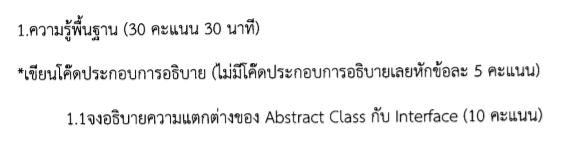
**OOP Final 55**



public abstract class A

{

public abstract void do() ;

}

public interface B

{

void do()

}

1. abstract class ทุก method ไม่ได้เป็น abstract method

interface ทุก method เป็น abstract method

2. abstract class ใช้ extends ในการสืบทอด

Interface ใช้ implements ในการสืบทอด



สามารถใช้ตัวแปรร่วมกับ class อื่นได้ง่ายกว่าการแยก class

Public class A{

Private Int x,y;

Public A()

{}

Public static class B

{

Public static void main(String[] args)  
 {

System.out.println(x);

}

}  
}



public static void main(String[] args)

{

ArrayList<Shape> s = new ArrayList<Shape>();

Triangle t = new Triangle();

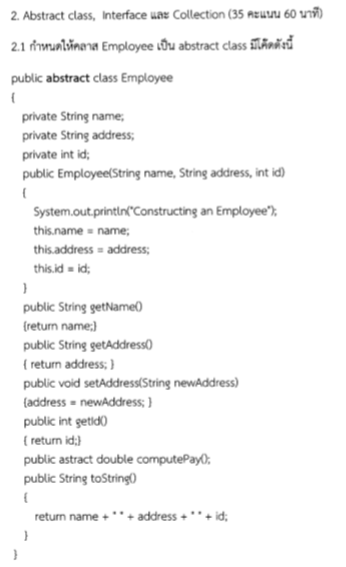
Rectangle r = new Rectangle();

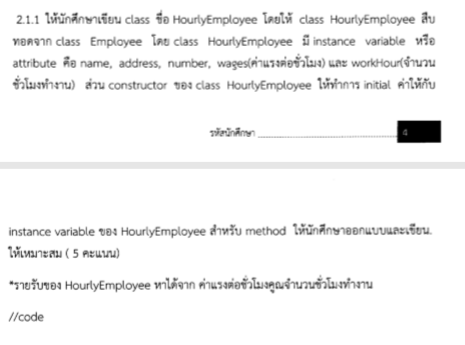
s.add(t);

s.add(r);

}

class ที่ extends class อื่นมา สามารถเก็บใน Collection ของ super class ที่ class นั้นสืบทอดมาได้





Public class HourlyEmployee extends Employee

{

Private int wages;

Private double workhour;

Public HourlyEmployee(String name,String address,int ID,int wages,double workHour)

{

Super(name,address,Id);

This.wages = wages;

This.workHour = workhour;

}

Public double computePay()

{

Return wages\*workhour;

}

Public int getWages()

{

Return wages;

}

Public void setWages(int wages)

{

This.wages = wages;

}

Public double getWorkHour()

{

Return this.workHour;

}

Public void setWorkHour(double workhour)

{

This.workHour = workhour;

}

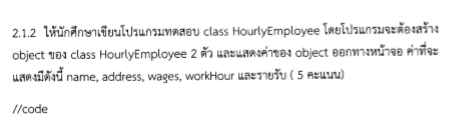
Public String toString()

{

Return super.toString() +”wages = ” + wages + “workhour = ” + workhour;

}

}



**Public class Main**

**{**

**Public static void main(String[] args)**

**{**

**HourlyEmployee e1 = new HourlyEmployee(“A”,”Phuket”,”1”,”100”,”5”);**

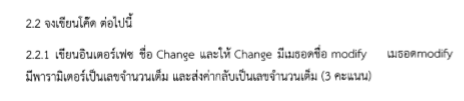
**HourlyEmployee e2 = new HourlyEmployee(“B”,”Phuket”,”1”,”100”,”5”);**

