|  |
| --- |
| Alexander Drabek |
| E-Portfolio |
| 5 Examples in Java |

|  |
| --- |
| Alexander Drabek  2013-02-18 |

|  |  |
| --- | --- |
| Example 1 –From Exercise sheet 2(Week2 Ex1) | |
| **Self-Reflection:**  The task was clear structured and consists of 7 points. In order to test the program, I have looked on the each point of the requirements and I have commented “1.x checked”. Where ‘x’ is the number of the requirement.  Skeleton of the class was provided and my task was to create a program which will convert unit from gallons to liters. I gave the output using System.out.println(); . | |
| **Code with comments:**  **VolumeConversion class :**  /\*\* Alexander Drabek-1219460  \* Exercise 1 from Laboratory Practice Notes2.pdf  \*/  public class VolumeConversion  {  public static final double gallonToLiterConversion=4.546;//1.1 checked this value will not change in future.  public static void main(String [] args)  {  // Declare variables  double liter;//1.3 checked creating variable ,type double  int numberOfGallons=14; //1.2 checked  String someTxt= new String("The number of liters in ");  //1.4 checked, some text in string object.  String someTxtTwo= new String(" gallons is ");  //1.5 checked, second string with a text  // Perform conversion calculation  liter=numberOfGallons\*gallonToLiterConversion;  //1.6 checked  // Complete output with appropriate text  System.out.println(someTxt+numberOfGallons+someTxtTwo+liter);//1.7 checked  }  } | **Test result**:C:\Documents and Settings\1219460\Desktop\ex1.JPG |

|  |  |  |  |
| --- | --- | --- | --- |
| Example 2 –From Exercise sheet 6(Ex3) | | | |
| **Self-Reflection:**  Our task was to develop a program which will have most of his code in the main() method. The program should take a number from user using Scanner class and check if it is in a valid range (in this case 1 to 7) It should also, assign appropriate day in the week to the given number by just printing it out.  In case of typing number out of a range program will display error message using System.out.println(“ ”); . | | | |
| **Code with comments: -Week class:**  /\*\*  \* Develop the following program in the main() method only.  \*/  import java.util.Scanner;  // importing Scanner class, from java.util package  public class Week  {  public static void main(String[] args){  Scanner inputVarible= new Scanner(System.in);  //inputVarible is an object with the methods that allow //capture data using keyboard.  System.out.println("I am waiting for the number in range of 1 to 7");//actually it is not waiting yet.  int number=inputVarible.nextInt(); //using Scanner class //to Input data from keyboard and store result in number //variable.  if (number<=7 && number>=1)  System.out.println("OK");//just saying everything is OK  else  System.out.println("Error!");//number is out of range ( week is 1 to 7 )!  /\*\*if user type 1 and enter it will execute case called 1.  \* if user put other number it will execute appropriate case according to the number given  \* if number is out of range switch will do nothing because //there is no default action  \*/  switch (number){  case 1 : System.out.println("1 is Monday");//printing to Screen  break; //really important if we want to execute only statements from 1 case.  case 2 : System.out.println("2 is Tuesday");  break;  case 3 : System.out.println("3 is Wednesday");  break;  case 4 : System.out.println("4 is Thursday");  break;  case 5 : System.out.println("5 is Friday");  break;  case 6 : System.out.println("6 is Saturday");  break;  case 7 : System.out.println("7 is Sunday");  break; } } } //combine all endings in order to fit to the page. | | | **Test Results:**  C:\Documents and Settings\1219460\Desktop\exampl2.JPG |
| Example 3 –From Exercise sheet 11(Ex2) | | |
| **Self-Reflection:**  One of the difficulties I faced in this task, was inheritance and polymorphism errors.  When the program gets longer, I needed to be careful from where I wanted to access the particular method or variable. In this task, I have noticed and corrected important mistake which was declaring and initialising variables inside the constructor. It is improper solution and leads to errors.  This task required from me understanding of various layouts and in general how to use GUI in Java.  I have used Border layout.  In addition, I created button “Exit” which will do pretty the same work as closing the program using default X button on the top. I used dispose() method which in future tasks will be probably used to manage memory in more efficient way. I also re-used my code from previous weeks.  The task required from me producing two classes. Account class and orchestrating /Demo class. | | |
