MID TERM – FPP REVIEW

Lessons for Midterm Examination

Lesson – 1 **-** Introduction to Java and IDE

Lesson – 2 **-** Introduction to Fundamental Programming Structures in Java

Lesson – 3 **-** Objects and Classes

Lesson – 4 **-** Recursion

Lesson – 5 – Inheritance, Interface and Polymorphism

Lesson – 6 **-** Building GUIs in Java with Swing

Lesson – 7 **-** Inner classes

Course Resources : www.online.cs.mum.edu => Choose CS390-2018-12A-12D(RM)
Reading Resources : Demo Code and Homework Solutions

Important points for the FPP Exam

1. The Midterm examination held on 12/8/2018 – Saturday Morning.
2. The midterm will be timed. It will begin at 9.30 am and will end at 12:00 noon.
3. Midterm should be closed book.
4. Bring Pencil/Pen, Eraser and necessary things. You are responsible to keep your wrting desk neat and clean. [Use waste paper to keep the pencil sharpened dust].
5. Mobile should be in Silent or Swich off mode. You are not allowed to keep the mobile. So bring backpack to keep your belongings. Keep the backpack infront of the dias.
6. Necessary hints, Syntax and API will be given in the question paper itself.
7. The midterm contains five programms.

   Programming Part [Lesson – 3 – 7]. It may contain programs for the concept of

   a. Class, Objects & Constructors, Overriding Object Methods [Complete coding – Lesson – 3 & Object methods → toString(), equals(), hashCode(), clone()]
   b. Recursion – Only Recursive method Implementation – Lesson - 4
   c. Abstract class & Polymorphism –Partial coding – Lesson – 5
   d. Interface & Polymorphism –Partial coding – Lesson – 5
   e. Swing – Event Handling – Partial Coding – Lesson-6 & 7
      a. Listener Implementation Choice is up to you.

Sample Review Questions

Concentrate on the following

1. **Lesson -3 (Need the knowledge of class and object creation, constructors, overloading, access modifiers, static, getters and setters, Immutable, final etc.,)**

   Write a main class to create an array of objects for the given TrailorRental class. Display the status of 5 objects. Write a static method called changeRental() to increase the 10% of monthly rental for all the objects. [Note : In changeRental() pass objects as an array argument with return type void].

```
class TrailorRental
{
    private double mRent;
    private String tNo;
    TrailorRental (String tNo, double mRent)
    {
            this.tNo= tNo;
            this.mRent = mRent;
    }
    public double getmRent() {
            return mRent;
    }
    public void setmRent(double mRent) {
            this.mRent = mRent;
    }
    public String toString() {
            return "[" + this.tNo + "," + this.mRent + "]";
    }
}

public class ModifyRent {

    public static void main(String[] args) {
            TrailorRental[] obj = { new TrailorRental("11A",650), new
TrailorRental("11J",550)};
            System.out.println("Original Rental for the Trailors : " );
            System.out.print(obj[0]);
            System.out.print(obj[1]);
            changeRental(obj);
            System.out.println("\nModified Rental for the Trailors : " );
            System.out.print(obj[0]);
            System.out.print(obj[1]);
    }
    public static void changeRental(TrailorRental[] objects)
    {
            double temp =0.0;
            if (objects!= null || objects.length >0)
            {

                    for(TrailorRental ind : objects)
```

```
                {
                        temp = ind.getmRent() + ind.getmRent() * 0.10;
                        ind.setmRent(temp);
                }

            }
        }

    }
```

Refer : Overriding Object Methods from Lesson – 5 Demo code and Homework. hashCode() use Java 8 way of implementation in your examination.

