|  |  |
| --- | --- |
| **Pre-AP® Computer Science** | **Lab 04B**  **Practice/Perform Major Java Assignment** |
| **The Expo Graphics Program** | **60, 70, 80, 90, 100 and 110 Point Versions** |
| **Assignment Purpose:**  The purpose of this program is to demonstrate knowledge of calling methods,  and use correct parameter passing with some of the graphics methods found  in the **Expo** class. | |

Write a program, which displays several geometric designs using the provided **Expo** class. This is your first Practice/Perform lab. You will have 1 day to *practice* this assignment. On the practice day you can ask questions and get help. Then you need to *perform*. On this day you need to do the lab, from scratch, for a grade – with no help. Some teachers call this *The Day of Reckoning*.

Whether practicing or performing, you will be provided with a skeleton program. Your job is to use the proper methods along with the correct parameter values to match the output shown on this assignment. The syntax of some **Expo** methods is shown below. Some of these methods will be necessary for this lab, but not all of them.

**Selected Methods from the Expo class used for Lab 04B**

**Expo.setColor(g,Expo.red);**

Changes the graphics color to **red**. Many other colors are available including **blue**, **yellow**, **black** and **white**.

**Expo.drawPixel(g,100,200);**

Draws a very small single dot (pixel) on the computer screen 100 pixels over and 200 pixels down.

**Expo.drawPoint(g,100,200);**

Draws a larger, more visible dot on the computer screen 100 pixels over and 200 pixels down.

**Expo.drawLine(g,100,200,300,400);**

Draws a line segment connecting the starting coordinate point (100,200) with the ending point (300,400).

**Expo.drawCircle(g,300,200,100);**

Draws an open circle with a radius of 100 pixels whose center is located at the coordinate (300,200).

**Expo.fillCircle(g,300,200,100);**

Draws a solid (filled in) circle with a radius of 100 pixels whose center is located at the coordinate (300,200).

**Expo.drawOval(g,300,200,100,50);**

Draws an open oval with a horizontal radius of 100 pixels and a vertical radius of 50 pixels.

The center of this oval is located at the coordinate (300,200).

**Expo.fillOval(g,300,200,100,50);**

Draws a solid (filled in) oval with a horizontal radius of 100 pixels and a vertical radius of 50 pixels.

The center of this oval is located at the coordinate (300,200).

**Expo.drawRectangle(g,100,200,300,400);**

Draws an open rectangle whose upper-left-hand coordinate is (100,200) and whose lower-right-hand coordinate is (300,400).

**Expo.fillRectangle(g,100,200,300,400);**

Draws a solid (filled in) rectangle whose upper-left-hand coordinate is (100,200) and whose lower-right-hand coordinate is (300,400).

**Expo.drawArc(g,300,200,100,50,90,180);**

An “arc” is a piece of an “oval”. This will draw a specific piece of an open oval with a horizontal radius of 100 pixels and a vertical radius of 50 pixels. The center of this oval is located at the coordinate (300,200).

This oval is drawn clockwise starting at the 90 degree position (3:00) and stop at the 180 degree position (6:00).

This arc will be ¼ of an open circle.

**Expo.fillArc(g,300,200,100,50,90,360);**

A “filled arc” is a piece of a “filled oval”. This will draw a specific piece of a “filled-in” oval with a horizontal radius of 100 pixels and a vertical radius of 50 pixels. The center of this filled oval is located at the coordinate (300,200). This oval is drawn clockwise starting at the 90 degree position (3:00) and stop at the 360 degree position (12:00). This filled arc will draw ¾ of a filled arc which will resemble Pacman.

|  |  |
| --- | --- |
| **Lab04B Student Version** | **Do not copy this file, which is provided.** |
| **// Lab04Bst.java**  **// The Expo Graphics Program**  **// This is the student, starting version, of Lab 04B.**  **import java.awt.\*;**  **import java.applet.\*;**  **public class Lab04Bst extends Applet**  **{**  **public void paint(Graphics g)**  **{**  **// DRAW CUBE**  **// DRAW TARGET**    **// DRAW TRIANGLE**      **// DRAW SMILEY FACE**    **// DRAW JPII**    **}**  **}** | |

**60, 70, 80, 90 and 100 Point Versions**

The 60-point version displays any 1 of the 5 pictures below.

