 and 41 are all prime numbers. Make sure your program conforms to the following requirements:

1. Accept the upper limit from the user (as an integer). (5 points)
2. Make sure the number is positive. If it is not, terminate the program. (5 points).

3. Go from 1 to the number. If you happen to find a number that is prime, print it. (35 points)

4. Add comments wherever necessary. (5 points)

3.1 Sample Runs

There are 3 sample runs here. (It’s OK if you have an extra comma at the end)

Enter the upper limit: 25
The prime numbers are:
2, 3, 5, 7, 11, 13, 17, 19, 23

Enter the upper limit: 70
The prime numbers are:
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67

Enter the upper limit: -7
Number is negative. Goodbye.

4 Generic Guidelines

1. Please make sure you’re only using the concepts already discussed in class. That is, please try and restrict yourself to input/output statements, variables and selection statements and loops. Do not use arrays.

2. Each program is worth 50 points.

3. Please make sure that you’re conforming to specifications (program name, print statements, expected inputs and outputs etc.).

4. Please make sure your code is readable.

5. Please make sure you’ve compiled and run your program before you turn it in. Compilation errors can be quite costly.