| **Code with comments:**  **Account Class:**  /\*\*  \* Definition of Account. Set of Methods (and variables);  \* that future Account object can perform ( and change a state of a variables)  \*/  public class Account {  //instance variables.  // declaring as private because of limiting an access //to them.  private int openingBalance;  private int currentBalance;  private int creditLimit;  //following the rule:"private variable and public //methods"  public static int numOfAccounts = 0;  //when start 0 account created.  public String accountHolderName;  public String accountHolderAddress;    public Account( int startAmount, int balance, int credit, String name,String adress) {  //constructor with parameters  openingBalance = startAmount;  currentBalance = balance;  creditLimit = credit;//passing the parameters  numOfAccounts++;//increasing number of //account each time object is created  accountHolderName= name;  accountHolderAddress=adress;  }  public void setBalance( int amountBalance) {  currentBalance = amountBalance;//simple set method  }  public int getBalance() {  return currentBalance;//simple get method  }  public void setCreditLimit(int amountCredit) {  creditLimit = amountCredit;  }  public int getCreditLimit() {  return creditLimit;  }  public void showData() {  //showing all data ,  System.out.println("Name is "+accountHolderName+" and address is "+accountHolderAddress+" \n balance = " + currentBalance + " credit = " + creditLimit);  }  public void setAccountHolderAddress(String setsAdress) {  accountHolderAddress=setsAdress ;  }  public String getAccountHolderAddress() {  return accountHolderAddress;  }  public void setAccountHolderName(String setsName) {  accountHolderName=setsName ;  }  public String getAccountHolderName() {  return accountHolderAddress;  }  }  **Account demo Class:**  /\*\*  \* Demo class presenting Account objects using GUI.  \*/  import java.awt.\*;//Abstract Window Toolkit  import java.awt.event.\*;  // Provides interfaces and classes for dealing with //different types of events fired by AWT components import javax.swing.\*;//improved awt classes still //dependent on awt.  public class AccountDemoEx2Week11 extends JFrame  {  JTextField credit;//declaring variables in the class not inside a constructor!  JTextField balance;  public AccountDemoEx2Week11(){  //constructor  super();//invoking super class constructor  // get the container object  Container containerOne = getContentPane();//returns the contentPane object and assigning into containerOne  // set the layout of the container object  containerOne.setLayout(new BorderLayout());//setting type of layout  // now add the buttons  balance= new JTextField("Current Balance");  credit = new JTextField("Current credit limit");  JButton save=new JButton("Save");  JButton exite = new JButton("Exit");  containerOne.add(save, BorderLayout.NORTH);//adding and assigning things to the position on GUI(container)  containerOne.add(exite, BorderLayout.SOUTH);  containerOne.add(balance, BorderLayout.WEST);  containerOne.add(credit, BorderLayout.EAST);  //assigning the position in layout to credit text filed.  save.addActionListener(new ActionListener()//action listener for save button  {  public void actionPerformed(ActionEvent e)  {//all this things will be executed when the save button will be clicked.  System.out.println("Object " + e.getActionCommand());  if (e.getActionCommand().equals("Save"))  {  String bal=balance.getText();  int bala=Integer.parseInt(bal);//balance in int  String cred = credit.getText();  int credi=Integer.parseInt(cred);  Account createdByGuiAccount =new Account(0, bala, credi,"Default\_Name","Poland ,Katowice[default]");  //we are creating an object and passing some random/default data because task is only about the balance and credit.  createdByGuiAccount.showData();//we are showing what was just done. Including new balance and credit limit.  }  }  });  exite.addActionListener(new ActionListener()//action listener for exite button  {  public void actionPerformed(ActionEvent e)  {//all this things will be executed when the exite button will be clicked.  System.out.println("Object " + e.getActionCommand());  if (e.getActionCommand().equals("Exit"))  {  dispose();//clear the memory  }  }  });  setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE); //if we want to close using default X button, it will actually kill program.  pack();  //resizes the Window to the minimum //size to satisfy the preferred size of each of the //components in the layout    setVisible(true);// shows the window  }  public static void main(String [] args) {  AccountDemoEx2Week11 showTime = new AccountDemoEx2Week11();//creating an object ,GUI and all this things.  }  } | **Test result**:  C:\Documents and Settings\1219460\Desktop\example3.JPG | |

