```
//Student's Full Name- Tasfique Enam
//Student's ID- J16020825/5886429
//Modification Date 18/05/2019
//Purpose of this file- RentalItem abstract Class
package assignment2;

public abstract class RentalItem { //abstract class
    int id;
    double ratePerDay;
    int numOfDays;
    double rentalCalculation;

    public RentalItem () { //default constructor
        id= 0;
        ratePerDay = 0;
        numOfDays = 0;
    }
    public RentalItem (int id, double ratePerDay, int numOfDays) { //non default
constructor
        this.id = id;
        this.ratePerDay = ratePerDay;
        this.numOfDays = numOfDays;
    }

    public int getId() { //getter for ID
        return id;
    }
```

```
    public void setId(int id) { //setter for ID
        this.id = id;
    }

    public double getRatePerDay() { //getter for Rate per day
        return ratePerDay;
    }

    public void setRatePerDay(double ratePerDay) { //setter for rate per day
        this.ratePerDay = ratePerDay;
    }

    public int getNumOfDays() { //getter for number of days
        return numOfDays;
    }

    public void setNumOfDays(int numOfDays) { //setter for number of days
        this.numOfDays = numOfDays;
    }


    public abstract double calculateRental(); //abstract method for calculating the
total




}
```

```
//Student's Full Name- Tasfique Enam
//Student's ID- J16020825/5886429
//Modification Date 18/05/2019
//Purpose of this file- Rental Class
package assignment2;
import java.util.ArrayList;
import java.util.Scanner;
import java.util.Iterator;

public class Rental {
    private int id; //declaraing attributes.
    private Customer customerObj; //customer obj
    private ArrayList<RentalItem> rentalItems; //array list of rentalItem.



    public Rental () { //default constructor.
        id = 0;
        customerObj = null;
        rentalItems = new ArrayList (); //creating rental item arraylist
    }


    public void setRentalID (int id) { //setter for rental ID
        this.id = id;
    }


    public void setCustomer(Customer customerObj) { //set method to set the
customer's details
```

```
        this.customerObj = customerObj;
    }

    public void addRentalItem (RentalItem rentalObj) { //adding the rental item
object to the arraylist called rentalItems

        rentalItems.add(rentalObj);
        /*for (Iterator it = rentalItemList.iterator(); it.hasNext();) {
            Object rentalItemList1 = it.next();
        }*/
    }

    public void removeRentalItem() { //method to remove rental Items.
        rentalItems.removeAll(rentalItems);
    }

    public double calculateTotalRental () { //calculation method for Total Rental
        double total =0;
        for (int x=0; x<rentalItems.size(); x++) { //using for loop because
            total += rentalItems.get(x).calculateRental(); //calculation.
        }
        return total;

    }

    @Override
    public String toString () { //toString method to Display
        int x;
```

```
        String str = "";

        str ="\n**********CUSTOMER INFORMATION**********\n"+"The
Name of Customer is "+customerObj.getName()+"\n"+
                "Contact Number "+customerObj.getContactNo()+"\n"+
                "The Rental ID is "+id +"\n"+
                "\n**********RENTAL INFORMATION********** \n"+
              "The Total Rental Cost "+calculateTotalRental()+"\n";

        for(x=0; x<rentalItems.size(); x++) {
          str = str + rentalItems.get(x).toString();
        }

        return str;

    }



    public static void main (String [] args) { //main method
        int option=0; //option for switch statement
        int option2; //option for Rental Item selection
        int rentalIndexNumber =1;
        int indexNumber=0;
        Scanner read = new Scanner(System.in);

        Customer customerObj = null; //customer obj
        Rental rentalObj = null; //rental obj
```

```
        boolean errorMessage = true; //used for error message validation.

        ArrayList<Rental> rentalList = new ArrayList<>(); //arraylist of rental

        System.out.println("     WELCOME!");

        do{
            do{
                try{

System.out.print("**************************************************
**********\n" + //Selection menu for the user to select
                        "1. Add a Customer.\n" +
                        "2. Add a Rental Item\n" +
                        "3. Remove a Rental Item\n"+
                        "4. Display all Rental Items and Total Rental Charges\n"+
                        "5. Display a Single Rental Item and it's Total Charges.\n"+
                        "6. Exit\n"+

"**********************************************************\n" +
                        "Please the enter the number '1' '2' '3' '4' '5' '6' :"
                    );
                    option = Integer.parseInt(read.next());

                    errorMessage = false;

                } catch(Exception e) { //catch for invalid input from the user.
