University of Stirling

Computing Science and Mathematics

ITNP001: Principles and Practice of Programming

Tutorial 3 (Week starting 12 October 2015)

This version contains sample answers in boxes

1. Assume that a program has set up a text field, referred to by global variable `number`.

   Write a section of program which will obtain the text entered by the user into `number`, will convert it to an `int`, will store the value in a new `int` variable called `num`, and will then test the value of `num`, and will cause a message to be output (using `drawString`) to indicate whether the value is positive, or is negative, or is zero.

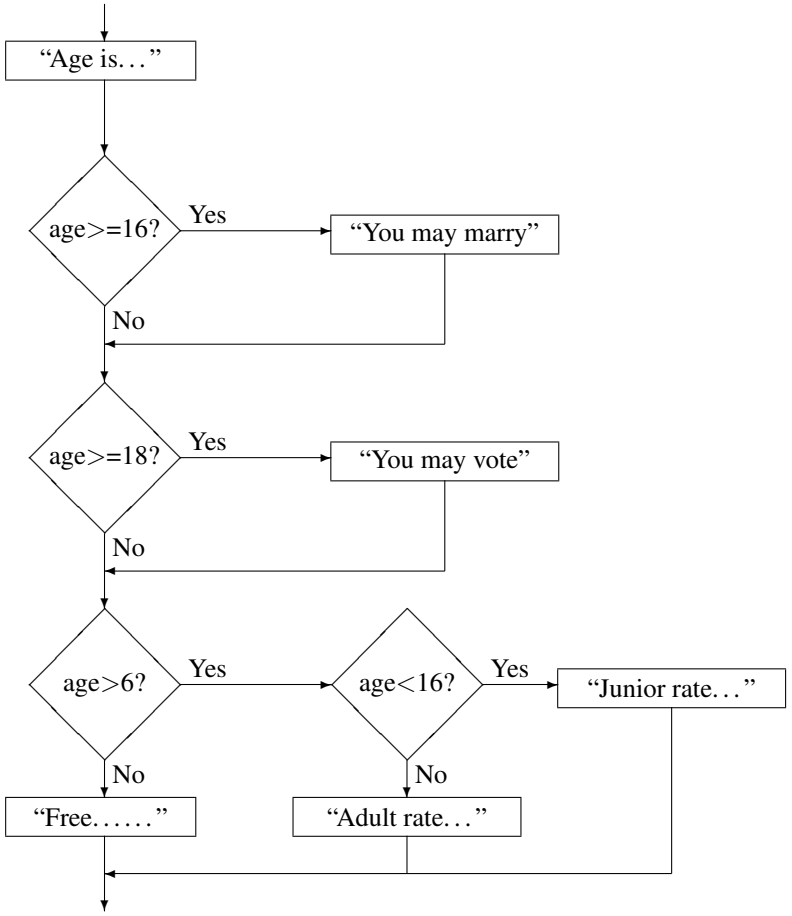
```
String input = number.getText();
int num = Integer.parseInt(input);
if (num > 0) {
   g.drawString("Positive", 20,40);
}
else {
   if (num < 0) {
      g.drawString("Negative", 20,40);
   }
   else {
      g.drawString("Zero", 20,40);
   }
}
```

2. Here is an `actionPerformed` method that involves several `if`s:

```
public void actionPerformed(ActionEvent e) {
   Graphics g = panel.getGraphics();
   int age = Integer.parseInt(ageEntry.getText());
   g.drawString("Age is " + age, 50, 50);
   if (age >= 16) {
      g.drawString("You may marry", 50, 70);
   }
   if (age >= 18) {
      g.drawString("You may vote", 50, 90);
   }
   if (age > 6) {
      if (age < 16) {
         g.drawString("Junior rate bus travel", 50, 110);
      }
      else {
         g.drawString("Adult rate bus travel", 50, 110);
      }
   }
   else {
      g.drawString("Free bus travel", 50, 110);
   }
}
```

   Draw a flow diagram to show the possible paths of execution through this method body.

