# Hands-on Experiment # 12: Worksheet

Section 2 Date 21 November 2019

No more than 3 students per one submission of this worksheet.

Student ID 6238063321 Name Nattawat Pornthisan

Student ID 6238138821 Name Pattanun Sangvijitr

## Part A: Getting Familiar with Problem (Do not code here)

In this lab, we aim to write a program to draw many geometric shapes (Square, Triangle) using standards keyboard characters. In order to draw a figure, there are 3 input parameters: character, the number of rows (size), and indent. Assume character is “#”, the number of rows is 5 and the indent is 2 (the spaces is represented by “.”).

|  |  |  |  |
| --- | --- | --- | --- |
| fill() | ..#####  ..#####  ..#####  ..#####  ..##### | ..######  ..#######  ..######## ­..#########  ..########## | ..#  ..##  ..###  ..####  ..##### |
| draw() | ..#####  ..# #  ..# #  ..# #  ..##### | ..######  ..#######  ..#### ### ­..### # ###  ..########## | ..#  ..##  ..# #  ..# #  ..##### |
|  | Square | Triangle | RightTriangle |

Assume the size is 6 rows using a character ‘+’, **draw and fill** the following shapes and **compute** their perimeters and areas.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Square | Triangle | RightTriangle |
| Draw | ++++++  + +  + +  + +  + +  ++++++ | +  + +  + +  + +  + +  + + + + + + | +  ++  +++  ++++  +++++  ++++++ |
| Fill | ++++++  ++++++  ++++++  ++++++  ++++++  ++++++ | +  + +  + + +  + + + +  + + + + +  + + + + + + | +  ++  + +  + +  + +  ++++++ |
| VerticalFlipped | ++++++  ++++++  ++++++  ++++++  ++++++  ++++++  ++++++  + +  + +  + +  + +  ++++++ | + + + + + +  + + + + +  + + + +  + + +  + +  +  + + + + + +  + +  + +  + +  + +  + | ++++++  +++++  ++++  +++  ++  +  ++++++  + +  + +  + +  ++  + |
| HorizontalFlipped | ++++++  ++++++  ++++++  ++++++  ++++++  ++++++  ++++++  + +  + +  + +  + +  ++++++ | +  + +  + + +  + + + +  + + + + +  + + + + + +  +  + +  + +  + +  + +  + + + + + + | +  ++  +++  ++++  +++++  ++++++  +  ++  + +  + +  + +  ++++++ |
| Area | 36 | 18 | 18 |
| Perimeter | 24 | 19.42 | 20.49 |

## Part B: Design Your Class (Do not code here)

The below figure shows a part of the program: Shape and Square. Shape is a superclass of any shapes and there are 3 *protected* variables (row, character, and indent) – represented by the “#” symbol.

A screenshot of a cell phone

Description automatically generated

Class “Shape”

