Lab Report – User Defined Object Programming Assignment

John Marshall
CSCI112 Spring 2023

Assignment Analysis and Design

In your own words, describe the problem including input and output. Briefly describe how you developed your code. Briefly describe what your code does and how it works – including anything different or unique or special that you did in your software. If the software is long or complicated, describe how it is organized.
Include a copy of any pseudocode or other design documents you used. If you worked with anyone else, asked anyone for help, or looked anything up, then mention it here. Include proper references to source material.

The purpose of this assignment was to create an Array management System for the Pilot Objects of last week's assignment. This was made Slightly more complicated by having to incorporate an API. I developed most of my code by following along with the tutorial video but I did do a fair bit of editing. The Manager had to be able to save to a text file, load from a text file, edit a specific object, list the entire array of objects, add an object to the array, and delete an object off the array. The “editing of an object's properties” feature is the feature I am most proud of. It is basically a minute version of the entire assignment in one Method.

See UML document

Assignment Code

Include the code for your assignment unless otherwise directed by the assignment or by your instructor, which will be a copy of your Python project submitted with the report. You can put the report and the Python project all in one submission. In the report, either tell the reader that it is attached file or include the code.

```
/*John Marshall
user Defined Objects
Feb 27 2023
*/
```

```
import java.util.Scanner;

public class Main {

    public static PilotManger pilotManagerList = new PilotManger();
    public static void main(String[] args) {

        pilotManagerList.loadPilots("newPilotsList");
        menu();

    }

    public static void menu(){
        int selection = 0;
        Scanner ms = new Scanner(System.in);

        while(selection != 7){
            System.out.println("\t\tPilot Array API");
            System.out.println("1 ===>> To List Pilots");
            System.out.println("2 ===>> To Add a Pilot");
            System.out.println("3 ===>> To Edit a Pilot");
            System.out.println("4 ===>> To Delete a Pilot");
            System.out.println("5 ===>> To Save Pilots");
            System.out.println("6 ===>> To Load Pilots");
            System.out.println("7 ===>> To Exit System");

            selection = ms.nextInt();

            //list pilots
            if(selection == 1){
                pilotManagerList.print();

            //Add pilot
            }else if(selection == 2){
                Pilot newPilot;
                newPilot = pilotManagerList.getAPilot();
                pilotManagerList.append(newPilot);

            //Edit pilot
            }else if(selection == 3){

                //get index then run EditPilot method
                Pilot pilotTobeEdited = getPilotIDObject();
                int i = pilotManagerList.SeqSearch(pilotTobeEdited); //index of search
                pilotManagerList.editPilot(i);

            //Delete pilot
            }else if(selection == 4){
```

```
            Pilot pilotTobeEdited = getPilotIDObject();
            int i = pilotManagerList.SeqSearch(pilotTobeEdited); //index of search
            pilotManagerList.removeAt(i);

        //Save list
        }else if(selection == 5){
                pilotManagerList.savePilotList();

        //Load list
        }else if(selection == 6){
            String informationCorrect;
            String newListLoad;

            //prompt to load
            System.out.println("What is the file Name you would like to load From? *100 Pilot  Limit");
            newListLoad = ms.next();

            //Echo to user
            System.out.println("Desired File Name: " + newListLoad +"txt");
            System.out.println("Enter Y to confirm");
            informationCorrect = ms.next();
            informationCorrect = informationCorrect.toUpperCase();

            //load list
            if(informationCorrect.equals("Y")){
                pilotManagerList = new PilotManger();
                pilotManagerList.loadPilots(newListLoad);

            } else {
                System.out.println("CONFIRMATION NOT MET, Please try again.");
            }


        }else if(selection == 7){
        System.out.println("Thank you, GoodBye");
        }else{
            System.out.println("Invalid Input");
        }

    }//end while
    ms.close();
}//end Menu

public static Pilot getPilotIDObject(){
    Pilot temp = new Pilot();
    int pilotID;
    Scanner kb = new Scanner(System.in);
    String informationCorrect;
```

```
        //prompt for ID
        System.out.println("Pilot ID to search For \n\n");
        System.out.println("what is the Pilot ID Number of desired Pilot?");
        pilotID = kb.nextInt();


        //Echo ID
        System.out.println("Desired ID Number: " + pilotID);
        System.out.println("Enter Y to confirm");
        informationCorrect = kb.next();
        informationCorrect = informationCorrect.toUpperCase();

