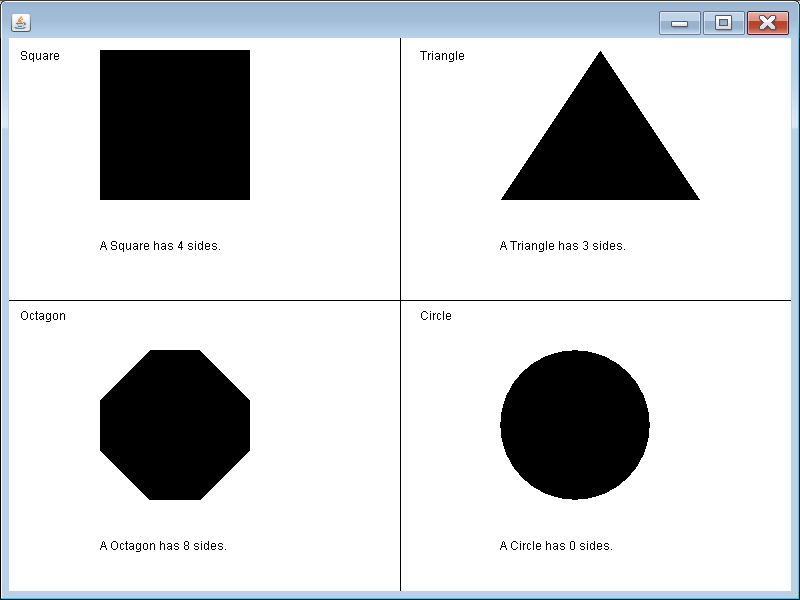
|  |  |
| --- | --- |
| **AP Computer Science** | **GraphicsLab06 Java Assignment** |
| **The Polymorphic Shapes**  **Graphics Program** | **100 Point Version Only** |
| **Assignment Purpose:**  The purpose of this program is to become comfortable with interfaces, abstract classes and polymorphism. | |

Write a program which uses an **interface**, an **abstract class** and 4 *concrete* classes to generate the output below:

**100 Point Version Output**



Please note, the purpose of this lab is NOT to merely be a graphics assignment. To get credit students will need to show their code to confirm they are using an **interface**, an **abstract class**, and *polymorphism* properly. After that, they will need to show the code being compiled and executed. This program is done as a graphics application without the use of an applet and a webpage file.

Do not alter the provided **main** or **paint** methods.

|  |
| --- |
| **GraphicsLab06st.java Provided Student File** |
| // GraphicsLab06st.java  // The Polymorphic Graphics Shapes Program  // Student, starting Version  import java.util.ArrayList;  import java.awt.\*;  import java.awt.event.\*;  public class GraphicsLab06st  {  public static void main(String args[])  {  Windows win = new Windows();  win.setSize(800,600);  win.addWindowListener(new WindowAdapter() {public void  windowClosing(WindowEvent e) {System.exit(0);}});  win.show();  }  }  class Windows extends Frame  {  public void paint(Graphics g)  {  ArrayList<Shape> shapes = new ArrayList<Shape>();  shapes.add(new Square());  shapes.add(new Triangle());  shapes.add(new Octagon());  shapes.add(new Circle());  drawGrid(g);  for(Shape shape: shapes)  {  shape.drawShape(g);  shape.displayName(g);  shape.displayNumSides(g);  }  }  public void drawGrid(Graphics g)  {  g.drawLine(0,300,800,300);  g.drawLine(400,0,400,600);  }  } |

**100 Point Version Specifics**

For this assignment you need to create an **interface**, an **abstract class**, and 4 *concrete* classes.

Details for each are below:

The **Shape** interface needs to contain these 3 **abstract** methods:

|  |  |
| --- | --- |
| **drawShape** | Draws the square, triangle, octagon or circle on the screen |
| **displayName** | Displays the title at the top of the cell |
| **displayNumSides** | Displays the message at the bottom of the cell which indicate how many sides the shape has. |

The **AbstractShape** abstract class needs to contain these 6 **attributes** & these 2 *concrete* methods:

|  |  |
| --- | --- |
| **numSides** | The number of sides a particular shape has |
| **shapeName** | The name of the shape:  Examples: “Square”, “Triangle”, “Octagon” or “Circle” |
| **titleX**  **titleY** | The x and y value for the title displayed at the top of the cell. |
| **messageX**  **messageY** | The x and y value for the message displayed at the bottom of the cell. |
| **displayName** | Displays the title at the top of the screen. |
| **displayNumSides** | Displays the message at the bottom of the screen. |

These 4 *concrete* classes will each extend the **AbstractShape** class:

* **Square**
* **Triangle**
* **Octagon**
* **Circle**