Model Answer

Benha University
2nd Term (2013) Final Exam
Class: 2nd Year Students
Subject: Object Oriented Programming

Faculty of Computers & Informatics
Date: 26/5/2013
Time: 3 hours
Examiner: Dr. Essam Halim

Instruction to students:

1. Language allowed to answer is the **English language**.
2. You should attempt **50** out of the **60** MCQ questions in **Section I**.
3. You should attempt **2** out of **Section II** (Section II comprises questions 2, 3, 4, and 5).
4. The exam paper is **12 pages long**, and is in **2 sections**.
5. The approximate allocation of **marks** is shown in brackets by the questions.
6. **Section I** contains multiple choice questions. Answer for the multiple choice questions should be written in the next table move it to the answer sheet.

Section I : Key answer for the multiple choice questions

Questions	1	2	3	4	5	6	7	8	9	10
Answer Key										
Questions	11	12	13	14	15	16	17	18	19	20
Answer Key										
Questions	21	22	23	24	25	26	27	28	29	30
Answer Key										
Questions	31	32	33	34	35	36	37	38	39	40
Answer Key										
Questions	41	42	43	44	45	46	47	48	49	50
Answer Key										
Questions	51	52	53	54	55	56	57	58	59	60
Answer Key										

Answer the following questions:

Section I **[Total 75]**

Q (1): Multiple choice questions

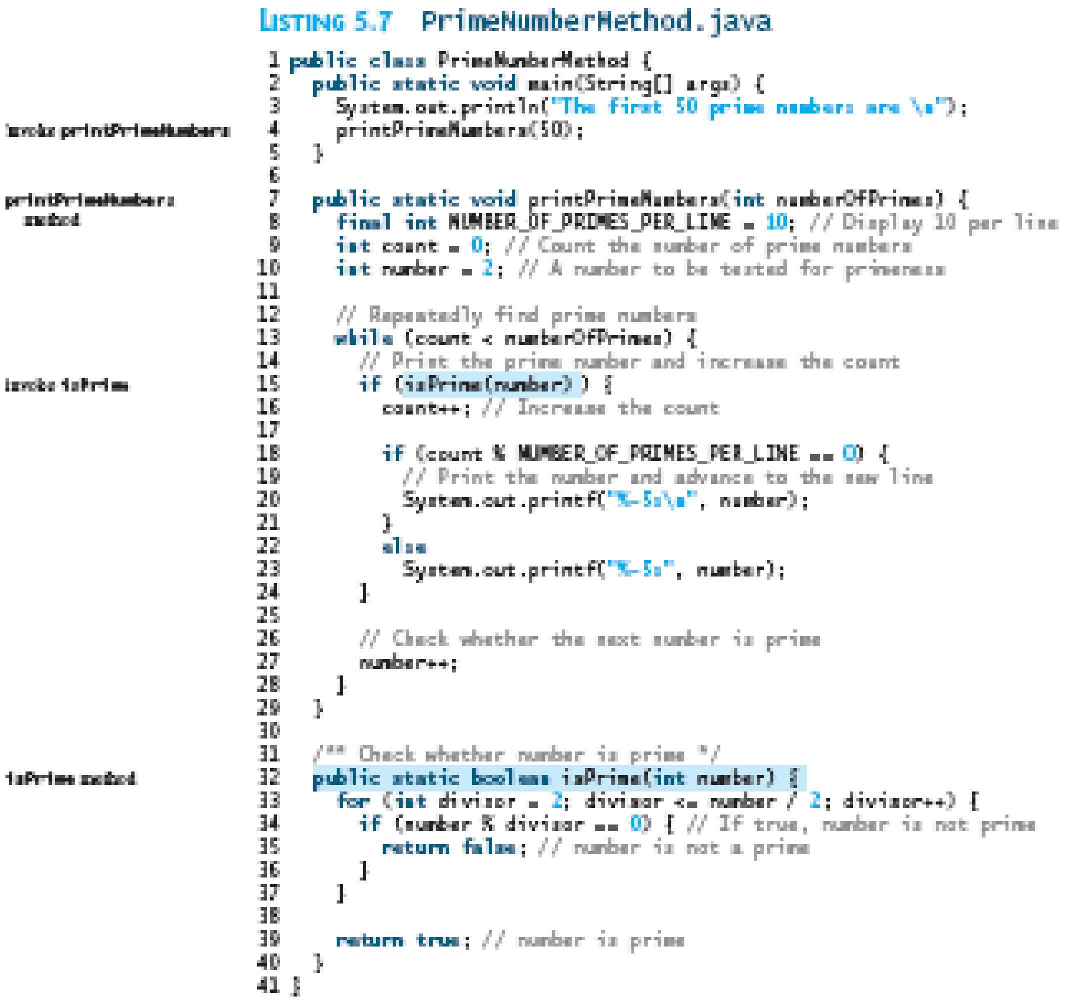
Answer only **50** of the following **60** multiple choice questions, by selecting the correct answer in each. Place the answer on the **special MCQ form**. **Each question [1.5 Mark]**