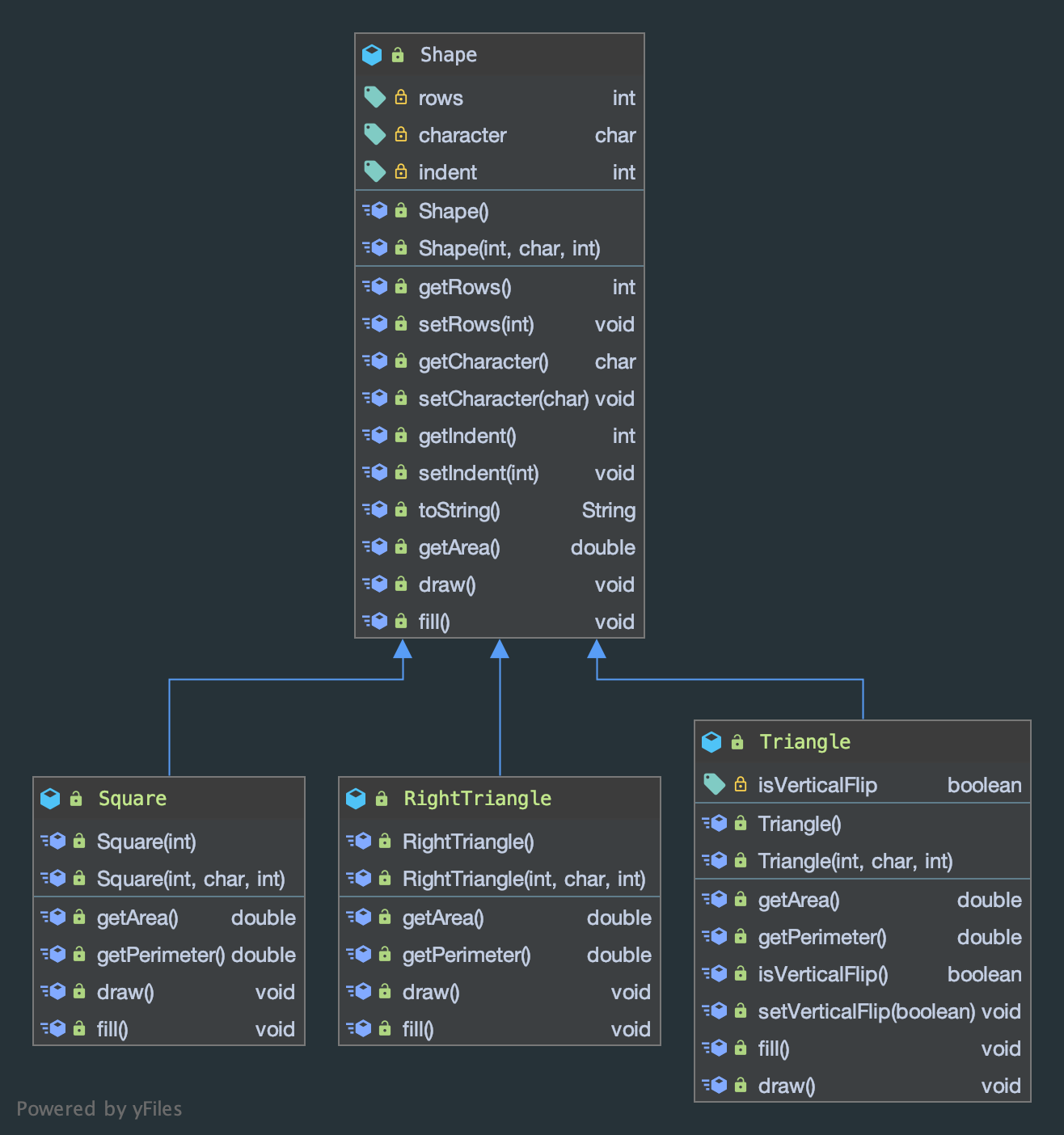
* There are three properties (variables): *row, character, indent*
* There are 2 constructors.
* There are getter & setter methods for all properties (variables).
* toString() shows all variables’ value; e.g., “row=5 and character=#”.
* **The draw and fill method shows a message “Do not know what shape to draw/fill yet!”. The getArea() method always returns 1.0 as a default value.**

Class “Square”

* There are 2 constructors.
* draw(): to draw outline of a square
* fill(): to fill a square
* getArea() and getPerimeter(): to compute area and perimeter of the object.

Write UML diagram of all shapes including: Shape, Square, Triangle, and RightTriangle

* In Triangle, there is a variable called “isVerticalFlip”. If it is true, the figure is flipped by horizontal axis.
  + In order to get and set this variable, there are 2 extra methods: boolean isVerticalFlip() and void setVerticalFlip(boolean isVerticalFlip).



What is the parent class of Shape?

The parent class of Shape is **Shape**, according to the UML diagram above.

## Part C: Implement Your Classes

Implement all classes based on your design in Part B and copy your implemented classes to the provided space below. Note that TestDraw1.java, TestDraw2.java, and TestDraw3.java are given.

