CS: 113 Introduction to Object Oriented Programming (IOOP) Fall, 2022

Project 5 (200 points) Due 12/14/2022

Submit your solutions to canvas. For programming assignments do not send the entire project. All I want are the files ending in .java. **Each class will have to be defined in its own separate .java file.** All driver code should be put in one .java file, with each problem driver named accordingly as: `P1(), P2(), … Pn()..` Please make sure your name is included at the top of each .java file. **There will be no re-submission of work. There will be no exceptions. So, before you submit your work please ensure you have everything the way you want it.**

The grading breakdown for this project is as follows:

10% Readability – Is the program easy to read and understand (indentation, documentation, good use of white space, good output format, user prompts)

10% C++ – Does the program make good use of the C++ constructs (functions, control flow, etc.)

20% Robustness - Does the program compile and run, and not crash or throw exceptions

50% Correctness – Does the program solve the intended problem and work on a variety of reasonable inputs

10% Creativity Effort – How much effort was given to making a compelling viewing/interactive experience.

Problem 1

For this project you are to progressively build a 2D graphics engine usually the OOP techniques we have learned in this class. You should have an abstract class called `Shape2Di` with a polymorphic method:

```
public abstract void Draw(Graphics g);
```

There are a few main requirements for this project:

1. Extend an implement `Shape2Di` with at least 4 subclasses. They are:
    a. `Circle2Di`
    b. `Rectangle2Di`
    c. `Polygon2Di`
    d. `Line2Di`
2. You can inherit from the Polygon2Di class and make additional Shapes
3. You can use the link below to add other graphical shapes
    a. https://docs.oracle.com/javase/7/docs/api/java/awt/Graphics.html

4. Once you complete item 1 above you are to create to driver methods `P1()` and P2() where:
    a. P1() Creates a nice compelling static image. Use color and shapes and your imagination
    b. `P2()` Add the dimension of time and creates some animation or game.

Your first milestone is to use the code that was posted on canvas to do the following:

Implement an abstract class called `Shape2Di`. The attributes (data members) and method are listed below:

Fields

Modifier and Type	Field
`static java.awt.Color[]`	`COLORS`
`boolean`	`fill`
`java.awt.Color`	`fillColor`
`int`	`fillColorIndex`
`boolean`	`outline`
`java.awt.Color`	`outlineColor`
`int`	`outlineColorIndex`
`int`	`xPos`
`int`	`xVel`
`int`	`yPos`
`int`	`yVel`

Constructor Summary

Constructors

Constructor	Description
`Shape2Di()`	Default Constructor
`Shape2Di(int fillColorIndex, int xPosition, int yPosition)`	Parametric Constructor

Method Summary

All Methods | Instance Methods | Abstract Methods | Concrete Methods

Modifier and Type	Method	Description
void	Animate()	Move the shape along its velocity vector's trajectory
abstract void	Draw(java.awt.Graphics g)	Draw the shape to the screen at its current position
boolean	GetFill()	Returns the render fill state of true or false
java.awt.Color	GetFillColor()	Gets the current fill color of the shape
int	GetFillColorIndex()	Gets the current fill color index of the shape
boolean	GetOutline()	Returns the render outline state of true or false
java.awt.Color	GetOutlineColor()	Gets the current outline color of the shape
int	GetOutlineColorIndex()	Gets the current outline color index of the shape
int	GetX()	Gets the current x position of the shape
int	GetXSpeed()	Gets the speed of the shape for xVel
int	GetY()	Gets the current y position of the shape
int	GetYSpeed()	Gets the speed of the shape for xVel
void	Move(double xDelta, double yDelta)	Moves the shape by an amount (xDelta, yDelta)
void	SetFill(boolean setting)	Set the shape state such that the shape is rendered filled
void	SetFillColor(int fillColorIndex)	Sets the current fill color of the shape
void	SetOutline(boolean setting)	Set the shape state such that a outline is drawn
void	SetOutlineColor(int outlineColorIndex)	Sets the current outline color of the shape
void	SetPos(int x, int y)	Moves the shape to the location (x, y)
void	SetSpeed(int xV, int yV)	Sets the speed of the shape for both the xV, and yV

The `Circle2Di` data members are:

Field Summary

Fields

Modifier and Type	Field
int	diameter

Fields inherited from class Shape2Di

COLORS, fill, fillColor, fillColorIndex, outline, outlineColor, outlineColorIndex, xPos, xVel, yPos, yVel

The `Circle2Di` methods are:

Constructor Summary

Constructors

Constructor
Circle2Di()
Circle2Di(int colorIndex, int xPosition, int yPosition, int diameter)

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method	Description
void	Animate()	Animate moves the cricle a step along its velocity vector
void	Draw(java.awt.Graphics g)	public void Draw(Graphics g) Render the circle for both filled and outlined according to the states
int	GetDiameter()	Gets the diamater of the circle

