EECS 293
Software Craftsmanship
2016 Fall Semester

Programming Assignment 10

Due at the beginning of your discussion session on

November 7-11, 2016

Reading

- Chapter 8 in Code Complete
- Singleton pattern (page 151 in Section 6.3, Code Complete),
- Try-finally (page 404 in Section 17.3, Code Complete)
- Items 57, 58, 61, 62, and 63 in Effective Java
- Section 19.6 in Code Complete.

Grading Guidelines

An automatic C (or less) is triggered by an incomplete error-handling document.

Points will be deducted if code and branch coverage is incomplete. An automatic C (or less) is triggered by:

- Any routine with complexity greater than 4,
- Any substantially repeated piece of code, or by
- Improperly named routines.

Programming

Consider the following five pictures represented as 9×8 arrays:

```
. . . . . . . .
E E E E E E . .
E . . . . E . .
E . . . . E . .
E . . . . E . .
E . . . . E . .
E . . . . E . .
E . . . . E . .
E E E E E E . .

. . . . . . . .
. . . . . . . .
D D D D D D . .
D . . . . D . .
D . . . . D . .
D . . . . D . .
D D D D D D . .
. . . . . . . .
. . . . . . . .

. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . A A A A
. . . . A . . A
. . . . A . . A
. . . . A A A A
. . . . . . . .

. . . . . . . .
. . B B B B . .
. . B . . B . .
. . B . . B . .
. . B . . B . .
. . B B B B . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
. C C C . . . .
. C . C . . . .
. C . C . . . .
. C C C . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
```

If the pictures are placed on top of each other with E at the bottom, followed by D, A, B, and C at the top, the result is:

```
. C C C . . . .
E C B C B B . .
D C B C D B . .
D C C C . B . .
D . B . A B A A
D . B B B B . A
D D D D A D . A
E . . . A A A A
E E E E E E . .
```

If any part of a picture covers another, it hides that part of the pictures underneath.

Your code should have a main method. It will take as input a description of the underlying pictures and the final overlapped picture and determine the order in which the pictures were stacked.

The following rules apply to the input and output of your program. You should adhere to these rules strictly or you program can be at risk of failing automated test cases. The standard input should contain:

- Lines 1 and 2: picture height (rows) and width (columns) respectively.
- Subsequent lines: the pictures, represented with capital letters and dots separated by a single space. An empty line separates one picture from the next.

No two different pictures will be assigned the same capital letter. Except for the fact that each picture corresponds to a capital letter, there is no intrinsic limit to the number of pictures. There is no restriction on the shape of each picture. The final picture should show at least one letter from each of the source pictures. A correct input should allow for one and only one way to stack the pictures.

The standard output should contain the letters of the pictures in the order in which they were stacked. In case of error, it should contain only contain the string 'error'. An example of standard input and standard output is posted on blackboard.

Design Document

Submit a separate text file to document the error handling architecture that you will follow in your implementation. The architecture document should describe your error handling choices such as a strategy for handling erroneous user input, decisions on local or global error handling, error propagation through the code, presence and location of a barricade, and the other factors in the defensive programming checklist at the end of chapter 8. You will be asked to demonstrate that your code follows the error handling architecture.

Run

Create a new task called run (invoked with 'ant run' or 'make run') that executes your code taking the input from standard input and putting its output on standard output.

General Considerations

Your implementation may contain as many auxiliary private methods as you see fit, and additional helper classes may be defined. Your code should have an exhaustive unit test suite. Your code should have a reasonable number of comments, but documentation is going to be the topic of a future assignment. As a general guideline at this stage of the course, comments should be similar to those accepted in EECS 132.

Discussion Guidelines

The project discussion will focus on defensive programming. In particular, you will be asked to demonstrate that your code follows the error handling architecture.

Submission

Create a repository called pictures.git where you will post your submission. Make small regular commits and push your revised code and test cases on the git repository. Submit a separate text file to document the error handling architecture.