|  |  |
| --- | --- |
| Example 4 –From Exercise sheet 14 (Ex1) | |
| **Self Reflection:**  Naming for the classes are slightly different; instead of Holiday, I have HolidaySimpleWay and corresponding orchestrating class. In addition, I have provided class premier and corresponding orchestrating class.  I sometimes omitted some of the letters in the word “orchestrating” because I am dyslexic.  During creation process of this classes, I was trying to approach this task in Object Oriented way.  Picture 1.1 presents orchestrating class of Holiday.  Picture 1.2 presents orchestrating class of Premier.  This orchestrating class is creating one Holiday object and one Premier object from hard coded data and output all the objects instance variables.  I was struggling with simplifying the orchestrating classes and hardcoded data in it. | |
| **Code with comments:**  HolidaySimpleWay class:  /\*\*  \* I understand using this .  \* It is not the same variable!  \*/  public class HolidaySimpleWay {  private String startYear;  private String startMonth;  private String startDay;  private String endYear;  private String endMonth;  private String endDay;  //date will be presented dd-mm-yyyy or similar-//(collection of variable)  private String price;  //prefer to declare as int and create a second variable to carry currency.  private String place;    public HolidaySimpleWay()    {  //private String startYear;  //default data, not so important  startMonth="1";  startDay="1";  startYear="2013";//should be set to current year.  endYear="2013";  endMonth="12";  endDay="28";//because of February, normally i would put 30/31  price="0";  place="Home";    }    public void setPlace(String place) {  this.place = place; //this method is for setting a place where user2 want to spend holidays  }  public String getPlace() {  return place;  }    public void setPrice(String price) {  this.price = price;  }  public String getPrice() {  return price;  }    public void setStartYear(String startYear ) {  this.startYear = startYear;  }  public String getStartYear() {  return startYear;  }  public void setStartMonth(String startMonth) {  this.startMonth = startMonth;  }  public String getStartMonth() {  return startMonth;  }    public void setStartDay(String startDay) {  this.startDay = startDay;  }    public String getStartDay() {  return startDay;  }    public void setEndYear(String endYear ) {  this.endYear = endYear;  }  public String getEndYear() {  return endYear;  }    public void setEndMonth(String endMonth) {  this.endMonth = endMonth;  }  public String getEndMonth() {  return endMonth;  }    public void setEndDay(String endDay) {  this.endDay = endDay;  }  public String getEndDay() {  return endDay;  }  public String showAllData()  {  //showing all data, accessing through get methods  return ("Holiday will start on : "+ getStartDay()+"-"+getStartMonth()+"-"+getStartYear()+" and end on : "+ getEndDay()+"-"+ getEndMonth()+"-"+getEndYear()+"\n The holiday will took place at "+ getPlace()+" whole cost will be "+getPrice() );  }  }  **Premier class:**  /\*\*  \* @Alexander Drabek  \* @version 1.0  \*/  public class Premier extends HolidaySimpleWay//subclass of HolidaySimpleWay  {  // instance variables  private String hotelName;  private String hotelGrade;  private String resortName;  public Premier()  {  super();  hotelName="";//we don't know the valueyet so ""  hotelGrade="";//we don't know the value yet so ""  resortName="";//we don't know the value yet so ""  }  public void setHotelName( String hotelName)  {  this.hotelName= hotelName;  //i am aware that this.hotelName it is not the same variable as hotelName  //for me it is more clear in this way .  }    public void setHotelGrade( String hotelGrade)  {  this.hotelGrade= hotelGrade;  }  public void setResortName( String resortName)  {  this.resortName= resortName;  }  public String getHotelName(){  return hotelName;  }  public String getHotelGrade(){  return hotelGrade;  }  public String getResortName(){  return resortName;  }  //I had showAllDataPremier and showAllDataHoliday() but now i changed to the same name and let overriding working.  public String showAllData(){  return (super.showAllData()+".\n The hotel name is "+getHotelName()+" and it grade is "+getHotelGrade()+" .The hotel is placed in "+ getResortName());  }  }  **ochastreingSimpleWayTwoObjects class:**  /\*\*  \* Write a description of class ochastreingSimpleWay here.  \* HolidaySimpleWay  \* @author (your name)  \* @version (a version number or a date)  \*/  public class ochastreingSimpleWayTwoObjects  {  public ochastreingSimpleWayTwoObjects()  {  //empty  }  public static void main(String [] args)  {  System.out.println("Hello in week14...");//Sometimes i like to just print some text when starting.  //It helps with "where is my program" problem which i can not even clasify as an error.    //Declaring objects.  HolidaySimpleWay object1= new HolidaySimpleWay();  HolidaySimpleWay object2= new HolidaySimpleWay();  //object1 parameters  object1.setPlace("Katowice");  object1.setPrice("1000");  object1.setStartYear("2012");  object1.setStartMonth("1");  object1.setStartDay("20");  object1.setEndYear("2012");  object1.setEndMonth("2");  object1.setEndDay("20");    //object2 parameters  object2.setPlace("Sirilanka");  object2.setPrice("9999000");  object2.setStartYear("2012");  object2.setStartMonth("4");  object2.setStartDay("20");  object2.setEndYear("2012");  object2.setEndMonth("6");  object2.setEndDay("19");  System.out.println("object1");  System.out.println(object1.showAllData());  System.out.println("object2");  System.out.println(object2.showAllData());  }  }  **OrchestratingPremier class:**  /\*\*  \* Write a description of class OrchestratingPremier here.  \*  \* @Alexander Drabek)  \* @version 1.0  \*/  public class OrchestratingPremier  {  public OrchestratingPremier()  {  }  public static void main(String [] args)    {  System.out.println("Hello in Orchestrating Class");  Premier object1= new Premier();  //creating new object  HolidaySimpleWay object2= new HolidaySimpleWay();  //giving hardcoded data.  object1.setPlace("Katowice");  object1.setPrice("1000");  object1.setStartYear("2012");  object1.setStartMonth("1");  object1.setStartDay("20");  object1.setEndYear("2012");  object1.setEndMonth("2");  object1.setEndDay("20");    object1.setHotelName("Alex's Hotel");  object1.setHotelGrade("5");  object1.setResortName("Alex Resort");  System.out.println("Object1 of a class Premier");  System.out.println(object1.showAllData());    System.out.println("");  //object2 parameters, hardcoded data  object2.setPlace("Sirilanka");  object2.setPrice("9999000");  object2.setStartYear("2012");  object2.setStartMonth("4");  object2.setStartDay("09");  object2.setEndYear("2012");  object2.setEndMonth("6");  object2.setEndDay("19");  System.out.println("Object2 of a class Holiday");  System.out.println(object2.showAllData());  } } | **Test result**:  Picture 1.1  C:\Users\Study\Desktop\ex4-part1.jpg  Picture 1.2C:\Users\Study\Desktop\example 4 part2.png |

|  |  |
| --- | --- |
| Example 5 –From Exercise sheet 15(Ex1) | |
| **Self Reflection:**  I explained equal() method inside program. I made good looking GUI using JLabels and GridLayout.  First column contains labels and the second contains text fields. Previously, I was using text inside JTextField to describe the meaning of the field but It could cause run-time errors when the user create objects without changing default “data”.  Program contains fields for all data .If user want to change data for second object or correct them, user need to just type data again and click appropriate button.  Picture 1.1 shows program behavior when objects are equal.  Picture 1.2 shows program behavior when objects are different. | |
| **Code with comments:**  **HolidaySimpleWay class:**  /\*\*  \* I understand "this" and I am using it properly.  \* It is not the same variable!  \*/  public class HolidaySimpleWay {  private String startYear;  private String startMonth;  private String startDay;  private String endYear;  private String endMonth;  private String endDay;//date will be presented dd-mm-yyyy //or similar-(collection of variable)  private String price;//prefer to declare as int and create a //second variable to carry currency.  private String place;    public HolidaySimpleWay()    {  //private String startYear;  //default data, not so important  startMonth="1";  startDay="1";  startYear="2013";//should be set to current year.  endYear="2013";  endMonth="12";  endDay="28";//because of February, normally i would put 30/31  price="0";  place="Home";    }    public void setPlace(String place) {  this.place = place; //this method is for setting a place where user2 want to spend holidays  }  public String getPlace() {  return place;  }    public void setPrice(String price) {  this.price = price;  }  public String getPrice() {  return price;  }    public void setStartYear(String startYear ) {  this.startYear = startYear;  }  public String getStartYear() {  return startYear;  }  public void setStartMonth(String startMonth) {  this.startMonth = startMonth;  }  public String getStartMonth() {  return startMonth;  }    public void setStartDay(String startDay) {  this.startDay = startDay;  }    public String getStartDay() {  return startDay;  }    public void setEndYear(String endYear ) {  this.endYear = endYear;  }  public String getEndYear() {  return endYear;  }    public void setEndMonth(String endMonth) {  this.endMonth = endMonth;  }  public String getEndMonth() {  return endMonth;  }    public void setEndDay(String endDay) {  this.endDay = endDay;  }  public String getEndDay() {  return endDay;  }  public boolean equals(HolidaySimpleWay obj)// parameter will be an object of HolidaySimpleWay  {  //I will compare all things to assure that object is the same  //I used && which means short evaluating circuit so long code doesn't matter.  