The `Rectangle2Di` data members are:

java.lang.Object
    Shape2Di
        Rectangle2Di

```
public class Rectangle2Di
extends Shape2Di
```

Field Summary

Fields

Modifier and Type	Field
int	height
int	width

Fields inherited from class Shape2Di

COLORS, fill, fillColor, fillColorIndex, outline, outlineColor, outlineColorIndex, xPos, xVel, yPos, yVel

The `Rectangle2Di` methods are:

Constructors

Constructor	Description
Rectangle2Di()	Constructor for shapes
Rectangle2Di(int colorIndex, int xPosition, int yPosition, int width, int height)	

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method	Description
void	Animate()	Animate moves the polygon a step along its velocity vector
void	Draw(java.awt.Graphics g)	Render the polygon at it position as filled and outlined according to the object state
int	GetHeight()	Gets the height of the rectangle
int	GetWidth()	Gets the width of the rectangle

Methods inherited from class Shape2Di

GetFill, GetFillColor, GetFillColorIndex, GetOutline, GetOutlineColor, GetOutlineColorIndex, GetX, GetXSpeed, GetY, GetYSpeed, Move, SetFill, SetFillColor, SetOutline, SetOutlineColor, SetPos, SetSpeed

The `Polygon2Di` data members are:

Class Polygon2Di

java.lang.Object
    Shape2Di
        Polygon2Di

```
public class Polygon2Di
extends Shape2Di
```

Field Summary

Fields

Modifier and Type	Field
int[]	txCoords
int[]	tyCoords
int[]	xCoords
int[]	yCoords

Fields inherited from class Shape2Di

COLORS, fill, fillColor, fillColorIndex, outline, outlineColor, outlineColorIndex, xPos, xVel, yPos, yVel

The `Polygon2Di` methods are are:

Constructors

Constructor	Description
Polygon2Di(int colorIndex, int xPosition, int yPosition, int[] xCoords, int[] yCoords)	

Method Summary

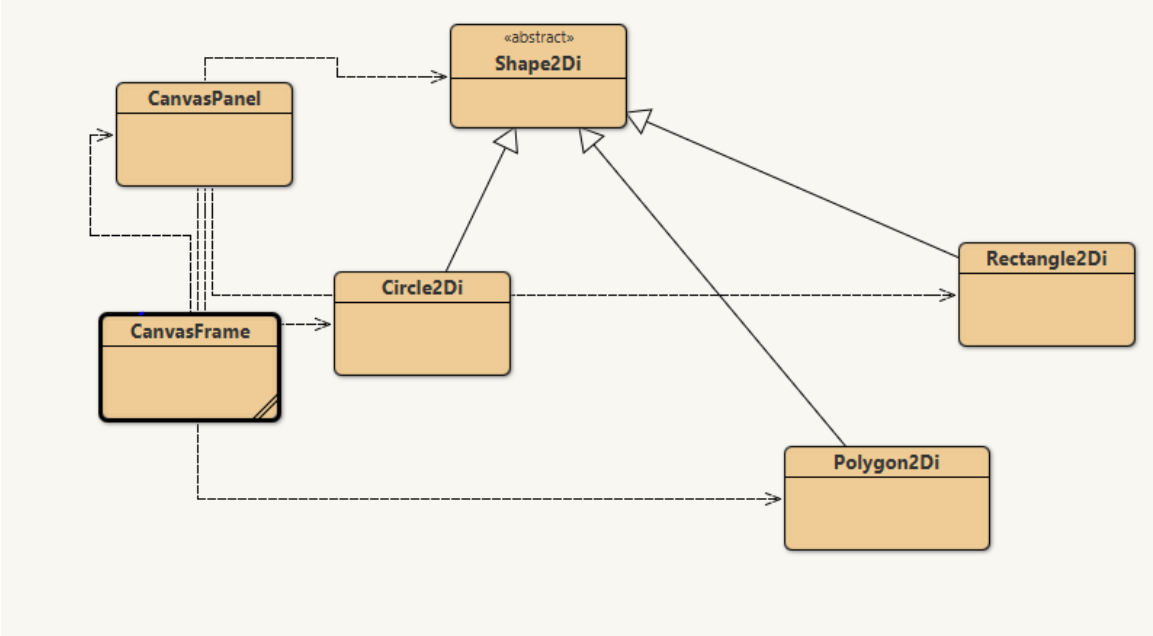
All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method	Description
void	Animate()	Animate moves the polygon a step along its velocity vector
void	Draw(java.awt.Graphics g)	Draw the shape to the screen at its current position

Methods inherited from class Shape2Di

GetFill, GetFillColor, GetFillColorIndex, GetOutline, GetOutlineColor, GetOutlineColorIndex, GetX, GetXSpeed, GetY, GetYSpeed, Move, SetFill, SetFillColor, SetOutline, SetOutlineColor, SetPos, SetSpeed

The class hierarchy looks like this:



An image looks like this: