Programming Assignment
4:Triangles
Due Saturday, September 30th at 11:59 pm

1. Overview

The purpose of this assignment is to give you experience with C concepts including loops, conditionals, and logical expressions and with input validation and command-line arguments.

Also, this project is an exercise in planning (i.e., writing algorithms) as much as it is an exercise in C programming. For this assignment, you will write several versions of a program that prints a triangle or triangles to the screen.

Program 4a

The first version (p4a) will print to the screen a right triangle with a user-specified number of rows, as seen in the figure below. The number of rows will be limited by the width of the screen (you will do range-checking) and the triangle will be composed of the 'a' character. We will assume a width of 80 characters for the screen.

Here's one example run:

```
etkraem@joey12:./p4a
This program will print a right triangle of 'a' characters.
The characters will appear in the lower left of the diagonal.
Enter the number of rows (1-80): 5
a
aa
aaa
aaaa
aaaaa
```

Here's another example run, but this time the user enters a few wrong values for the number of rows. Notice how the program keeps prompting the user until a valid value is entered.

```
etkraem@joey12:./p4a
This program will print a right triangle of 'a' characters.
The characters will appear in the lower left of the diagonal.
Enter the number of rows (1-80): 90
Enter the number of rows (1-80): 88
Enter the number of rows (1-80): 0
Enter the number of rows (1-80): 6
a
aa
aaa
aaaa
aaaaa
aaaaaa
```

Program 4b

The next version, p4b, will also print a right triangle, this time composed of 'b' characters that fill the upper right diagonal. This program will accept the number of rows as a **command line parameter**. Again, the number of rows must be in the range 1.. 80. If the number of rows is missing or out of range, the program will display a "usage" message and exit. **The triangle should display to standard output. Usage messages should display to standard error.** See a few sample runs below:

User fails to provide the number of rows:

```
etkraem@joey12:./p4b
usage: ./p4b num_rows
       num_rows in range 1..80
```

User specifies a number of rows but it is out of range:

```
etkraem@joey12:./p4b 0
usage: ./p4b num_rows
       num_rows in range 1..80
```

User specifies a number of rows within the valid range:

```
etkraem@joey12:./p4b 5
bbbbb
 bbbb
  bbb
   bb
    b
```

Program p4c

The next version, p4c, will also print a right triangle, this time composed of 'c' characters that fill the upper left diagonal. We again use command line parameters and validate the arguments, with triangle displays going to standard output and usage messages to standard error.

Example 1:

```
etkraem@joey12:./p4c
usage: ./p4c num_rows
       num_rows in range 1..80
```

Example 2:

```
etkraem@joey12:./p4c 99
usage: ./p4c num_rows
       num_rows in range 1..80
```

Example 3:

```
etkraem@joey12:./p4c 7
ccccccc
cccccc
ccccc
cccc
ccc
cc
c
```

Program p4d

The next version, p4d, will also print a right triangle, this time composed of 'd' characters that fill the lower right diagonal. We again use command line parameters and validate the arguments, with triangle displays going to standard output and usage messages to standard error.

Example 1:

```
etkraem@joey12:./p4d
usage: ./p4c num_rows
       num_rows in range 1..80
```

Example 2:

```
etkraem@joey12:./p4d 99
usage: ./p4c num_rows
       num_rows in range 1..80
```

Example 3:

```
etkraem@joey12:./p4d 7
      d
     dd
    ddd
   dddd
  ddddd
 dddddd
ddddddd
```

Program p4e

Program p4e, will print a square that appears to be composed of 4 triangles (a, b, c and d) and a center 'd'. It will again use command line parameters and in addition to checking for a number of rows in the range 1..80, it will also only accept odd numbers of rows. The square will display to standard output. Any usage messages should display to standard error.

Example 1:

```
etkraem@joey12:./p4e 7
a b b b b b d
a a b b b d d
a a a b d d d
a a a * d d d
a a a c d d d
a a c c c d d
a c c c c c d
```

Example 2:

```
etkraem@joey12:./p4e 11
a b b b b b b b b b d
a a b b b b b b b d d
a a a b b b b b d d d
a a a a b b b d d d d
a a a a a b d d d d d
a a a a a * d d d d d
a a a a a c d d d d d
a a a a c c c d d d d
a a a c c c c c d d d
a a c c c c c c c d d
a c c c c c c c c c d
```

Example 3:

```
etkraem@joey12:./p4e 10
usage: ./p4e num_rows
       num_rows is odd in range 1..79
```

Example 4:

```
etkraem@joey12:./p4e 0
usage: ./p4e num_rows
       num_rows is odd in range 1..79
```

Example 5:

```
etkraem@joey12:./p4e
usage: ./p4e num_rows
       num_rows is odd in range 1..79
```

A few things to note:

1. You should start with some graph paper, and plan out your approach, the variables you will need, and what type of loop or loops you will use.
2. Use simple test cases at first (within range). Then test that you do not accept inputs from the user with a number of rows that is too large, or that are even or odd.
3. Later, test using the supplied test cases and compare your outputs to the supplied outputs. Recall that the provided scripts generate output files with the names of the supplied outputs. You'll need to modify the scripts to generate output files with different names. You might also write your own script to "diff" them. The Gradescope autograder will test similar cases.

2. What to Turn In

Your programs should be named `p4a.c, p4b.c, p4c.c, p4d.c and p4e.c` and should be submitted through the Gradescope link (coming soon).

The code will be evaluated not only on its correctness but also on its adherence to the coding style standards. **Remember to use meaningful variable names, and to indent and comment properly.**

3. Notes on Collaboration

You are required to work individually on this assignment. **Please do not consult *anyone* other than your instructor or the TA on *any* aspect of this assignment**.

Review the section on "Academic Integrity" on the Course Syllabus for the policy on what happens when copying or sharing code.

For any of the programming assignments in lecture and/or lab, it is considered cheating to do any of the following:

- discuss in detail the C code in your program with another student
- show or share the C code in your program with another student
- use C code obtained from another student, or any other unauthorized source, either modified or unmodified (each student is responsible for protecting his or her files from access by others)
- use re-engineering tools
- submit work of others, from the Internet or any other source, even if attributing the work to others

4. Grading Rubric

p4a:

____ / 05	input verification
____ / 05	functionality / correct output

p4b:

____ / 05	command line paramaters and input verification
____ / 05	functionality / correct output

p4c:

____ / 05	input verification
____ / 10	functionality / correct output

p4d:

____ / 05	input verification
____ / 10	functionality / correct output

p4e:

____ / 10	input verification
____ / 25	functionality / correct output

Overall:

___ / 10	formatting of code (header comment + other comments, indentation, meaningful variable names)
___ / 5	clean submission (only the needed files, no warnings)

____ / 100