Section I : Key answer for the multiple choice questions										
Questions	1	2	3	4	5	6	7	8	9	10
Answer Key	E	B	E	B	E	BCD	E	C	B	BC
Questions	11	12	13	14	15	16	17	18	19	20
Answer Key	D	C	AB	B	A	C	AC	B	BCD	C
Questions	21	22	23	24	25	26	27	28	29	30
Answer Key	A	BD	D	D	ABC	D	C	ACE	B	D
Questions	31	32	33	34	35	36	37	38	39	40
Answer Key	B	A	B	B	B	A	C	C	B	A
Questions	41	42	43	44	45	46	47	48	49	50
Answer Key	B	C	CDE	BC	A	D	C	C	ABDE	B
Questions	51	52	53	54	55	56	57	58	59	60
Answer Key	AC	C	B	A	C	C	C	AD	C	C

Section IIyou should attempt 2 out of section II (questions 2, 3, 4 and 5) [Total 25]

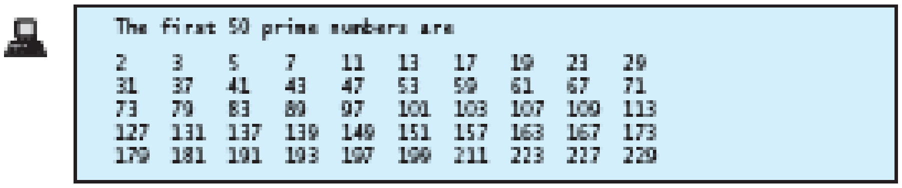
Q (2): write a program that **displays the first 50 prime numbers in five lines**, each of which contains **ten numbers**. [12.5 Marks]

Answer Q(2):


invoke printPrimeNumbers
printPrimeNumbers method
invoke isPrime
isPrime method

LISTING 5.7 PrimeNumberMethod.java

```java
public class PrimeNumberMethod {
  public static void main(String[] args) {
    System.out.println("The first 50 prime numbers are \n");
    printPrimeNumbers(50);
  }

  public static void printPrimeNumbers(int numberOfPrimes) {
    final int NUMBER_OF_PRIMES_PER_LINE = 10; // Display 10 per line
    int count = 0; // Count the number of prime numbers
    int number = 2; // A number to be tested for primeness

    // Repeatedly find prime numbers
    while (count < numberOfPrimes) {
      // Print the prime number and increase the count
      if (isPrime(number)) {
        count++; // Increase the count

        if (count % NUMBER_OF_PRIMES_PER_LINE == 0) {
          // Print the number and advance to the new line
          System.out.printf("%-5s\n", number);
        }
        else
          System.out.printf("%-5s", number);
      }

      // Check whether the next number is prime
      number++;
    }
  }

  /** Check whether number is prime */
  public static boolean isPrime(int number) {
    for (int divisor = 2; divisor <= number / 2; divisor++) {
      if (number % divisor == 0) { // If true, number is not prime
        return false; // number is not a prime
      }
    }

    return true; // number is prime
  }
}
```



```
The first 50 prime numbers are

2    3    5    7    11   13   17   19   23   29
31   37   41   43   47   53   59   61   67   71
73   79   83   89   97   101  103  107  109  113
127  131  137  139  149  151  157  163  167  173
179  181  191  193  197  199  211  223  227  229
```

