SE-4831: Software Quality Assurance
Lab 9: ASCII Art and N Version programming

1. Introduction

N-Version programming is a technique that is often used to produce reliable systems under less than optimal specifications.

With N-version programming, independent development teams are provided a specification, and they are required to develop an independent implementation of the software. After N systems are developed, they are each executed in parallel, and a voting mechanism compares their outputs to generate a "true" program output.

N-version programming has been applied to software in switching trains, performing flight control computations on modern airliners, electronic voting, optical scan recognition, and other similar systems.

The key concept behind N-version programming is that independent development teams will make independent mistakes. Thus, if one team makes a mistake, one of the other teams is not likely to make the same mistake, and by comparing the results of N versions, mistakes can be overcome. This diversity also can improve system security as well, in that a security fault in one implementation may or may not be in another implementation.

2. Lab Activity

The full implementation of an N-version system is beyond the scope of this class. However, we are going to perform a trivial demonstration of the task this week through the construction of a simple ASCII art rendering program.

In whatever language you desire, and using whatever techniques you desire, you will be responsible for developing a console program which will translate an image description into an ASCII art file. This program, referred to as the rendering, must run in command line format under Windows and accept parameters passed on the command line.

3. Program specification

3.1. Command Line Invocation

When your program is to be executed, the program shall receive two command line parameter. The first parameter represents the name of the ASCII description file that you are to read. The second parameter represents the output file into which your ASCII art is to be placed.

For example, if you develop in Java, the command line might be:

java –jar myProgram.jar input.txt output.txt

If you develop in C, the command line would be

myProgram.exe input.txt output.txt

3.2. ASCII Description File

The ASCII description file is a text file which describes the ASCII art that is being drawn. Each new line indicates a new command. **Command parameters are tab delimited**. Blank lines shall be ignored, and invalid lines shall be ignored.

The ASCII description file shall have two fundamental commands, SIZE and LINE.

3.2.1. SIZE command

The size command shall set the dimensions for the ASCII canvas. The first parameter represents the width of the canvas, and the second parameter represents the height of the canvas. For example, the command:

SIZE 100 50

will create a canvas which is 100 characters wide by 50 characters tall. Parameters must be integers.

Only the first valid size command in a file shall be accepted. Any subsequent SIZE parameters shall be ignored.

3.2.2. LINE Command

The line command shall draw a line on the canvas between two points, (x1, y1) and (x2, y2). In drawing the line, the given character shall be assigned the text '*' if the line crosses that location. Otherwise, the character shall be stored as a space.

For example, the command

LINE 10 10 20 10

shall draw a line between the points (10, 10) and (20, 10).

For the purposes of this assignment, the upper left hand corner of the canvas shall be assigned the coordinate (0,0). X represents the horizontal distance and y represents the vertical distance.

```
SIZE  60    50
LINE  10    20    10    40
LINE  10    40    50    40
LINE  50    40    50    20
LINE  50    20    30    10
LINE  30    10    10    20
```

Figure 1: Sample image description file.

3.3. Output File

The output file that is written from this program shall consist of exactly y lines, where y represents the height of the canvas. Each line shall consist of exactly x spaces, asterisks, or combinations thereof, where x represents the width of the canvas. Each line shall end with a single newline character / carriage return.

Aside from those three characters, no other values shall be written into the output file.

3.4. Example Execution

Figure 1 provides a sample image definition file. The file sets the size of the canvas to 60 by 60, and draws 5 lines on the canvas. A sample rendering of this image is given in Figure 2.

4. Lab Deliverables

By Friday, February 21, 2014, you shall upload the following to the course page:

1. A completed program executable of your image render. If developed in Java, this is a executable jar file which can be readily executed by issuing the command java –jar Filename input.txt output.txt If developed in another language, this is a command line executable file which can be invoked from the windows shell.

If you have any questions, consult your instructor.

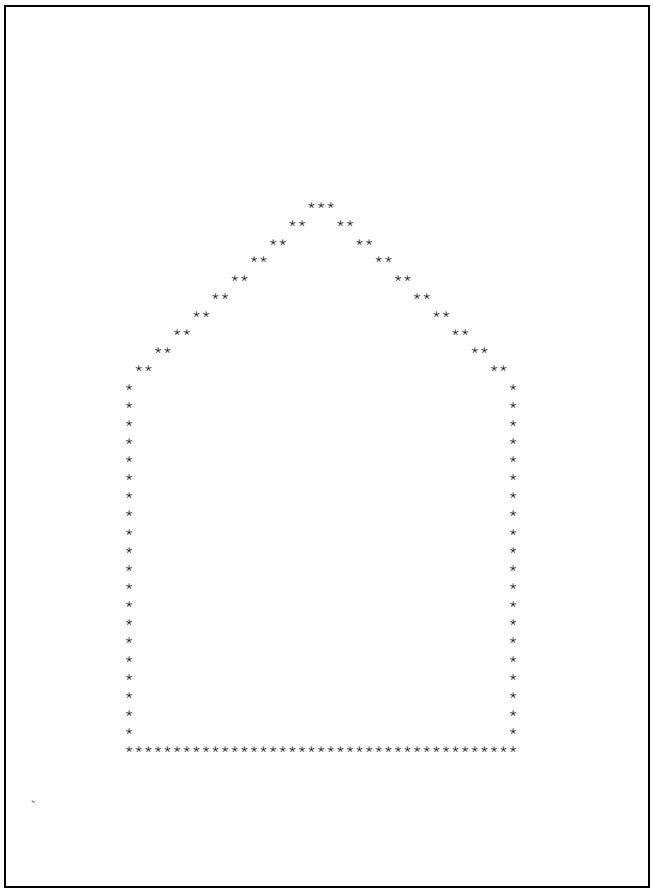

Figure 2: Rendering of sample image file.