```

```
            System.out.println("\nYou have entered an Invalid Selection\n");
            read.reset();
        }

    }while(errorMessage);

        switch(option) {
            case 1: //Add a new Customer.
            {
                do{
                    try{

System.out.print("\n*****************************\n");
                        System.out.println("Adding a New Customer.");

System.out.print("*****************************\n");

                        boolean f;
                        int id =0;
                        do{
                            f = true;
                            System.out.println("Enter Rental ID "); //taking the
details of the Customer..
                            id = Integer.parseInt(read.next());

                            for(int j=0; j<rentalList.size(); j++) {
                                if (rentalList.get(j).id == id) {
```

```
                    System.out.println("The ID you have Entered
Already exists, Please Try again. ");
                            /* i =0;
                               j =0;*/
                    f = false;
                    break;
                }

            }

        } while(!f);



        System.out.println("Enter the Name of the Customer");
        String name = read.next();
        System.out.println("Enter the phone of the Customer.");
        String contactNo = read.next();

        customerObj = new Customer(name, contactNo); //setting
customer details in customer object.

        rentalObj = new Rental(); // creation of a rental object
        rentalObj.setRentalID(id); //setting id in rental object
        rentalObj.setCustomer(customerObj); //setting customer
in rental obj

        rentalList.add(rentalObj); //adding the rental object into
the array list.
```

```
                    System.out.print("\nNow there are "+rentalList.size()+"
Customer(s) in the System\n "); //show output when the customer object have
been successfully inputted into the arraylist.

                    System.out.println("\n"+name+" is Customer Number
"+rentalList.size()+"\n"); //showing which customer is which number.


                    errorMessage = false;


                } catch (Exception e) { //exception for wrong input
                    System.out.println("\nInvalid Input, Please try again. \n");
                    read.reset();
                }
            } while (errorMessage);
            break;
        }

        case 2: // Add an item
        {


            do{
                try{
                    System.out.println("\nWhich Customer Number are you?
");
                    rentalIndexNumber = Integer.parseInt(read.next());


                    if(rentalList.size()!=0 && rentalIndexNumber <=
rentalList.size() && rentalIndexNumber > 0){ //doing validation if the user
selected option 2 directly without inputting customer details.
```

```
                        //System.out.println("\nRental Number : " +
(indexNumber+1));
                        //indexNumber++;


System.out.print("\n*****************************\n"); //asing user what
they want to rent.
                        System.out.println("Adding a New Rental Item.");

System.out.print("*****************************\n");
                        System.out.println("\nWhat do you want to Rent?");
                        System.out.println("1. Boat");
                        System.out.println("2. Bicycle");
                        System.out.println("3. JetSki");

                        System.out.println("Enter '1' , '2', '3' or Any number to go
back \n");

                        option2 = Integer.parseInt(read.next()); //taking in user's
input

                        switch(option2) {
                           case 1: //Renting a Boat.
                           {
                              do{
                                 try {
                                    boolean f;
```

```
                int id =0;
                do{
                    f = true;
                    System.out.println("Enter the ID of the Boat
"); //taking the details of the boat.
                    id = Integer.parseInt(read.next());

                    for(int j=0; j<rentalList.size(); j++) {
                        for(int i =0;
i<rentalList.get(j).rentalItems.size(); i++) {
                            if
(rentalList.get(j).rentalItems.get(i).getId() == id) {
                                System.out.println("The ID you have
Entered Already exists, Please Try again. ");
                                /* i =0;
                                    j =0;*/
                                f = false;
                                break;
                            }
                    }
                }

                } while(!f);

                System.out.println("Enter the Rate per Day");
                double ratePerDay =
Double.parseDouble(read.next());
                System.out.println("Enter the Number of Days
to Rent");
```

```
                    int numOfDays = Integer.parseInt(read.next());
                    System.out.println("Enter the Capacity");
                    int capacity = Integer.parseInt(read.next());

                    RentalItem boatObj = new Boat(id, ratePerDay, numOfDays, capacity); //inputting the details of the boat into the boat object

                    rentalList.get(rentalIndexNumber-1).addRentalItem(boatObj); //inputting the object into the arraylist.

                    System.out.println("\nThe Boat have been Successfully Added.\n");

                    errorMessage = false;

                } catch(Exception e) { //error message if the user inputs the wrong data.
                    System.out.println("You have Entered an Invalid Input, Please enter the Correct Input\n");
                    System.out.println("You are now being Redirected to the Main Menu, Please enter the Input correctly.");
                    read.reset();
                }

            }while (errorMessage);

            break;
```

```
                    }

                    case 2: //Renting a Bicycle.