2. Lesson – 4 – Recursion

a. The two criteria for correct recursion. (Base case, and progress to the base case(Recursive Case))

Using Recursion to count the number of vowels in a string.

```
public class CountVowel {
        public static void main(String[] args)
        {
                String string;
                Scanner in=new Scanner(System.in);
                System.out.print("Enter any Word : ");
                string=in.next();
                System.out.println("Total Vowels in the String " + string + " is:
"+vowelCount(string));
}
        public static int vowelCount(String string)
        {
                char c;
                int count=0;
                if(string.length()==0)
                        return 0;
                else
                {
                        c=string.charAt(0);
                        if(c=='a'||c=='e'||c=='i'||c=='o'||c=='u')
                                count++;
                        return count + vowelCount(string.substring(1));
                }
        }
}
```

3. Lesson – 5 - Inheritance, Interfaces & Abstract classes

a. Inheritance, Abstract classes *and* Interfaces.(Pre Java 8):
Know the syntax rules to use the inheritance, making abstract class and how to implement an interface. Overriding a superclass method. Implementing an interface method. "super.method" and "super()" in a constructor. The difference between abstract classes and interfaces.

b. Be able to explain what polymorphism is and why it is used. Be able to do polymorphic type assignment (like Employee e = new Manager(. . .)).
c. Able to convert Non-OO code to OO-Code using Interface or abstract class. Refer closedcurve.bad and closedcurve.good package from Lesson-5.

Refer the examples of Lesson-5 Homework – 2, 3 & 4. In these exmples you have calculated the total salary of DeptEmployees(Prob2), Total salary of Employees(Prob3) and Plymorphically invoke Figure to print in Prob4.

Similar kind of task you have to perform in the exam.

4. Lesson – 6 –Swing & Lesson – 7 – Inner Class

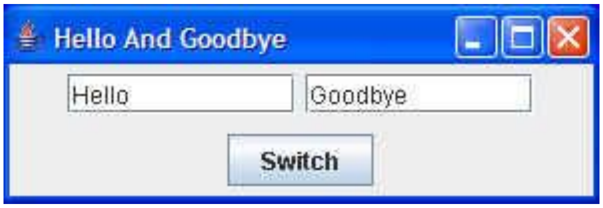
a. Know the 4 kinds of inner classes to implement Listeners.

Designing code will be given to you. Only concentrate on the following

a. Need to practice how to implement ActionListener[Anonymous or Inner or Current class(this) or Lambdas]
b. Need to know how to convert the TextField string input into int, float, double and vice versa.

Example :

Write Swing code which, when run, produces the following screen. Clicking the Switch button causes the values in the two text boxes to be switched (after the click, for example, “Goodbye” appears in the left box and “Hello” in the right box).



Hello

Goodbye

<u>Solution</u>

```
public class HelloGoodbye extends JFrame {
    JPanel mainPanel, topPanel, centerPanel;
    JTextField leftText, rightText;

    public HelloGoodbye() {
        setTitle("Hello Goodbye");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        defineTopPanel();
        defineCenterPanel();
        mainPanel = new JPanel();
        mainPanel.setLayout(new BorderLayout());
        mainPanel.add(topPanel, BorderLayout.NORTH);
        mainPanel.add(centerPanel, BorderLayout.CENTER);
        getContentPane().add(mainPanel);
        pack();
    }
    public void defineTopPanel(){
        topPanel = new JPanel();
        topPanel.setLayout(new FlowLayout(FlowLayout.CENTER));
        leftText = new JTextField(10);
        leftText.setText("Hello");
        rightText = new JTextField(10);
        rightText.setText("Goodbye");
        topPanel.add(leftText);
        topPanel.add(rightText);
    }

// Concentrate on highlighted part

    public void defineCenterPanel() {
        centerPanel = new JPanel();
        centerPanel.setLayout(new FlowLayout(FlowLayout.CENTER));
        JButton b = new JButton("Switch");
        b.addActionListener(new ButtonListener());
        centerPanel.add(b);
    }

    class ButtonListener implements ActionListener {
        public void actionPerformed(ActionEvent evt) {
            String left = leftText.getText();
            String right = rightText.getText();
            leftText.setText(right);
            rightText.setText(left);
        }
    }

    public static void main(String[] args) {
                        HelloGoodbye hg = new HelloGoodbye();
                hg.setVisible(true);
    }
}
```