Source code for class Shape, Square, Triangle, and RightTriangle

Shape

*public class* Shape {  
  
 *protected int* rows;  
 *protected char* character;  
 *protected int* indent;  
  
 *public* Shape() {  
 *this*(0, ' ', 0);  
 }  
  
 *public* Shape(*int* x, *char* c, *int* i) {  
 setRows(x);  
 setCharacter(c);  
 setIndent(i);  
 }  
  
 *public int* getRows() { *return* rows; }  
 *public void* setRows(*int* x) { rows = x; }  
  
 *public char* getCharacter() { *return* character; }  
 *public void* setCharacter(*char* c) { character = c; }  
  
 *public int* getIndent() { *return* indent; }  
 *public void* setIndent(*int* i) { indent = i;}  
  
 *public* String toString() {  
 *return* "(rows: " + getRows()  
 + ", characters: " + getCharacter()  
 + ", indent: " + getIndent()  
 + ")";  
 }  
  
 *public double* getArea() {  
 *return* 1.0;  
 }  
  
 *public void* draw() {  
 System.*out*.println("Do not know what shape to draw/fill yet!");  
 }  
  
 *public void* fill() {  
 System.*out*.println("Do not know what shape to draw/fill yet!");  
 }  
  
}

Square

*public class* Square *extends* Shape {  
  
 *public* Square(*int* x) {  
 *super*();  
 setRows(x);  
 }  
  
 *public* Square(*int* x, *char* c, *int* i) {  
 *super*(x, c, i);  
 }  
  
 *public double* getArea() { *return* rows \* rows; }  
 *public double* getPerimeter() { *return* 4 \* rows; }  
  
 *public void* draw() {  
 String spaces = "";  
 *int* a = 0;  
 *while* (a++ < indent) spaces = " ".concat(spaces);  
 *for* (*int* x = 0; x < rows; x++) {  
 System.*out*.print(spaces);  
 *if* (x > 0 && x < rows - 1) System.*out*.printf("%c%s%c%n", character, " ".repeat(rows - 2), character);  
 *else* System.*out*.printf("%s%n", (character + "").repeat(rows));  
 }  
 }  
  
 *public void* fill() {  
 String spaces = "";  
 String body = (character + "").repeat(rows);  
 *int* a = 0;  
 *int* b = 0;  
 *while* (a++ < indent) spaces = " ".concat(spaces);  
 *while* (b++ < rows) System.*out*.printf("%s%s%n", spaces, body);  
 }  
  
}

Triangle

*public class* Triangle *extends* Shape {  
  
 *protected boolean* isVerticalFlip;  
  
 *public* Triangle() {  
 *super*();  
 isVerticalFlip = *false*;  
 }  
  
 *public* Triangle(*int* x, *char* c, *int* i) {  
 *super*(x, c, i);  
 isVerticalFlip = *false*;  
 }  
  
 *public double* getArea() { *return* rows \* rows / 2.0; }  
 *public double* getPerimeter() { *return* 2 \* Math.*sqrt*(rows \* rows + (rows / 2.0) \* (rows / 2.0)) + rows; }  
  
 *public boolean* isVerticalFlip() { *return* isVerticalFlip; }  
 *public void* setVerticalFlip(*boolean* b) { isVerticalFlip = b;}  
  
 *public void* fill() {  
 String body = "";  
 *if* (!isVerticalFlip) {  
 *for* (*int* x = 0; x < rows; x++) {  
 *if* (indent > 0) System.*out*.printf("%s", " ".repeat(indent));  
 body = (character + " ").repeat(x + 1).trim();  
 *if* (x == rows - 1) System.*out*.printf("%s%n", (character + " ").repeat(x + 1).trim());  
 *else* System.*out*.printf("%s%s%s%n", " ".repeat(rows - x - 1), body, " ".repeat(rows - x - 1));  
 }  
 *return*;  
 }  
 *for* (*int* x = rows - 1; x > -1; x--) {  
 *if* (indent > 0) System.*out*.printf("%s", " ".repeat(indent));  
 body = (character + " ").repeat(x + 1).trim();  
 *if* (x == rows - 1) System.*out*.printf("%s%n", (character + " ").repeat(x + 1).trim());  
 *else* System.*out*.printf("%s%s%s%n", " ".repeat(rows - x - 1), body, " ".repeat(rows - x - 1));  
 };  
 }  
  