                    {
                       do{

                          try{
                             boolean f;
                             int id =0;
                             do {
                                f = true;

                             System.out.println("\nEnter the ID of the
Bicycle ");
                             id = Integer.parseInt(read.next());



                             for(int j=0; j<rentalList.size(); j++) {
                                for(int i =0;
i<rentalList.get(j).rentalItems.size(); i++) {
                                   if
(rentalList.get(j).rentalItems.get(i).getId() == id) {
                                      System.out.println("The ID you have
Entered Already exists, Please Try again. ");
                                   /* i =0;
                                    j =0;*/
                                   f = false;
                                   break;
```

```
                }
            }
        }
        } while(!f);

        System.out.println("Enter the Rate per Day");
        double ratePerDay =
Double.parseDouble(read.next()); //check here
        System.out.println("Enter the Number of Days
to Rent");
        int numOfDays = Integer.parseInt(read.next());
        System.out.print("Enter the Bike Type \n" +
            "1. 'Mountain Bike' \n"+
            "2. 'Kids Bike' \n"+
            "3. 'Other Bike' \n"+
            "Enter '1' , '2', '3' \n"
            );
        String type = read.next();

        if( type.equalsIgnoreCase("1") ||
type.equalsIgnoreCase("2") || type.equalsIgnoreCase("3")){ //doing verification
if the user inputs the wrong number
            RentalItem bicycleObj = new Bicycle(id,
ratePerDay, numOfDays, type);
            rentalList.get(rentalIndexNumber-
1).addRentalItem(bicycleObj);

            System.out.println("\nThe Bicycle have been
Successfully Added.\n");
        } else {
```

```
                    System.out.println("You have Entered a Bike
Type that doesn't Exist \n");
                    System.out.println("You are now being
Redirected to the Main Menu, Please enter the Input correctly.");
                }

                errorMessage = false;

            } catch (Exception e) {
                System.out.println("You have Entered an
Invalid Input, Please enter the Correct Input\n");
                System.out.println("You are now being
Redirected to the Main Menu, Please enter the Input correctly.");
                read.reset();
            }

        }while (errorMessage);

        break;

    }

    case 3: //Renting a JetSki.
    {

        do{

            try{
```

```
                                    boolean f;
                                    int id =0;
                                    do {
                                       f = true;

                                    System.out.println("\nEnter the ID of the JetSki
");
                                    id = Integer.parseInt(read.next());



                                    for(int j=0; j<rentalList.size(); j++) {
                                       for(int i =0;
i<rentalList.get(j).rentalItems.size(); i++) {
                                          if
(rentalList.get(j).rentalItems.get(i).getId() == id) {
                                             System.out.println("The ID you have
Entered Already exists, Please Try again. ");
                                          /* i =0;
                                           j =0;*/
                                          f = false;
                                          break;
                                       }
                                    }
                                    }
                                    } while(!f);

                                    System.out.println("Enter the Rate per Day");
```

```
                    double ratePerDay =
Double.parseDouble(read.next());
                    System.out.println("Enter the Number of Days
to Rent");
                    int numOfDays = Integer.parseInt(read.next());
                    System.out.println("Enter the Horsepower");
                    double horsePower =
Double.parseDouble(read.next());

                    RentalItem jetskiObj = new JetSki(id,
ratePerDay, numOfDays, horsePower); //putting the data into jetski object

                    rentalList.get(rentalIndexNumber-
1).addRentalItem(jetskiObj); //putting the dobject into an arraylist.

                    System.out.println("\nThe JetSki have been
Successfully Added.\n");

                    errorMessage = false;



                } catch (Exception e) {
                    System.out.println("\nInvalid Input \n");
                    read.reset();
                }

            } while (errorMessage);
```

```
                    break;


                }
                default:
                {
                    System.out.println("\nYou have selected the wrong number \n"); // if the user have inputted the wrong number, in the selection menu.
                    break;
                }
            }


            }else{
                System.out.println("\nThe Customer Number you have Selected doesn't exist in the System \n");
                }
            errorMessage = false;



        } catch (Exception e) {
            System.out.println("\nInvalid Input, Please try again. \n");
            read.reset();
        }

    } while (errorMessage);
    break;


}
```

```
case 3: // Remove a Rental Item.
{
   do{
      try {
         int rentalcount=0;


System.out.print("\n**********************************************************\n");
                  System.out.println("Display of all Rental Items and Total Rental Charges"); //display of all the Rental items and it's total rental charges.