boolean theSame = false;//variable which return  if (this.getClass() == obj.getClass())//  {  //if whole start date match ->proceed to assess the end date if it also match-> proceed to assess the place and the price  HolidaySimpleWay innerObject = (HolidaySimpleWay) obj;  if ((innerObject.getStartDay()).equals(this.getStartDay()) && (innerObject.getStartMonth()).equals(this.getStartMonth()) && (innerObject.getStartYear()).equals(this.getStartYear()))  if ( (innerObject.getEndDay()).equals(this.getEndDay()) && (innerObject.getEndMonth()).equals(this.getEndMonth()) && (innerObject.getEndYear()).equals(this.getEndYear()))  if ((innerObject.getPlace()).equals(this.getPlace()) && (innerObject.getPrice()).equals(this.getPrice()) )  {  theSame = true;//finally I can admit that this objects are equal.  }  }  return theSame;//returning state -false mean objects are not the same.  }  public String showAllData()  {  return ("Holiday will start on : "+ getStartDay()+"-"+getStartMonth()+"-"+getStartYear()+" and end on : "+ getEndDay()+"-"+ getEndMonth()+"-"+getEndYear()+"\n The holiday will took place at "+ getPlace()+" whole cost will be "+getPrice() );  } }  **ochastreingSimpleWayTwoObjectsGUI class:**  import javax.swing.JOptionPane;  import java.awt.\*;  import java.awt.event.\*;  import javax.swing.\*;  /\*\*  \* Write a description of class ochastreingSimpleWay here.  \* HolidaySimpleWay  \* @author (your name)  \* @version (a version number or a date)  \*/  public class ochastreingSimpleWayTwoObjectsGUI extends JFrame  {  //declaring labels and fields.  //declaring other parameters(place and price) labels +fields  JTextField place;//declaring text field for place of holiday, rest in the same way  JLabel placeLabel;//declaring Label for place of holiday, rest in the same way  JTextField price;  JLabel priceLabel;  //start dates and labels  JTextField startDay;  JLabel startDayLabel;  JTextField startMonth;  JLabel startMonthLabel;  JTextField startYear;  JLabel startYearLabel;  //ending dates and labels  JTextField endDay;  JLabel endDayLabel;  JTextField endMonth;  JLabel endMonthLabel;  JTextField endYear;  JLabel endYearLabel;  //declaring objects  HolidaySimpleWay object1,object2;  public ochastreingSimpleWayTwoObjectsGUI()  {  super();  // get the container object  Container containerOne = getContentPane();  // set the layout of the container object  containerOne.setLayout(new GridLayout(10,2));  // initializing labels and text fields  place= new JTextField();  placeLabel= new JLabel("Type place of your holiday");  price= new JTextField();  priceLabel= new JLabel("Type price of your holiday");  startDay= new JTextField();  startDayLabel= new JLabel(" Start day of your holiday");  startMonth= new JTextField();  startMonthLabel= new JLabel("Start month of your holiday");  startYear= new JTextField();  startYearLabel= new JLabel("Start year of your holiday");  endDay= new JTextField();  endDayLabel= new JLabel("Finishing day of your holiday");  endMonth= new JTextField();  endMonthLabel= new JLabel("Finishing month of your holiday");  endYear= new JTextField();  endYearLabel= new JLabel("Finishing year of your holiday");  JButton creation1=new JButton("Create 1st Object.");  JButton creation2=new JButton("Create 2nd Object.");  JButton comparision=new JButton("Compare Objects.");  //Adding to GUI Label + text field for each attribute of object  containerOne.add(startDayLabel);  containerOne.add(startDay);  containerOne.add(startMonthLabel);  containerOne.add(startMonth);  containerOne.add(startYearLabel);  containerOne.add(startYear);  containerOne.add(endDayLabel);  containerOne.add(endDay);  containerOne.add(endMonthLabel);  containerOne.add(endMonth);  containerOne.add(endYearLabel);  containerOne.add(endYear);  containerOne.add(placeLabel);  containerOne.add(place);  containerOne.add(priceLabel);  containerOne.add(price);    //Adding to GUI buttons  containerOne.add(creation1);  containerOne.add(creation2);  containerOne.add(comparision);  //actions for buttons    //action for Button which compare two objects  comparision.addActionListener(new ActionListener()  {  public void actionPerformed(ActionEvent e)  {  if(object1.equals(object2)){  System.out.println("Objects are equal");  }  else  System.out.println("Objects are different");  }  });    //actions for button which create first object  creation1.addActionListener(new ActionListener()  {  public void actionPerformed(ActionEvent e)  {  //Declaring objects.  