 *public void* draw() {  
 String body = "";  
 *if* (!isVerticalFlip) {  
 *for* (*int* x = 0; x < rows; x++) {  
 *if* (indent > 0) System.*out*.printf("%s", " ".repeat(indent));  
 *if* (x > 1 && x < rows - 1) body = (character + " ") + " ".repeat(x - 1) + character;  
 *else* body = (character + " ").repeat(x + 1).trim();  
 *if* (x == rows - 1) System.*out*.printf("%s%n", body);  
 *else* System.*out*.printf("%s%s%s%n", " ".repeat(rows - x - 1), body, " ".repeat(rows - x - 1));  
 }  
 *return*;  
 }  
 *for* (*int* x = rows - 1; x > -1; x--) {  
 *if* (indent > 0) System.*out*.printf("%s", " ".repeat(indent));  
 *if* (x > 1 && x < rows - 1) body = (character + " ") + " ".repeat(x - 1) + character;  
 *else* body = (character + " ").repeat(x + 1).trim();  
 *if* (x == rows - 1) System.*out*.printf("%s%n", body);  
 *else* System.*out*.printf("%s%s%s%n", " ".repeat(rows - x - 1), body, " ".repeat(rows - x - 1));  
 }  
 }  
}

Right Triangle

*public class* RightTriangle *extends* Shape {  
  
 *public* RightTriangle() {  
 *super*();  
 }  
  
 *public* RightTriangle(*int* x, *char* c, *int* i) {  
 *super*(x, c, i);  
 }  
  
 *public double* getArea() { *return* rows \* rows / 2.0; }  
 *public double* getPerimeter() { *return* 2 \* Math.*sqrt*(rows \* rows + (rows / 2.0) \* (rows / 2.0)) + rows; }  
  
 *public void* draw() {  
 String c = character + "";  
 *for* (*int* p = 0; p < rows; p++) {  
 *if* (indent > 0) System.*out*.printf("%s", " ".repeat(indent));  
 *if* (p == 0 || p == 1 || p == rows - 1) System.*out*.printf("%s%n", c.repeat(p + 1));  
 *else* System.*out*.printf("%c%s%c%n", character, " ".repeat(p - 1), character);  
 }  
 }  
  
 *public void* fill() {  
 String c = character + "";  
 *for* (*int* p = 0; p < rows; p++) {  
 *if* (indent > 0) System.*out*.printf("%s", " ".repeat(indent));  
 System.*out*.printf("%s%n", c.repeat(p + 1));  
 }  
 }  
  
}

What statement the you have to put in the constructor of Square and Triangle in order to make the class compiled without error? Why those statements are required?

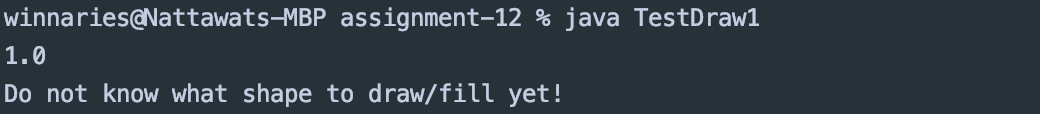
We use the ***super*** statement to call the constructor function of its parent class, which is class *Shape*. If done properly, it will enable the children class to use the inherited variable from those of parent class.

## Part D: Test Your Classes

1) What is the result of TestDraw1.java (code below)?

|  |
| --- |
| **public** **class** TestDraw1 {  **public** **static** **void** main(String[] args) {  Shape shape = **new** Shape(7,'\*', 2);  System.***out***.println(shape.getArea());  shape.draw();  }  } |

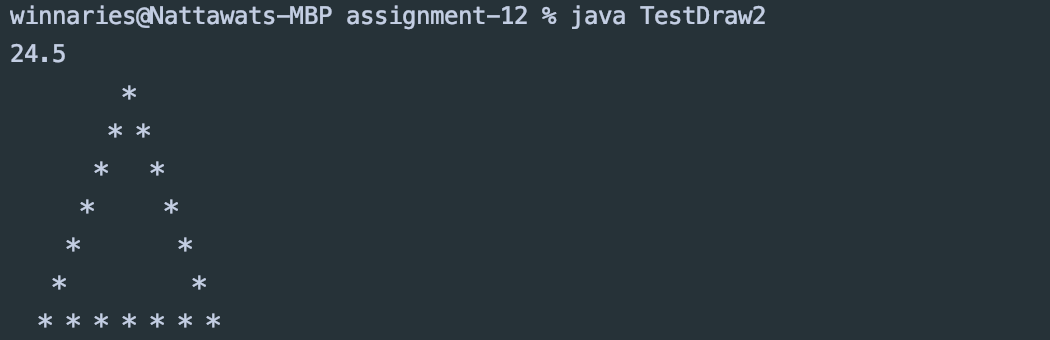
Include the screenshots below.