Q (3): Write a program that demonstrates using panels as sub-containers. The program creates a user interface for a microwave oven, **as shown in Figure 1**. [12.5 Marks]

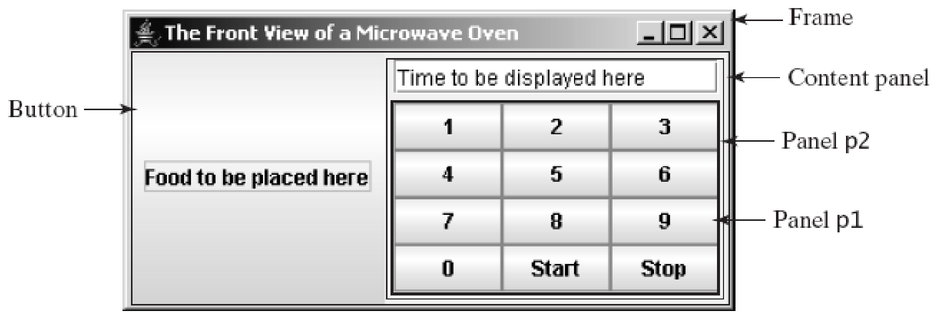


Figure 1

Answer Q(3):

LISTING 12.6 TestPanels.java

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

public class TestPanels extends JFrame {
  public TestPanels() {
    // Create panel p1 for the buttons and set GridLayout
    JPanel p1 = new JPanel();
    p1.setLayout(new GridLayout(4, 3));


    // Add buttons to the panel
    for (int i = 1; i <= 9; i++) {
      p1.add(new JButton("" + i));
    }

    p1.add(new JButton("" + 0));
    p1.add(new JButton("Start"));
    p1.add(new JButton("Stop"));

    // Create panel p2 to hold a text field and p1
    JPanel p2 = new JPanel(new BorderLayout());
    p2.add(new JTextField("Time to be displayed here"),
      BorderLayout.NORTH);
    p2.add(p1, BorderLayout.CENTER);

    // add contents into the frame
    add(p2, BorderLayout.EAST);
    add(new JButton("Food to be placed here"),
      BorderLayout.CENTER);
  }

  /** Main method */
  public static void main(String[] args) {
    TestPanels frame = new TestPanels();
    frame.setTitle("The Front View of a Microwave Oven");
    frame.setSize(400, 250);
    frame.setLocationRelativeTo(null); // Center the frame
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
  }
}
```

Q (4): (**Using the MessagePanel class**) Write a program that **displays four messages**, as shown in **<u>Figure 2</u>**. [12.5 Marks]

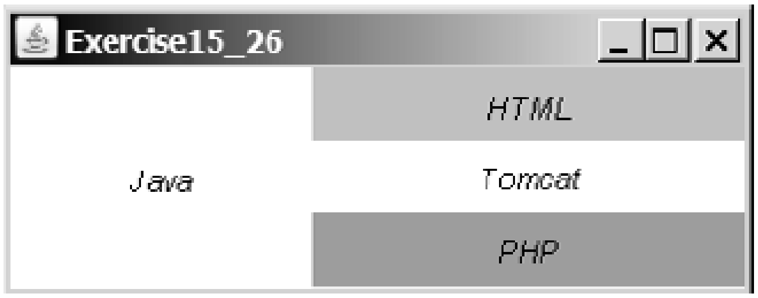


Figure 2

Answer Q(4):

