**CSC 285 Session 9**

**Putting It All Together (Part 3)**

**// RecordXe.java**

**import java.io.\*;**

**public class Recordxe**

**{ //object creating**

**private int recordxe; //4**

**private String name; //15**

**private String majorname; //15**

**private double majorscore; //8**

**private double majorscore2;**

**private double optionalclass1;**

**private double optionalclass2;**

**private int earned;**

**public void read( RandomAccessFile file ) throws IOException**

**{**

**recordxe = file.readInt();**

**char n[] = new char[ 15 ];**

**for ( int i = 0; i < n.length; i++ )**

**n[ i ] = file.readChar();**

**name = new String( n );**

**char title[] = new char[ 15 ];**

**for ( int i = 0; i < title.length; i++ )**

**title[ i ] = file.readChar();**

**majorname = new String( title );**

**earned = file.readInt();**

**majorscore = file.readDouble();**

**majorscore2 = file.readDouble();**

**optionalclass1 = file.readDouble();**

**optionalclass2 = file.readDouble();**

**}//read**

**// Write a record to the specified RandomAccessFile**

**public void write( RandomAccessFile file ) throws IOException**

**{**

**StringBuffer buf;**

**file.writeInt(recordxe );**

**if ( name != null )**

**buf = new StringBuffer( name );**

**else**

**buf = new StringBuffer( 15 );**

**buf.setLength( 15 );**

**file.writeChars( buf.toString() );**

**if ( majorname != null )**

**buf = new StringBuffer( majorname );**

**else**

**buf = new StringBuffer( 15 );**

**buf.setLength( 15 );**

**file.writeChars( buf.toString() );**

**file.writeInt( earned );**

**file.writeDouble( majorscore);**

**file.writeDouble( majorscore2 );**

**file.writeDouble( optionalclass1);**

**file.writeDouble( optionalclass2);**

**}//write**

**public void setrecordxe( int r ) { recordxe = r; }**

**public int getrecordxe() { return recordxe; }**

**public void setname( String n ) { name= n; }**

**public String getname() { return name; }**

**public void setmajorname( String mn ) { majorname = mn; }**

**public String getmajorname() { return majorname; }**

**public void setearned( int ed ) {earned = ed; }**

**public int getearned() { return earned; }**

**public void setmajorscore( double ms ) { majorscore = ms; }**

**public double getmajorscore() { return majorscore;}**

**public void setmajorscore2( double m2 ) { majorscore2 = m2;}**

**public double getmajorscore2() { return majorscore2;}**

**public void setoptionalclass1( double oc ) { optionalclass1 = oc; }**

**public double getoptionalclass1() { return optionalclass1;}**

**public void setoptionalclass2( double oc2 ) { optionalclass2 = oc2; }**

**public double getoptionalclass2() { return optionalclass2;}**

**public static int size() { return 100; } //SIZE OF RECORD IN BYTES!!!!**

**}**

**//============================================================================================**

**//CreateXe.java**

**import java.io.\*;**

**public class CreateXe{**

**private Recordxe blank;**

**private RandomAccessFile file;**

**public CreateXe()**

**{**

**blank = new Recordxe();**

**try{**

**file = new RandomAccessFile( "studentinfro.dat", "rw" );**

**// file.seek(file.length());//EOF**

**for ( int i = 0; i < 100; i++ )**

**blank.write( file );**

**file.close();**

**}//try**

**catch( IOException e ) {**

**System.err.println( "File not opened properly\n" +**

**e.toString() );**

**System.exit( 1 );**

**}**

**}**

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

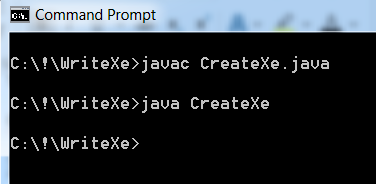
**{**

**CreateXe accounts = new CreateXe();**

**}**

**}**

**//=====================================================================================================**



**// WriteXe.java**

**/\***

**\*CIT 239 Spring Professor Jack, Final project.**

**\* This is a program about the GPA calcuation and evaluation**

**\* The student need to enter their name, major, their major score(2) optional class score (2) and credits;**

**\* The program will give the grade to the student, the score cannot over 100 or less than the 0, it will become 0;**

**\* And it will show error when the gpa under 0.0 or upper than 4.0, credits must be integer.**

**\*/**

**import java.io.\*;**

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

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

**public class WriteXe extends Frame**

**implements ActionListener {**

**private TextField nameField, majornameField, majorscoreField,**

**majorscore2Field, optionalclass1Field,**

**optionalclass2Field, earnedField, recordxeField;**

**private Button next,**

**finished;**

**private RandomAccessFile output;**

**private Recordxe data;**

**public WriteXe()**

**{**

**super("Read Student Academic Preference");**

**data = new Recordxe();**

**try {**

**output = new RandomAccessFile ("studentinfro.dat", "rw");**

**} //try**

**catch ( IOException e )**

**{**

**System.err.println( e.toString() );**

**System.exit( 1 );**

**}//catch**

**setSize (600, 300);**

**setLayout(new GridLayout(9,2));**

**setFont (new Font("verdana", Font.BOLD, 12));**

**add( new Label( "Record Number" ) );**

**recordxeField = new TextField();**

**add( recordxeField );**

**add (new Label ("Student Name"));**

**nameField = new TextField(20);**

**add( nameField );**

**add(new Label ("Student Major"));**

**majornameField = new TextField(20);**

**add( majornameField );**

**add(new Label (" Major score1 ") );**

**majorscoreField = new TextField (20);**

**add(majorscoreField);**

**add(new Label (" Major score2 ") );**

**majorscore2Field = new TextField (20);**

**add(majorscore2Field);**

**add(new Label (" Optional class1 ") );**

**optionalclass1Field = new TextField (20);**

**add(optionalclass1Field);**

**add(new Label (" Optional class 2 ") );**

**optionalclass2Field = new TextField (20);**

**add(optionalclass2Field);**

**add(new Label (" Cridits You Earned ") );**

**earnedField = new TextField (20);**

**add(earnedField);**

**next = new Button( "Next" );**

**next.addActionListener( this );**

**add( next );**

**finished = new Button( "Finished" );**

**finished.addActionListener( this );**

**add( finished );**

**setVisible(true);**

**}**

**public void addRecordxe()**

**{**

**int recordxeNumber = 0;**

**int earned;**

**Double m, s, c,o;**

**if ( !recordxeField.getText().equals( "" ) ) {**

**try {**

**recordxeNumber = Integer.parseInt( recordxeField.getText() );**

**}**

**catch(NumberFormatException nfe) {**

**System.err.println("Record Number must be entered as an Integer");**

**}**

**if ( recordxeNumber > 0 && recordxeNumber <=100 ) {**

**data.setrecordxe( recordxeNumber);**

**try {**

**data.setname( nameField.getText() );**

**data.setmajorname( majornameField.getText() );**

**earned = //credits earned**

**Integer.parseInt( earnedField.getText() );**

**data.setearned( earned );**

**}**

**catch(NumberFormatException nfe) {**

**System.err.println("Credits which you earned must be entered as an Integer");**

**}**

**try {**

**m = new Double ( majorscoreField.getText() );**

**if (m>100 || m <0){**

**System.err.println("no, you first major score cannot be over 100 or less than zero");**

**}**

**else {**

**data.setmajorscore( m.doubleValue() );**

**}**

**}**

**catch(NumberFormatException nfe) {**

**System.err.println("Your major score must be entered