The 70-point version displays any 2 of the 5 pictures below

The 80-point version displays any 3 of the 5 pictures below

The 90-point version displays any 4 of the 5 pictures below

The 100-point version displays ALL 5 pictures below

**NOTE**

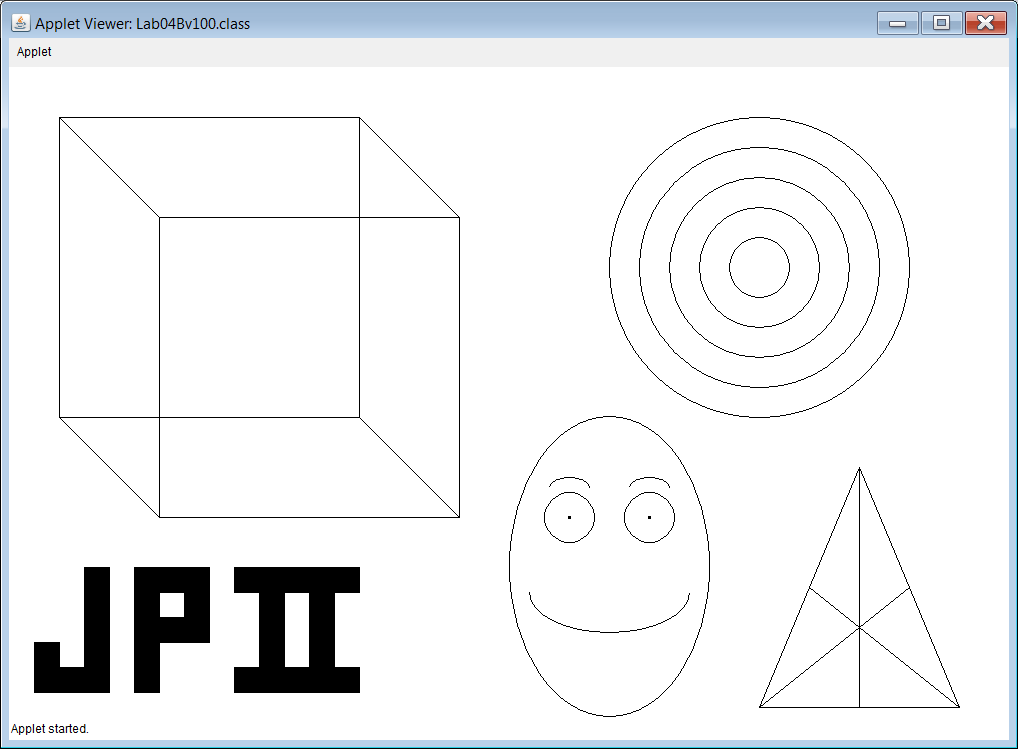
The pictures do not need to look exactly as they appear below. They should be very similar.

They also should not overlap with any other picture.

The **Target** must have 5 *concentric circles*. The spacing between all the circles should be the same.

The **Smiley Face** should be made up of 1 **oval**, 2 **circles**, 2 **points**, and 3 **arcs**.

In the **Triangle**, each of the 3 *interior lines* connects a *corner* to the *midpoint* of the line segment on the opposite side. The **Triangle** also needs to be *isosceles*.



**110 Point Version**

The 110 Point Version has all of the pictures from the 100 Point Version, with one improvement.

The target needs to be colored in. This requires using **fillCircle** commands instead of **drawCircle**.

The order of the colors – from largest/outermost circle to the smallest/innermost circle is:

**black**, **white**, **blue**, **red**, and finally **yellow** for the bull’s eye.