Answer:



3. Overleaf you will find a fairly straighforward Java application with two sliders. Read it carefully. (It is available as example DiceTotal on the Java examples Web page.)

   (a) Which possible "events" does the program have event handler methods for, and in each case which method is the event handler? What are the other methods for?

   (b) What path does Java take through the program when the program *starts up*?

   (c) What path does Java take through the program *when either of its sliders is adjusted*?

   (d) How would the program behave if line `*1` were omitted by mistake?

   (e) How would the program behave if, instead, line `*2` were omitted by mistake?

   (f) How would the program behave if, instead, lines `*3` and `*4` were omitted by mistake?

   (g) How would the program behave if, instead, `slider2` in line `*5` were replaced by `slider1`?

(a) The program has a single event handler method: for the adjustment of a slider (method `stateChanged`). Methods `main` and `createGUI` are used when then program is first launched. The other two methods (`showThrows` and `showTotal`) are there to carry out sub-tasks on behalf of `stateChanged`, and are not called directly as event handlers.

(b) At program start-up: Java VM calls `main`, and runs through it, taking a temporary diversion through `createGUI` part way through, and then returns. The JVM then awaits an event.

(c) When a slider is adjusted: Java VM calls `stateChanged`, from which it calls `showThrows`, runs through that and returns to `stateChanged`, and then calls `showTotal`, runs through and returns to `stateChanged`, then returns from `stateChanged`. Then awaits next event.

(d) If line `*1` were omitted by mistake then `slider2` would not inform the program when it was adjusted — the slider could be dragged back and forth, but nothing would happen, even though the slider's actual setting *would* be changing. However `slider1` still functions properly, and when it is adjusted, `stateChanged` looks up the values of both `slider1` and `slider2` and displays them correctly.

(e) If line `*2` were omitted by mistake then the second slider would not appear on the screen.

(f) If lines `*3` and `*4` were omitted by mistake then the displayed text would not be erased before being redrawn — and numbers would be scribbled on top of each other.

(g) If `slider2` in line `*5` were replaced by `slider1` then `slider1` would supply the value for both dice — they would always be displayed as being the same, and adjustments to `slider2` would appear to be completely ignored!

SBJ 1st October 2015

```
// Dice throws program: offers the user two sliders (range 1-6),
// and displays the current selected values and their total

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;

public class Dice extends JFrame implements ChangeListener {

   private JSlider slider1, slider2;
   private JPanel panel;

   public static void main(String[] args) {
      Dice demo = new Dice();
      demo.setSize(250,250);
      demo.createGUI();
      demo.setVisible(true);
   }

   private void createGUI() {
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      Container window = getContentPane();
      window.setLayout(new FlowLayout());

      slider1 = new JSlider(1, 6, 3);
      slider1.addChangeListener(this);
      window.add(slider1);

      slider2 = new JSlider(1, 6, 3);
      slider2.addChangeListener(this);          Line *1
      window.add(slider2);                      Line *2

      panel = new JPanel();
      panel.setPreferredSize(new Dimension(200, 150));
      panel.setBackground(Color.white);
      window.add(panel);
   }

   public void stateChanged(ChangeEvent e) {
      Graphics g = panel.getGraphics();
      g.setColor(Color.white);                  Line *3
      g.fillRect(0, 0, 200, 150);               Line *4
      int value1 = slider1.getValue();
      int value2 = slider2.getValue();          Line *5
      g.setColor(Color.black);
      showThrows(g, value1, value2);
      showTotal(g, value1+value2);
   }

   private void showThrows(Graphics g, int die1, int die2) {
      g.drawString("Die 1 is "+die1, 40, 40);
      g.drawString("Die 2 is "+die2, 40, 60);
   }

   private void showTotal(Graphics g, int total) {
      g.drawString("Total is "+total, 40, 80);
   }
}
```