2) What is the result of TestDraw2.java (code below)?

|  |
| --- |
| **public** **class** TestDraw2 {  **public** **static** **void** main(String[] args) {  Triangle triangle1 = **new** Triangle(7,'\*', 2);  System.***out***.println(triangle1.getArea());  triangle1.draw();  }  } |

Include the screenshots below.



3) What is the result of TestDraw3.java (code below)?

|  |
| --- |
| **public** **class** TestDraw3 {  **public** **static** **void** main(String[] args) {  Shape triangle1 = **new** Triangle(7, '\*', 2);  System.***out***.println(triangle1.getArea());  triangle1.fill();  }  } |

Include the screenshots below.



What is the property name that allows you to assign subclass to superclass (on the highlight line in the code above)?

***Runtime polymorphism*** is the property that allows the program to assign *subclass* into its *superclass*.

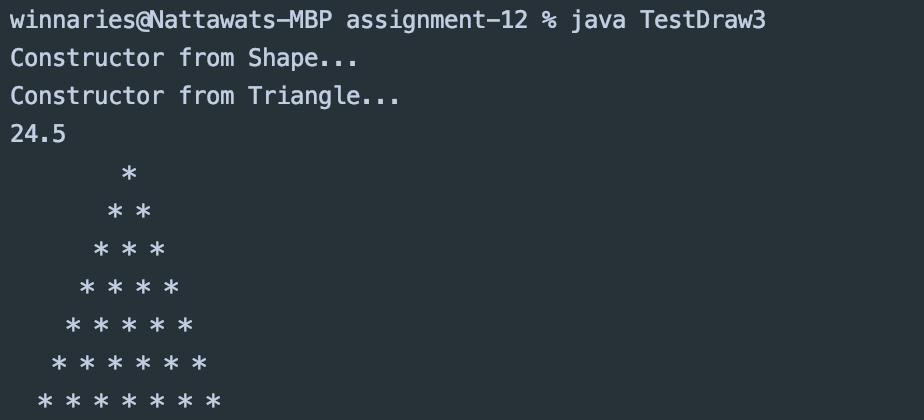
When statement

Shape triangle1 = new Triangle(7, ‘\*’, 2);

Is executed, what constructors need to be called and in what order? (Try to figure out without using System.out.println() in the constructor.)

The constructor from the its parent class should only be called first, via a *super* statement, before executing the constructor from the subclass.

Run the statement in the previous question with System.out.println() in the constructor, or use “Java Visualizer” to see the execution order. Capture the screen the show the result of execution order. Why the execution is in that order?