        if(informationCorrect.equals("Y")){
            temp.setIdNumber(pilotID);
        } else{
            System.out.println("CONFIRMATION NOT MET, Please try again.");
        }

        return temp;
    }// end getPilotIDObject

}
```

```
import java.io.File;
import java.io.PrintWriter;
import java.util.Scanner;

public class PilotManger {

    protected int length; // stores the current length of elements
    protected int maxSize; // Max Size of length
    protected Pilot [] list; // array holds list of object

    //Begin Constructors-----------------------------------------------------------

    public PilotManger(){
        length = 0;
        maxSize = 10;
        list =  new Pilot[maxSize];
    }//end default constructor

    public PilotManger(int size){
```

```
    length = 0; // initiates the current length at zero

    if(size <=0 ){
        System.err.println("The array must be a positive integer. Creating an Array with 100 Elements");
        maxSize = 100;
    } else {
        maxSize = size;
    }

    list =  new Pilot[maxSize];

}//end constructor

//End Constructors-----------------------------------------------

//Begin Methods--------------------------------------------------

public void append(Pilot newPilot){

    //checks if there's space
    if (length == maxSize){
        System.err.println("Can not Append, List is Full");
    }
    else{

        //assigns newPilot to Array at Index of length and then increments length
        list[length] = newPilot;
        length ++;
    }
}//End Append()


public void print(){
    //loop that print elements in array
    int count = 0;
        while (count<length){
            System.out.println(list[count].toString());
            count++;
        }

}//end Print()


public void loadPilots(String fileName){
    //Record Structure

    // 1st item int idNumber
    // 2nd Item String name
    // 3rd item String hireDate
    // 4th item int licenseNumber
    // 5th item Double rating
    // 6th item Double flightHours
    //idNumber,name,hireDate,licenseNumber,rating,flightHours



    try{
        java.io.File file = new java.io.File(fileName + ".txt");
        Scanner tempData = new Scanner(file);

        while(tempData.hasNext()){
            Pilot temp = new Pilot();

            String record = tempData.nextLine();
            String[] fields = record.split(",");
            temp.setIdNumber(Integer.parseInt(fields[0]));
            temp.setName(fields[1]);
            temp.setHireDate(fields[2]);
            temp.setLicenseNumber(Integer.parseInt(fields[3]));
            temp.setRating(Double.parseDouble(fields[4]));
            temp.setFlightHours(Double.parseDouble(fields[5]));

            append(temp);
        }//end While

       //print(); Test Print
    }catch(Exception e){
        System.out.println(e.getMessage());
        e.printStackTrace();
    }

}//end loadPilots

public Pilot getAPilot(){
    Scanner kb = new Scanner(System.in);
    int idNumber;            //Pilot ID
    String fullName;            //Pilot Name
    String hireDate;        //pilot Hire Date
    int licenseNumber;      // pilot LicenseNumber
    double rating;          //pilot Rating
    double flightHours;     //pilot Flight Hours
    String informationCorrect; //user confirm

    Pilot newPilot = new Pilot(); //Return Obj


    //collect user Information
    System.out.println("CREATING NEW PILOT");
    System.out.println("Pilots ID Number:");
    idNumber = kb.nextInt();

    System.out.println("Pilots Full Name:");
    fullName = kb.next();

    System.out.println("Pilots Hire Date:");
    hireDate = kb.next();

    System.out.println("Pilots License Number:");
    licenseNumber = kb.nextInt();

    System.out.println("Pilots Rating: FORMAT 0.0");
    rating = kb.nextDouble();

    System.out.println("Pilots Flight Hours: FORMAT 0.0");
    flightHours = kb.nextDouble();


    //confirm Info
    System.out.println("Please Confirm the Following Information\n\n");
    System.out.println("Pilots ID Number: " + idNumber);
    System.out.println("Pilots Full Name: " + fullName);
    System.out.println("Pilots Hire Date: " + hireDate);
    System.out.println("Pilots License Number: " + licenseNumber);
    System.out.println("Pilots Rating: " + rating );
    System.out.println("Pilots Flight Hours: " + flightHours);

    System.out.println("Enter Y to confirm");
    informationCorrect = kb.next();
    informationCorrect = informationCorrect.toUpperCase();


    //set Values
    if(informationCorrect.equals("Y")) {


        newPilot.setIdNumber(idNumber);
        newPilot.setName(fullName);
        newPilot.setHireDate(hireDate);
        newPilot.setLicenseNumber(licenseNumber);
        newPilot.setRating(rating);
        newPilot.setFlightHours(flightHours);

        System.out.println("PILOT CREATED");

    }else {
        System.out.println("CONFIRMATION NOT MET DATA DELETED");
```

```
    }
    return newPilot;
}//end GetAPilot

public void savePilotList(){

    // 1st item int idNumber
    // 2nd Item String name
    // 3rd item String hireDate
    // 4th item int licenseNumber
    // 5th item Double rating
    // 6th item Double flightHours
    //idNumber,name,hireDate,licenseNumber,rating,flightHours

    try {
        //create File
        java.io.File file = new File("pilotSaveList.txt");
        java.io.PrintWriter pw = new PrintWriter(file);

        int count = 0;

        while (count < length){

            String printLine = "";

            Pilot temp =  list[count];

            //concatenate
            printLine =  printLine + temp.getIdNumber() + "," ;
            printLine =  printLine + temp.getName() + "," ;
            printLine =  printLine + temp.getHireDate() + "," ;
            printLine =  printLine + temp.getLicenseNumber() + "," ;
            printLine =  printLine + temp.getRating() + "," ;
            printLine =  printLine + temp.getFlightHours() + "," ;

            //print to File
            pw.println(printLine);
            System.out.println(printLine);
            count++;
        }
        pw.close();

    }catch (Exception e){
        System.out.println(e.getMessage());
        e.printStackTrace();
    }//end catch
}//End SavePilotList
```

```
public int SeqSearch(Pilot searchItem){
    int loc;
    boolean found = false;
    for(loc = 0; loc < length; loc++)
        if(list[loc].equals(searchItem)){
            found = true;
            break;
        }
    if (found) {
        return loc;
    }else{
     return -1;
    }
}//end SeqSearch

public void editPilot(int index){

    // 1st item int idNumber
    // 2nd Item String name
    // 3rd item String hireDate
    // 4th item int licenseNumber
    // 5th item Double rating
    // 6th item Double flightHours

    Scanner kb = new Scanner(System.in);
    int sel=0;

    while(sel != 7){

        System.out.println(list[index].toString());
        System.out.println("");
        System.out.println("Enter the number of the field you would like to edit");
        System.out.println("1 ===>> ID number");
        System.out.println("2 ===>> Name");
        System.out.println("3 ===>> Hire Date");
        System.out.println("4 ===>> License Number");
        System.out.println("5 ===>> Rating");
        System.out.println("6 ===>> Flight Hours");
        System.out.println("7 ===>> Exit selection");

        sel = kb.nextInt();


        //IDNumber Edit
        if(sel == 1){

            String informationCorrect;
            int newIDNumber;
            System.out.println("New ID Number?");
            newIDNumber = kb.nextInt();

            //Echo to user
            System.out.println("Desired ID Number: " + newIDNumber);
            System.out.println("Enter Y to confirm");
            informationCorrect = kb.next();
            informationCorrect = informationCorrect.toUpperCase();

            if(informationCorrect.equals("Y")){
                list[index].setIdNumber(newIDNumber);
            } else{
                System.out.println("CONFIRMATION NOT MET, Please try again.");
            }

        //Name Edit
        }if(sel == 2){
            String informationCorrect;
            String newName;
            System.out.println("New Name?");
            newName = kb.next();

            //Echo to user
            System.out.println("Desired Name: " + newName);
            System.out.println("Enter Y to confirm");
            informationCorrect = kb.next();
            informationCorrect = informationCorrect.toUpperCase();

            if(informationCorrect.equals("Y")){
                list[index].setName(newName);
            } else{
                System.out.println("CONFIRMATION NOT MET, Please try again.");
            }

        //Hire Date
        }if(sel == 3){

            String informationCorrect;
            String newDate;
            System.out.println("New Hire Date?");
            newDate = kb.next();

            //Echo to user
            System.out.println("Desired Hire Date: " + newDate);
            System.out.println("Enter Y to confirm");
            informationCorrect = kb.next();
            informationCorrect = informationCorrect.toUpperCase();

            if(informationCorrect.equals("Y")){
                list[index].setHireDate(newDate);
            } else{
                System.out.println("CONFIRMATION NOT MET, Please try again.");
            }

        //License Number
        }if(sel == 4){

            String informationCorrect;
            int newLicenseNum;
            System.out.println("New ID License Number?");
            newLicenseNum = kb.nextInt();

            //Echo to user
            System.out.println("Desired License Number: " + newLicenseNum);
            System.out.println("Enter Y to confirm");
            informationCorrect = kb.next();
            informationCorrect = informationCorrect.toUpperCase();

            if(informationCorrect.equals("Y")){
                list[index].setLicenseNumber(newLicenseNum);
            } else{
                System.out.println("CONFIRMATION NOT MET, Please try again.");
            }

        //Rating
        }if(sel == 5){

            String informationCorrect;
            double newRating;
            System.out.println("New Rating?");
            newRating = kb.nextDouble();

            //Echo to user
            System.out.println("Desired Rating: " + newRating);
            System.out.println("Enter Y to confirm");
            informationCorrect = kb.next();
            informationCorrect = informationCorrect.toUpperCase();