System.out.print("**********************************************************\n");
                  System.out.println("\n");
                  Iterator <Rental> itr = rentalList.iterator(); //using itr to display all the elements


                  while(itr.hasNext()) {
                     Rental element = itr.next();
                     System.out.println("\n**********Rental Item Number "+(rentalcount+1)+"********** \n"); //keeping the count of the Rental Item
                     rentalcount++;

System.out.println("\n**********************************************************\n"+element);



                  }
```

```
System.out.print("\n*****************************\n"); //removal of a specific rental item.
                    System.out.println("Remove a Rental Item.");

System.out.print("*****************************\n");


                    System.out.println("\nEnter The Rental Item Number you want to Remove."); //asking the user which rental item they want to remove.
                    int rentalIndex =  Integer.parseInt(read.next());


                    if((rentalIndex-1) >= 0 && (rentalIndex-1) < rentalList.size()) { //doing validation if a specific rentail item exists or not, if it doesn't it will show an error message.
                        rentalList.get(rentalIndex-1).removeRentalItem();
                        System.out.println("\nThe selected Rental Item have been Successfully Removed.\n");
                    } else {
                        System.out.println("\nThe selected RentalItem doesn't exist in the System.\n");
                    }
                    errorMessage = false;



                } catch (Exception e) {
                    System.out.println("\nInvalid Input, Please try again. \n"); //if the user inputs the wrong input
                    read.reset();
                }
```

```
            } while(errorMessage);

            break;
        }

        case 4: //display of Rental Items
        {

System.out.print("\n**********************************************************\n");
            System.out.println("Display of all Rental Items and Total Rental Charges"); //display of all the Rental items and it's total rental charges.

System.out.print("**********************************************************\n");
            System.out.println("\n");
            Iterator <Rental> itr = rentalList.iterator(); //using itr to display all the elements

            while(itr.hasNext()) {
                Rental element = itr.next();

System.out.println("**********************************************************\n"+element);

            }

            break;
```

```
}

case 5:
{

   do{
      try{

System.out.print("\n*****************************\n");
                    System.out.println("Display of a Single Rental Items and it's Total Rental Charges"); //display of a specific rental item and it's total charges.

System.out.print("*****************************\n");
                    System.out.println("\nSelect which Item Number you want to Display\n"); //asking the user which rental number they want to display
                    System.out.println("Select a Number.");
                    int rentalItemIndex = Integer.parseInt(read.next());

                    if( (rentalItemIndex-1) < rentalList.size() && (rentalItemIndex-1) >= 0 ) { //checking if the user selected rental item exists or not?
                       //rentalList.get(rentalItemIndex-1).toString();
                       //for(int i =0; i<rentalList.get(i).rentalItems.size(); i++){
                          System.out.print("Single Rental Item\n"+rentalList.get(rentalItemIndex-1));
                       //}

                    } else {
```

```
                    System.out.println("\nWhat you have selected doesn't exist."); //error message when the selected the
                    }
                    errorMessage = false;


                } catch(Exception e) {
                    System.out.println("\nInvalid Input, Please try again. \n"); //shows invalid input when the user inputs wrong input.
                    read.reset();
                }

            } while (errorMessage);

            break;
        }

        case 6: { //case 6 is used for exiting the program.
            System.exit(0);
            return;
        }
        default:
        {
            System.out.println("\nYou have selected a Number that is not available in the Menu Selection \n"); // if the user have inputted the wrong number, in the selection menu.
            break;
        }
```

```
                }

        }while (option!=6);
    }
}


//Student's Full Name- Tasfique Enam
//Student's ID- J16020825/5886429
//Modification Date 18/05/2019
//Purpose of this file- Jetski Class
package assignment2;


public class JetSki extends RentalItem {//subclass
    private double horsePower; //declaring of attributes.

    public JetSki () { //default constructor.
        super();
        horsePower = 0.0;
    }

    public JetSki (int id, double ratePerDay, int numOfDays, double horsePower)
{ //non default constructor.
        this.id = id;
        this.ratePerDay = ratePerDay;
        this.numOfDays = numOfDays;
        this.horsePower = horsePower;
```

```
}

public double getHorsePower() { //getter
    return horsePower;
}

public void setHorsePower(double horsePower) { //setter
    this.horsePower = horsePower;
}

@Override
public double calculateRental () { //calculating the rental
    if(horsePower<=250) { //if the horpower is less than 250
        super.rentalCalculation = super.ratePerDay;
    } else {
        super.rentalCalculation = super.ratePerDay * 1.5;
    }

    return (super.rentalCalculation * super.numOfDays);
}

@Override
public String toString () { //toString method to display
    String str;
      str = "\nJetSki ID "+super.id+"\n"+
            "The Rate Per Day is "+super.ratePerDay+"\n"+
```

```
                "The Number of Days Renting "+super.numOfDays+"\n"+
                "The HorsePower is "+getHorsePower()+"\n";

        return str;
    }


}

//Student's Full Name- Tasfique Enam
//Student's ID- J16020825/5886429
//Modification Date 18/05/2019
//Purpose of this file- Customer Class
package assignment2;


public class Customer { //declariing of attributes.
    private String name;
    private String contactNo;

    public Customer(String name, String contactNo) { //non default constructor.
        this.name = name;
        this.contactNo = contactNo;
    }

    public String getName() { //getter
        return name;
```

```
    }

    public void setName(String name) { //setter
        this.name = name;
    }

    public String getContactNo() { //getter
        return contactNo;
    }

    public void setContactNo(String contactNo) { //setter
        this.contactNo = contactNo;
    }

    @Override
    public String toString () { //to string to display
        String str;
        System.out.println("");
          str = "\nThe Name of Customer is \n"+getName()+"\n"+
                "Contact Number "+getContactNo()+"\n";
        return str;
    }


}

//Student's Full Name- Tasfique Enam
```

```
//Student's ID- J16020825/5886429
//Modification Date 18/05/2019
//Purpose of this file- Boat Class
package assignment2;


public class Boat extends RentalItem {//subclass
    private int capacity; //declaring of attributes.

    public Boat() { //default constructor.
        super();
        capacity = 0;
    }

    public Boat(int id, double ratePerDay, int numOfDays, int capacity) { //non
default constructor
        this.id = id;
        this.ratePerDay = ratePerDay;
        this.numOfDays = numOfDays;
        this.capacity = capacity;
    }

    public int getCapacity() { //getter
        return capacity;
    }

    public void setCapacity(int capacity) { //setter
        this.capacity = capacity;
```

```
    }


    @Override
    public double calculateRental() { // to do calculation of the rental for boat.
        if(capacity>10) { //if the capacity is more than 10
            super.rentalCalculation = (super.ratePerDay + 50); // rate per day will
have addition of 50
        } else {
            super.rentalCalculation = super.ratePerDay; //else it will not change the
rental will be same, if the capacity is less than 10.
        }
        return (super.rentalCalculation*super.numOfDays);
    }


    @Override
    public String toString () {//to string to display
        String str;
          str = "\nBoat ID "+super.id+"\n"+
                "The Rate Per Day is "+super.ratePerDay+"\n"+
                "The Number of Days Renting "+super.numOfDays+"\n"+
                "The Capacity is "+getCapacity()+"\n";


        return str;
    }


}
```

```
//Student's Full Name- Tasfique Enam
//Student's ID- J16020825/5886429
//Modification Date 18/05/2019
//Purpose of this file- Bicycle Class
package assignment2;

public class Bicycle extends RentalItem {//subclass
    private String type; //declaring of attributes.

    public Bicycle() { //default constructor
        super();
        type = null;
    }

    public Bicycle (int id, double ratePerDay, int numOfDays, String type) {
//non default constructor.
        this.id = id;
        this.ratePerDay = ratePerDay;
        this.numOfDays = numOfDays;
        this.type = type;
    }

    public String getType() { //getter
        return type;
    }

    public void setType(String type) { //setter
        this.type = type;
```

```
    }

    @Override //the method is overridden
    public double calculateRental () {
        if (type.equalsIgnoreCase("1")) { //if the user selects 1, which Mountain Bike
            super.rentalCalculation = super.ratePerDay + 10;
        }

        else if (type.equalsIgnoreCase("2")) { //if the user selects 2, which is Kids bike
            super.rentalCalculation = super.ratePerDay / 2;
        }

        else if(type.equalsIgnoreCase("3")) { // if the user selects 3, which is other bike.
            super.rentalCalculation = super.ratePerDay;
        }

        return (super.rentalCalculation*super.numOfDays);

    }

    @Override //the method is overridden
    public String toString () { //to string to display
        String str;
          str = "\nBicycle ID "+super.id+"\n"+
                "The Rate Per Day is "+super.ratePerDay+"\n"+
```

```
                "The Number of Days Renting "+super.numOfDays+"\n"+
                "The Type is "+getType()+"\n"+
                "\nType 1 = 'Mountain Bike' \n"+
                "Type 2 = 'Kid's Bike' \n"+
                "Type 3 = 'Other Bike' \n";

        return str;
    }
}
```