```
importjavax.swing.*;
importjava.awt.*;

public class Exercise15_26 extends JFrame {
public Exercise15_26() {
MessagePanel m1 = new MessagePanel("Java");
MessagePanel m2 = new MessagePanel("HTML");
MessagePanel m3 = new MessagePanel("Tomcat");
MessagePanel m4 = new MessagePanel("PHP");

m1.setCentered(true);
m2.setCentered(true);
m3.setCentered(true);
m4.setCentered(true);

m1.setBackground(Color.white);
m2.setBackground(Color.cyan);
m3.setBackground(Color.white);
m4.setBackground(Color.green);

    Font font = new Font("TimezRoman", Font.ITALIC, 14);

m1.setFont(font);
m2.setFont(font);
m3.setFont(font);
m4.setFont(font);

JPanel p = new JPanel(new GridLayout(3, 1));
p.add(m2);
p.add(m3);
p.add(m4);

add(m1, BorderLayout.CENTER);
add(p, BorderLayout.EAST);
  }

public static void main(String[] args) {
    Exercise15_26 frame = new Exercise15_26();
frame.setSize(400, 400);
frame.setTitle("Exercise15_26");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setLocationRelativeTo(null); // Center the frame
frame.setVisible(true);
  }
}
```

Q (5): (**Using the GUI Components**) write a program that displays the following: [12.5 Marks]

A. **A message on a panel and uses two buttons**, Left and Right, to move the message on the panel to the left or right,
B. **Adds three check boxes** named Centered, Bold, and Italic, and
C. **Adds three radio buttons** named Red, Green, and Blue. **The layout of the UI is shown in Figure 3**.

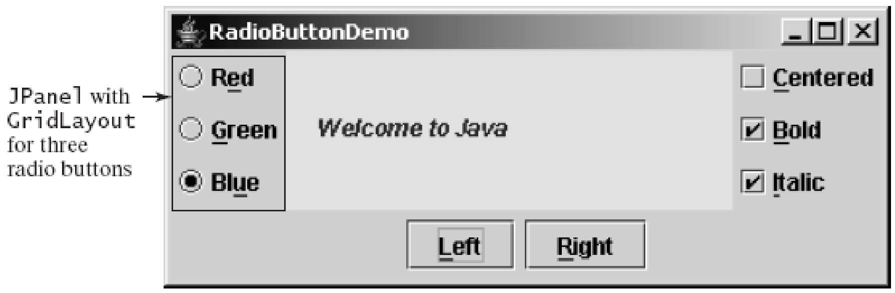


Figure 3

Answer Q(5):

```
importjava.awt.*;
importjava.awt.event.ActionListener;
importjava.awt.event.ActionEvent;
importjavax.swing.*;
public class ButtonDemo extends JFrame {
  // Create a panel for displaying message
protectedMessagePanelmessagePanel
    = new MessagePanel("Welcome to Java");
  // Declare two buttons to move the message left and right
privateJButtonjbtLeft = new JButton("<=");
privateJButtonjbtRight = new JButton("=>");
public static void main(String[] args) {
ButtonDemo frame = new ButtonDemo();
frame.setTitle("ButtonDemo");
frame.setLocationRelativeTo(null); // Center the frame
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setSize(250, 100);
frame.setVisible(true);
  }
publicButtonDemo() {
    // Set the background color of messagePanel
messagePanel.setBackground(Color.white);
    // Create Panel jpButtons to hold two Buttons "<=" and "right =>"
JPaneljpButtons = new JPanel();
jpButtons.add(jbtLeft);
jpButtons.add(jbtRight);
    // Set keyboard mnemonics
jbtLeft.setMnemonic('L');
jbtRight.setMnemonic('R');
    // Set icons and remove text
//    jbtLeft.setIcon(new ImageIcon("image/left.gif"));
//    jbtRight.setIcon(new ImageIcon("image/right.gif"));
//    jbtLeft.setText(null);
//    jbtRight.setText(null);
    // Set tool tip text on the buttons
jbtLeft.setToolTipText("Move message to left");
jbtRight.setToolTipText("Move message to right");
    // Place panels in the frame
setLayout(new BorderLayout());
add(messagePanel, BorderLayout.CENTER);
add(jpButtons, BorderLayout.SOUTH);
    // Register listeners with the buttons
jbtLeft.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
messagePanel.moveLeft();
      }
    });
jbtRight.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
messagePanel.moveRight();
      }
    });
  }}
```