            if(informationCorrect.equals("Y")){
                list[index].setRating(newRating);
            } else {
                System.out.println("CONFIRMATION NOT MET, Please try again.");
            }

        //Flight Hours
        }if(sel == 6){
            //USER INPUT FOR NEW LOAD FILE
            String informationCorrect;
            double newFlightHours;
```

```
            System.out.println("New Flight Hours?");
            newFlightHours = kb.nextDouble();

            //Echo to user
            System.out.println("Desired Flight Hours: " + newFlightHours);
            System.out.println("Enter Y to confirm");
            informationCorrect = kb.next();
            informationCorrect = informationCorrect.toUpperCase();

            if(informationCorrect.equals("Y")){
               list[index].setFlightHours(newFlightHours);
            } else {
               System.out.println("CONFIRMATION NOT MET, Please try again.");
            }

         }if(sel == 7){

         }else{
            System.out.println("Invalid Input");
         }
      }//end While menu

   } //End Edit Pilot

   public void removeAt(int index){
      if (index < 0 || index > length){
         System.err.println("Item index to be removed is out of range");
      } else{
         for(int i = index; i < length -1; i ++){
            list[i]=list[i+1];
            list[length - 1] =  null;
            length --;
         }
      }
   }//End removeAt

}
```

```
public class Employee {
   public int idNumber;
   public String name;
   public String hireDate;
   public String employeeType;
}


//Constructors *******************

//default Constructor
public void Employee(){

}



public void Employee(int idNum, String employeeName, String startDate, String type){
    this.idNumber = idNum;
    this.name = employeeName;
    this.hireDate = startDate;
    this.employeeType= type;

}



//Gets and Sets *******************


//get and set IDNumber
public int getIdNumber() {
    return idNumber;
}

public void setIdNumber(int idNumber) {
    this.idNumber = idNumber;
}
//End IDNumbers



//get and set Name
public String getName() {
    return name;
}

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


//get and Set EmployeeType
public String getEmployeeType() {
    return employeeType;
}
```

```
   public void setEmployeeType(String employeeType) {
      this.employeeType = employeeType;
   }
   //End EmployeeType


   //get and Set HireDate
   public String getHireDate() {
      return hireDate;
   }

   public void setHireDate(String hireDate) {
      this.hireDate = hireDate;
   }
   //End hireDate

   //end Gets and SETS******************


   //toString Method
   public String toString(){
      String item = "";
      item = item +"Employee ID Number: " + this.idNumber +"\n";
      item = item + "Employee Name: " + this.name + "\n";
      item = item + "Employee Type: " + this.employeeType + "\n";
      item = item + "Employee Hire Date: " + this.hireDate + "\n";


      return item;
   }

   //end toString



}//End Class
```

```
public class Pilot extends Employee{

   private int licenseNumber;
   private double rating;
   private double flightHours;

   public Pilot(){
       this.employeeType = "Pilot";
   }

   public Pilot(int idNum, String name, String hireDate, int licenseNum){
       this.idNumber = idNum;
       this.name = name;
       this.employeeType = "Pilot";
       this.hireDate = hireDate;
       this.licenseNumber = licenseNum;


   }

//Gets and Sets*************************************
   public int getLicenseNumber() {
       return licenseNumber;
   }

   public void setLicenseNumber(int licenseNumber) {
       this.licenseNumber = licenseNumber;
   }

   //End constructors

   //Gets and Sets *************************************
   public double getRating() {
       return rating;
   }

   public void setRating(double rating) {
       this.rating = rating;
   }

   public double getFlightHours() {
       return flightHours;
   }

   public void setFlightHours(double flightHours) {
       this.flightHours = flightHours;
   }

   //end Gets and Sets

   //Object Methods *************************************

   public void eject(){
       System.out.println(this.name + " has ejected from their AirCraft!");
   }
   public int compare(Employee e){
       if(idNumber > e.idNumber){
           return 1;
       } else if (idNumber < e.idNumber) {
           return -1;
       }else {
           return 0;
       }
   }

   @Override
   public String toString() {
       String item = "";
       item = item + super.toString();
       item = item + "Pilot License Number:" + licenseNumber + "\n";
       item = item + "Pilot Ratings:" + rating + "\n";
       item = item + "Flight Hours:" + flightHours + "\n";

       return item;
   }



   //object Deep Copy
   public void copy(Pilot source){
       this.idNumber = source.idNumber;
       this.name = source.name;
       this.hireDate = source.hireDate;
       this.employeeType = source.employeeType;
       this.licenseNumber = source.licenseNumber;
       this.rating = source.rating;
       this.flightHours = source.flightHours;
   }

   //equals Method
   public boolean equals(Employee e){
       return (this.idNumber == e.getIdNumber());
   }


}
```

