Chapter 10 Getting Started with GUI Programming

1. See the section "The Java GUI Class Hierarchy."

2. See the Java 2 API online documentation.

3. The AWT components are heavy weight while the Swing components are lightweight.

4. You use the constructor of the JFrme class to create a frame. Use the setSize method to set the size of the frame. Use the getSize method to find the size of the frame.

5. You can add a component to a button.
Answer: False

   You can add a button to a frame.
Answer: True

   You can add a frame to a panel.
Answer: False

   You can add a panel to a frame.
Answer: True

   You can add any number of components to a panel, to a frame, or to an applet.
Answer: True

   You can derive a class from JPanel, JFrame, or JApplet.
Answer: True

6. The layout manager provides a platform-independent way to place components in a GUI interface. The default layout manager for the content pane of a frame is BorderLayout. Tthe default layout manager for a JPanel is FlowLayout.

7. No. There is no setTitle() method in JPanel. This method is defined for Frame; Panel is used to organize components. The panel itself is invisible.

8. Using FlowLayout, the components are arranged in the container from left to right in the order in which they were added. If one row becomes filled, a new row is started. To use a FlowLayout manager, you need to set the layout in a container to FlowLayout, such as with setLayout(new FlowLayout()). There is no limit on the number of components that can be added to a FlowLayout container.

9. The GridLayout manager arranges components in a grid (matrix) formation with the number of rows and columns defined by the constructor. The components are placed in the grid from left to right, starting from the first row, then the second, and so on, in the order in which they were added. To use a GridLayout manager, you need to set the layout in a container to GridLayout, such as with setLayout(new GridLayout()). The number of components you can add to a GridLayout container is limited by the grid size. (Encourage students to try adding more components into the container, to see what happens when the number of components exceeds the grid size. Interestingly, the column is enlarged.)

10. The BorderLayout manager divides the window into five areas: East, South, West, North, and Center. Components are added to a BorderLayout using add(String, Component), where String is "East", "South", "West", "North", or "Center". You can use one of the following two constructors to create a new BorderLayout:

    public BorderLayout(int hGap, int vGap)
    public BorderLayout

    You can add only one component into a section. If you need to add multiple components in a section, group the components in a panel, and add the panel into the section.

11. The y coordinate should increase and the x coordinate should remain unchanged.

12. Use the setColor() method to set the color and the setFont() method to set the font. The getColor() and getFont() methods get the current color and font.

13. See the section "Drawing Geometric Figures."

14. Draw a thick line from (10, 10) to (70, 30). You must draw several lines next to each other to create the effect of one thick line.
    Answer:

```
for (int i = 0; i < 10; i++)
  g.drawLine(10, 10 + i, 70, 30 + i);
```

    Draw a rectangle of width 100 and height 50 with the upper-left corner at (10, 10).
    Answer:

```
g.drawRect(10, 10, 100, 50);
```

    Draw a rounded rectangle with width 100, height 200, corner horizontal diameter 40, and corner vertical diameter 20.
    Answer:

```
g.drawRoundRect(10, 10, 100, 200, 40, 20);
```

Draw a circle with radius 30.

Answer:
g.drawOval(10, 10, 60, 60);

Draw an oval with width 50 and height 100.

Answer:
g.drawOval(10, 10 50, 100);

Draw the upper half of a circle with radius 50.

Answer:
g.drawArc(10, 10, 100, 100, 0, 180);

Draw a polygon connecting the following points: (20, 40), (30, 50), (40, 90), (90, 10), (10, 30).
Answer:

```
int x[] = {20, 30, 40, 90, 10};
int y[] = {40, 50, 90, 10, 30};
g.drawPolygon(x, y, x.length);
g.fillPolygon(x, y, x.length);
```

Draw a 3D cube like the one in Figure 8.28.
Answer:

```
public void paintComponent(Graphics g) {
  //draw the front rect
  g.drawRect(10, 60, 40, 40);

  //draw the back rect
  g.drawRect(30, 40, 40, 40);

  //connecting the corners
  g.drawLine(10, 60, 30, 40);
  g.drawLine(10, 100, 30, 80);
  g.drawLine(50, 60, 70, 40);
  g.drawLine(50, 100, 70, 80);
}
```

15. A WindowEvent is generated by an instance of the Window class or its subclass. JButton is not a subclass of Window, therefore, it cannot generate the WindowEvent. JButton can generate MouseEvent and ActionEvent.

16. To register a listener object, you invoke the source object's add*X*Listener's method; for example, button.AddActionListener(this). To implement a listener

interface, you add implements *X*listener and implement all the handlers in the listener's object.

17. See the Java 2 API documentation on java.awt.AWTEvent.

18. To override a method defined in the listener interface, you provide the code in the body of the method for handling the event. You need to override all the methods defined in the listener interface. If a handler is not used in the program, you can give it an empty body.

19. The paintComponent() method is defined in the Component class. The Java runtime system invokes it to paint things on JFrame, JApplet, and JPanel. This method cannot be invoked by the system or by the programmer. The system automatically invokes it whenever the viewing area changes. The programmer invokes it through invoking the repaint() method. The programmer should never directly invoke the paintComponent() method.