object1= new HolidaySimpleWay();  //HolidaySimpleWay object2= new HolidaySimpleWay();  //object1 parameters  object1.setPlace(place.getText());  object1.setPrice(price.getText());  object1.setStartYear(startYear.getText());  object1.setStartMonth(startMonth.getText());  object1.setStartDay(startDay.getText());  object1.setEndYear(endYear.getText());  object1.setEndMonth(endMonth.getText());  object1.setEndDay(endDay.getText());  System.out.println(object1.showAllData());  }  });  //actions for button which create second object  creation2.addActionListener(new ActionListener()  {  public void actionPerformed(ActionEvent e)  {    //Declaring objects.  object2= new HolidaySimpleWay();  //setting object2 parameters  object2.setPlace(place.getText());  object2.setPrice(price.getText());  object2.setStartYear(startYear.getText());  object2.setStartMonth(startMonth.getText());  object2.setStartDay(startDay.getText());  object2.setEndYear(endYear.getText());  object2.setEndMonth(endMonth.getText());  object2.setEndDay(endDay.getText());  System.out.println(object2.showAllData());//show data of object  }  });  setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  // Causes this Window to be sized to fit the preferred size  // and layouts of its subcomponents  pack();  setVisible(true);// shows the window  }  public static void main(String [] args)  {  System.out.println("If you want to have different values in second Object ,change actual fields.");  ochastreingSimpleWayTwoObjectsGUI ochastreing1 = new ochastreingSimpleWayTwoObjectsGUI();//create a GUI  }  }  **Premier class:**  /\*\*  \* @Alexander Drabek  \* @version 1.0  \*/  public class Premier extends HolidaySimpleWay  //subclass of HolidaySimpleWay  {  // instance variables  private String hotelName;  private String hotelGrade;  private String resortName;  public Premier()  {  super();  hotelName="";//we don't know the value yet so ""  hotelGrade="";//we don't know the value yet so ""  resortName="";//we don't know the value yet so ""  }  public void setHotelName( String hotelName)  {  this.hotelName= hotelName;  //I am aware that this.hotelName it is not the same //variable as hotelName  //for me it is more clear in this way .  }    public void setHotelGrade( String hotelGrade)  {  this.hotelGrade= hotelGrade;  }  public void setResortName( String resortName)  {  this.resortName= resortName;  }  public String getHotelName(){  return hotelName;  }  public String getHotelGrade(){  return hotelGrade;  }  public String getResortName(){  return resortName;  }  public boolean equals(HolidaySimpleWay obj)// parameter will be an object of HolidaySimpleWay  {  //I will compare all things to assure that object is the same  //I used && which means short evaluating circuit so long code doesn't matter.  boolean theSame = false;//variable which return  if (this.getClass() == obj.getClass())  //compare if the objects are the same class  {  //if whole start date match ->proceed to assess the end date if it also match-> proceed to assess the place and the price  Premier innerObject = (Premier) obj;  if ((innerObject.getStartDay()).equals(this.getStartDay()) && (innerObject.getStartMonth()).equals(this.getStartMonth()) && (innerObject.getStartYear()).equals(this.getStartYear()))  if ( (innerObject.getEndDay()).equals(this.getEndDay()) && (innerObject.getEndMonth()).equals(this.getEndMonth()) && (innerObject.getEndYear()).equals(this.getEndYear()))  if ((innerObject.getPlace()).equals(this.getPlace()) && (innerObject.getPrice()).equals(this.getPrice()) )  if((innerObject.getHotelName()).equals(this.getHotelName()) && (innerObject.getHotelGrade()).equals(this.getHotelGrade()) && (innerObject.getResortName()).equals(this.getResortName()) )  {  theSame = true;//finally I can admit that this objects are equal.  }  }  return theSame;//returning state -false mean objects are not the same.  }  //I had showAllDataPremier and showAllDataHoliday() but now I changed to the same name and let overriding working.  public String showAllData(){  return (super.showAllData()+".\n The hotel name is "+getHotelName()+" and it grade is "+getHotelGrade()+" .The hotel is placed in "+ getResortName());  } } | **Test result**:  Picture 1.1  C:\Users\Study\Desktop\example5-obj equal.png  Picture 1.2  C:\Users\Study\Desktop\example5-obj diffrent.png |