Assignment Testing

Describe how you tested this program to verify that it runs correctly. Assignment Evaluation
Briefly describe what you learned from this project. What things did you struggle with? What was easy? Give your opinions of the process, including what you liked about the project and any suggestions you have for improving the project.

All testing should be done in the main method. All test were performed on the Pilot Manager

+ PilotManager – Test: Initiate an instance of the constructor. if it creates theinstance, it passes
+ PilotManager(int)- Test: Initiate an instance of the constructor with a specific size. Print the size of the array to see if it is correct. If so it passes

+ Append(pilot) Manually create three instances of the PIlot and use the append method to populate them.

+ loadPilots(String) Create a text file using the correct Comma format and then load it into the array manager. Use the print method to display the data, if data is correct, it passes

+ getAPilot()Pilot Use it to create an instance that can then be appended to the array. Use the print array method to print the array. If the instance of the object is there it passes.

+ SavePilotList( ) Use the method to save an array of pilots. If the file saves, it passes

+ SeqSearch(Pilot) Int - pass it an object with a know index, if it returns the right one it passes

+ editPilot(int) uses the method to edit an object in the array, print the object. If it's edited, it passes

+ removePilot(int): use it on a populated array and then print the array. If the desired object is missing. It passes

Conclusion:

On concluding this project I thought it was very labor intensive..One major problem that I ran into was when loading the array from the text file, the array kept printing copies of just the last object that was supposed to be on the array. I quickly figured out that I was simply editing one instance of the Pilot Object and creating an array of pointers to that one instance. The solution was to simply instantiate my object in the while loop i was using to populate the list.