Software Engineering Lab

Started on	Wednesday, 27 January 2021, 4:55 PM
State	Finished
Completed on	Wednesday, 27 January 2021, 4:57 PM
Time taken	2 mins 11 secs
Grade	Not yet graded

Question 1

Complete

Marked out of 50.00

Flag question

Write an object-oriented java program which composes two 2D asterix patterns and prints a composite 2D asterix pattern. Here the composite pattern is obtained by replacing each asterix in the first pattern with the second pattern. For example, let 5-rightarrow (for total height of 5) pattern be:

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

And 2-box be:

```
**
**
```

Then composition pattern 5-rightarrow-2-box is:

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

While the composition pattern 2-box-5-rightarrow is:

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

You should structure your code such that:

1. Each pattern is a type so that you can instantiate objects of that pattern.
2. The composed patterns should also be of the same type so that they can be composed with other patterns.
3. The code for composition should be generic and not duplicated in the different types.

Design ideas: Implement the functions “generatePattern()”, which generates the pattern in a two-dimensional character array, and “compose()” which takes the base type, and an object of another type to generate the composed character array. The compose() function should be implemented as a part of a super-class (say “Pattern”) which should be inherited by other pattern classes. Each pattern subclass should inherit the pattern of a superclass and take an object of another pattern subclass as argument to construct a composite pattern subclass and implement the “generatePattern()” by using the generatePattern() functions of superclass and argument.

Output: Write a program which generates a n-box-k-rightarrow pattern, where n and k are user input. You should print your input patterns and composed pattern. Note that your program should be extendable to other types of patterns, e.g. 5-diamond pattern:

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

```
import java.util.Scanner;

public class Solution {
public static void main(String[] args) {
int n,k;
        Scanner input = new Scanner(System.in);
        System.out.print("Enter n: ");
        n=input.nextInt();
        System.out.print("Enter k: ");
        k=input.nextInt();
        Box a = new Box(n);
        Arrow b = new Arrow(k);
        System.out.println("Box:\n");
        a.print();
        System.out.println();
        System.out.println("Arrow:\n");
        b.print();
        System.out.println();
        System.out.println("Composition:\n");
        a.compose(b);
    }
}

class Pattern{
int cols, rows;
    char [][] mat;
    public Pattern(){}
void fillSpaces(){
for(int i=0; i<rows; i++){
for(int j=0; j<cols; j++){
mat[i][j]=' ';
            }
        }
    }
void generatePattern(){}
void print(){
for(int i=0; i<rows; i++){
for(int j=0; j<cols; j++){
                System.out.print(mat[i][j]);
            }
            System.out.println();
        }
    }

void compose(Pattern other){
for(int i=0; i<rows; i++){
int star_cnt=0;
            for(int j=0; j<cols; j++){
if(mat[i][j]=='*')
                    star_cnt++;
            }
for(int x=0; x<other.rows; x++){
for(int rep=0; rep<star_cnt; rep++){
```

```
for(int y=0; y< other.cols; y++){
                        System.out.print(other.mat[x][y]);
                    }
                }
                System.out.println();
            }
        }
    }
}

class Box extends Pattern{
public Box(int n){
cols=n;
        rows=n;
        mat=new char[rows][cols];
        fillSpaces();
        generatePattern();
    }

void generatePattern() {
for(int i=0; i<rows; i++){
for(int j=0; j<cols; j++){
mat[i][j]='*';
            }
        }
    }
}

class Arrow extends Pattern{
public Arrow(int n){
cols=(n/2)+1;
        rows=n;
        mat=new char[rows][cols];
        fillSpaces();
        generatePattern();
    }

void generatePattern() {
int center=rows/2;
        for(int i=0; i<rows; i++){
int rep;
            if(i<=center)
                rep=i+1;
            else
                rep=rows-i;
            for(int j=0; j<rep; j++){
mat[i][j]='*';
            }
        }
    }
}
```

Finish review

QUIZ NAVIGATION

1

Finish review