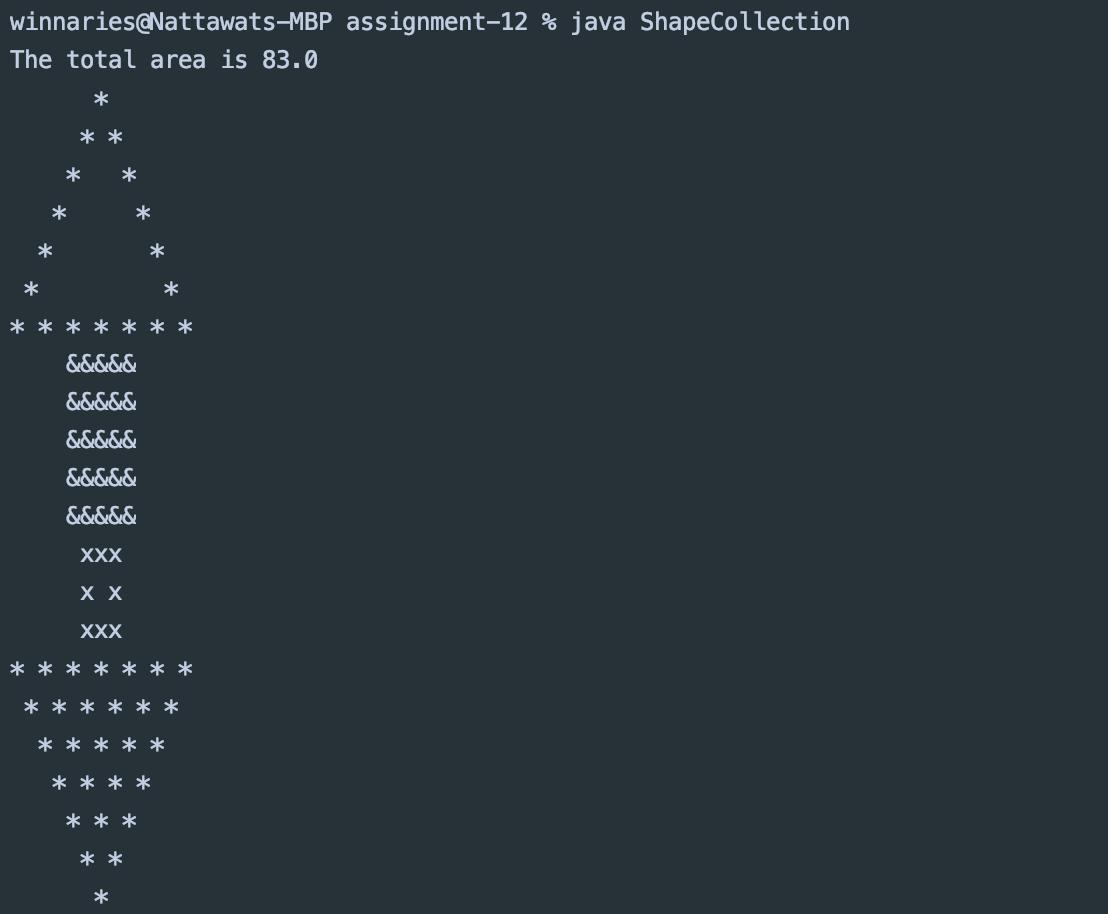
This image proves that the program will firstly run the constructor from the superclass, in this case *Shape*, before the subclass’ constructor.

## Part E: Apply Implemented Classes

Implement ShapeCollection.java (an initial code is given) to draw “an up and down arrow” along with the total area as shown in the figure below.

|  |  |
| --- | --- |
| The total area is 83.0  \*  \* \*  \* \*  \* \*  \* \*  \* \*  \* \* \* \* \* \* \*  &&&&&  &&&&&  &&&&&  &&&&&  &&&&&  xxx  x x  xxx  \* \* \* \* \* \* \*  \* \* \* \* \* \*  \* \* \* \* \*  \* \* \* \*  \* \* \*  \* \*  \* | To draw this picture, they are comprised of 4 shapes:   1. Triangle with 7 rows and no indent 2. Square with 5 rows and 4 spaces of indent 3. Square with 3 rows and 5 spaces of indent 4. Vertically Flipped Triangle of the first shape   In the class, you must create ArrayList to collect all the shapes. Also, there are 2 static methods including   * public static double computeArea(ArrayList<Shape> shapeList) * public static void draw(ArrayList<Shape> shapeList) |

Include the screenshots below.



List all your source code here.

*import* java.util.ArrayList;  
*import* java.util.*Iterator*;  
  
*public class* ShapeCollection {  
 *public static void* main(String[] args) {  
 ArrayList<Shape> upDownArrow = *new* ArrayList<Shape>();  
 upDownArrow.add(*new* Triangle(7, '\*', 0));  
 upDownArrow.add(*new* Square(5, '&', 4));  
 upDownArrow.add(*new* Square(3, 'x', 5));  
 Triangle triangle = *new* Triangle(7, '\*', 0);  
 triangle.setVerticalFlip(*true*);  
 upDownArrow.add(triangle);  
 System.*out*.printf("The total area is %.1f %n", *computeArea*(upDownArrow));  
 *draw*(upDownArrow);  
 }  
  
 *public static double* computeArea(ArrayList<Shape> shapeList) {  
 *double* totalArea = 0;  
 *for* (Shape shape : shapeList) {  
 totalArea += shape.getArea();  
 }  
 *return* totalArea;  
 }  
  
 *public static void* draw(ArrayList<Shape> shapeList) {  
 shapeList.get(0).draw();  
 shapeList.get(1).fill();  
 shapeList.get(2).draw();  
 shapeList.get(3).fill();  
 }  
}

Submit this worksheet (by only one member of the group) via <http://www.myCourseVille.com> (Assignments > Hands-on Experiment # 12) before noon of the day after your lecture.