```
importjava.awt.*;
importjava.awt.event.*;
importjavax.swing.*;

public class CheckBoxDemoextends ButtonDemo {
  // Create three check boxes to control the display of message
privateJCheckBoxjchkCentered = new JCheckBox("Centered");
privateJCheckBoxjchkBold = new JCheckBox("Bold");
privateJCheckBoxjchkItalic = new JCheckBox("Italic");

public static void main(String[] args) {
CheckBoxDemo frame = new CheckBoxDemo();
frame.setTitle("CheckBoxDemo");
frame.setLocationRelativeTo(null); // Center the frame
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setSize(500, 200);
frame.setVisible(true);
  }

publicCheckBoxDemo() {
    // Set mnemonic keys
jchkCentered.setMnemonic('C');
jchkBold.setMnemonic('B');
jchkItalic.setMnemonic('I');

    // Create a new panel to hold check boxes
JPaneljpCheckBoxes = new JPanel();
jpCheckBoxes.setLayout(new GridLayout(3, 1));
jpCheckBoxes.add(jchkCentered);
jpCheckBoxes.add(jchkBold);
jpCheckBoxes.add(jchkItalic);
add(jpCheckBoxes, BorderLayout.EAST);

    // Register listeners with the check boxes
jchkCentered.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
messagePanel.setCentered(jchkCentered.isSelected());
      }
    });
jchkBold.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
setNewFont();
      }
    });
jchkItalic.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
setNewFont();
      }
    });
  }

private void setNewFont() {
    // Determine a font style
intfontStyle = Font.PLAIN;
fontStyle += (jchkBold.isSelected() ? Font.BOLD :Font.PLAIN);
fontStyle += (jchkItalic.isSelected() ? Font.ITALIC :Font.PLAIN);

    // Set font for the message
    Font font = messagePanel.getFont();
messagePanel.setFont(
new Font(font.getName(), fontStyle, font.getSize()));
  }
}
```

```
importjava.awt.*;
importjava.awt.event.*;
importjavax.swing.*;

public class RadioButtonDemoextends CheckBoxDemo {
  // Declare radio buttons
privateJRadioButtonjrbRed, jrbGreen, jrbBlue;

public static void main(String[] args) {
RadioButtonDemo frame = new RadioButtonDemo();
frame.setLocationRelativeTo(null); // Center the frame
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setTitle("RadioButtonDemo");
frame.setSize(500, 200);
frame.setVisible(true);
  }

publicRadioButtonDemo() {
    // Create a new panel to hold check boxes
JPaneljpRadioButtons = new JPanel();
jpRadioButtons.setLayout(new GridLayout(3, 1));
jpRadioButtons.add(jrbRed = new JRadioButton("Red"));
jpRadioButtons.add(jrbGreen = new JRadioButton("Green"));
jpRadioButtons.add(jrbBlue = new JRadioButton("Blue"));
add(jpRadioButtons, BorderLayout.WEST);

    // Create a radio button group to group three buttons
ButtonGroup group = new ButtonGroup();
group.add(jrbRed);
group.add(jrbGreen);
group.add(jrbBlue);

    // Set keyboard mnemonics
jrbRed.setMnemonic('E');
jrbGreen.setMnemonic('G');
jrbBlue.setMnemonic('U');

    // Register listeners for check boxes
jrbRed.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
messagePanel.setForeground(Color.red);
      }
    });
jrbGreen.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
messagePanel.setForeground(Color.green);
      }
    });
jrbBlue.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
messagePanel.setForeground(Color.blue);
      }
    });

    // Set initial message color to blue
jrbBlue.setSelected(true);
messagePanel.setForeground(Color.blue);
  }
}
```

Good Luck ....