**System.out.println(e1 + “income = ” + e1.computePay());**

**System.out.println(e2 +”income = “ +e2.computePay());**

**}**

**}**



**public interface Change**

**{**

**int modify(int value);**

**}**



**public class Product implements Change**

**{**

**public int modify(int value)**

**{**

**return value\*2;**

**}**

**}**



**public class Sum implements Change**

**{**

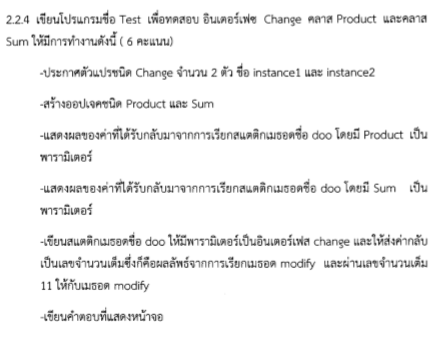
**public int modify(int value)**

**{**

**return value+2;**

**}**

**}**



**public class Test**

**{**

**public static void main(String[] args)**

**{**

**Change instance1 = new Product();**

**Change instance2 = new Sum();**

**System.out.println(doo(instance1));**

**System.out.println(doo(instance2));**

**}**

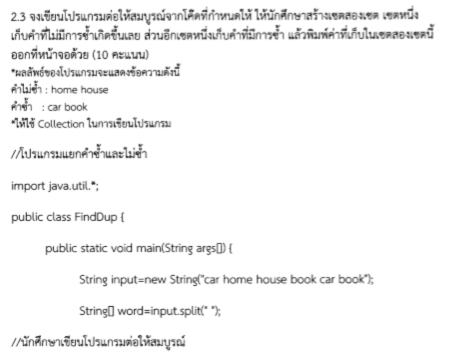
**public static int doo(Change c)**

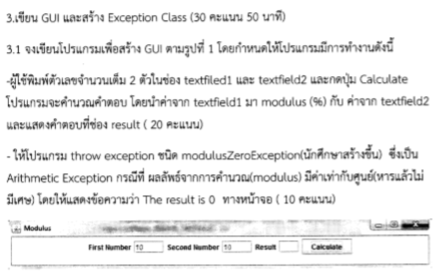
**{**

**return c.modify(11);**

**}**

**}**





**import java.awt.\*;**

**import javax.swing.\*;**

**import java.awt.event.\*;**

**import java.util.\*;**

**public class ModulusGUI extends JFrame**

**{**

**private JLabel jlabel1;**

**private JLabel jlabel2;**

**private JLabel jlabel3;**

**private JTextField jtf1;**

**private JTextField jtf2;**

**private JTextField jtf3;**

**private JButton jbtn;**

**private JPanel jpn;**

**public ModulusGUI()**

**{**

**jlabel1 = new JLabel("First Number ");**

**jlabel2 = new JLabel("Second Number ");**

**jlabel3 = new JLabel("Result ");**

**jtf1 = new JTextField(10);**

**jtf2 = new JTextField(10);**

**jtf3 = new JTextField(10);**

**jbtn = new JButton("Calculate");**

**jpn = new JPanel();**

**jpn.setLayout(new FlowLayout(FlowLayout.LEFT,5,5));**

**jpn.add(jlabel1);**

**jpn.add(jtf1);**

**jpn.add(jlabel2);**

**jpn.add(jtf2);**

**jpn.add(jlabel3);**

**jpn.add(jtf3);**

**jbtn.addActionListener(new ActionListener(){**

**public void actionPerformed(ActionEvent e)**

**{**

**try{**

**int i1 = Integer.parseInt(jtf1.getText());**

**int i2 = Integer.parseInt(jtf2.getText());**

**int ans = mod(i1,i2);**

**jtf3.setText(ans+"");**

**}**

**catch(NumberFormatException ne)**

**{**

**JOptionPane.showMessageDialog(null,"Input Number only");**

**}**

**catch(ModulusZeroException ae)**

**{**

**JOptionPane.showMessageDialog(null,"Error Divide by zero");**

**}**

**}**

**});**

**Container c = getContentPane();**

**c.setLayout(new BorderLayout());**

**c.add(jpn,BorderLayout.CENTER);**

**c.add(jbtn,BorderLayout.EAST);**

**this.setSize(750,100);**

**this.setLocationRelativeTo(null);**

**this.setDefaultCloseOperation(this.EXIT\_ON\_CLOSE);**

**this.setVisible(true);**

**}**

**public static int mod(int x,int y) throws ModulusZeroException**

**{**

**if(y==0)**

**throw new ModulusZeroException();**

**return x%y;**

**}**

**}**

**public class ModulusZeroException extends ArithmeticException**

**{**

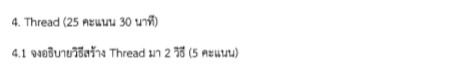
**public ModulusZeroException()**

**{**

**super();**

**}**

**}**



**1. การ extends Thread แล้วทำการ override method run()**

**public class ThreadA extends Thread{**

**public void run()**

**{**

**System.out.println(“Hello”);**

**}**

**}**

**ใน main**

**ThreadA ta = new ThreadA();**

**ta.start();**

**2. การ implements Runnable แล้วทำการ override method run()**

**public class ThreadB implements Runnable{**

**public void run()**

**{**

**System.out.println(“Hello”);**

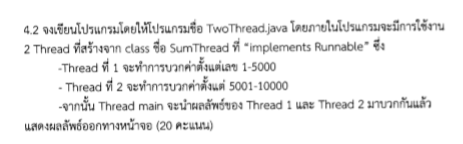
**}**

**}**

**ใน main**

**Thread ta = new Thread(new ThreadB());**

**ta.start();**



**public class TwoThread**

**{**

**public static void main(String[] args)**

**{**

**SumThread s1 = new SumThread(1,5000);**

**SumThread s2 = new SumThread(5001,10000);**

**Thread t1 = new Thread(s1);**

**Thread t2 = new Thread(s2);**

**try**

**{**

**t1.start();**

**t2.start();**

**t1.join();**

**t2.join();**

**}**

**catch(InterruptedException e){}**

**System.out.println("Sum = " + (s1.getSum()+s2.getSum()));**

**}**

**}**

**public class SumThread implements Runnable**

**{**

**private int start;**

**private int finish;**

**private int sum;**

**public SumThread(int start,int finish)**

**{**

**this.start = start;**

**this.finish = finish;**

**sum = 0;**

**}**

**public void run()**

**{**

**for(int i=start;i<=finish;i++)**

**{**

**sum+=i;**

**}**

**}**

**public int getSum()**

**{**

**return sum;**

**}**

**}**