as a double");**

**}**

**try {**

**s = new Double ( majorscore2Field.getText() );**

**if (s>100 || s <0){**

**System.err.println("no, you second major score cannot be over 100 or less than zero");**

**}**

**else {**

**data.setmajorscore2( s.doubleValue() );**

**}**

**}**

**catch(NumberFormatException nfe) {**

**System.err.println( "You second major score must be entered as a double too " );**

**}**

**try {**

**o = new Double ( optionalclass1Field.getText() );**

**if (o>100 || o <0){**

**System.err.println("no, you first optional class score cannot be over 100 or less than zero");**

**}**

**else {**

**data.setoptionalclass1( o.doubleValue() );**

**}**

**}**

**catch(NumberFormatException nfe) {**

**System.err.println( "Your optional class score must be entered as a double" );**

**}**

**try {**

**c = new Double ( optionalclass2Field.getText() );**

**if (c>100 || c <0){ // remove the number which user enter over 100 or less 0**

**System.err.println("no, you second optional class score cannot be over 100 or less than zero");**

**}**

**else {**

**data.setoptionalclass2( c.doubleValue() );**

**}**

**}**

**catch(NumberFormatException nfe) {**

**System.err.println( "Your optional class score must be entered as a double" );**

**}**

**try{**

**output.seek(**

**(long) ( recordxeNumber-1 ) \* Recordxe.size() );**

**data.write( output );**

**}**

**catch ( IOException io ) {**

**System.err.println(**

**"Error during write to file\n" +**

**io.toString() );**

**System.exit( 1 );**

**}//catch**

**}**

**//Clear TextFields!!!!!**

**nameField.setText("");**

**majornameField.setText("");**

**majorscoreField.setText("");**

**majorscore2Field.setText("");**

**optionalclass1Field.setText("");**

**optionalclass2Field.setText("");**

**earnedField.setText("");**

**recordxeField.setText("");**

**}**

**}**

**public void actionPerformed( ActionEvent e )**

**{**

**addRecordxe();**

**if ( e.getSource() == finished ) {**

**try {**

**output.close();**

**}//try**

**catch ( IOException io ) {**

**System.err.println( "File not closed properly\n" + io.toString() );**

**System.exit(1);**

**}**

**System.exit( 0 );**

**}**

**}**

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

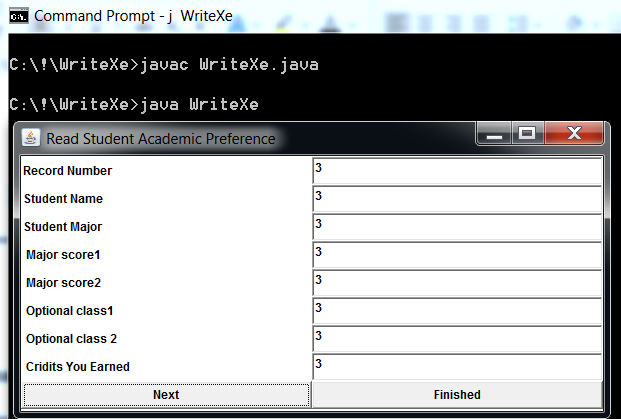
**{**

**new WriteXe();**

**}**

**}**

**//==============================================================================================**



**//=========================================================================================**

**//ReadXe.java**

**import java.io.\*;**

**public class Recordxe**

**{ //object creating**

**private int recordxe; //4**

**private String name; //15**

**private String majorname; //15**

**private double majorscore; //8**

**private double majorscore2;**

**private double optionalclass1;**

**private double optionalclass2;**

**private int earned;**

**public void read( RandomAccessFile file ) throws IOException**

**{**

**recordxe = file.readInt();**

**char n[] = new char[ 15 ];**

**for ( int i = 0; i < n.length; i++ )**

**n[ i ] = file.readChar();**

**name = new String( n );**

**char title[] = new char[ 15 ];**

**for ( int i = 0; i < title.length; i++ )**

**title[ i ] = file.readChar();**

**majorname = new String( title );**

**earned = file.readInt();**

**majorscore = file.readDouble();**

**majorscore2 = file.readDouble();**

**optionalclass1 = file.readDouble();**

**optionalclass2 = file.readDouble();**

**}//read**

**// Write a record to the specified RandomAccessFile**

**public void write( RandomAccessFile file ) throws IOException**

**{**

**StringBuffer buf;**

**file.writeInt(recordxe );**

**if ( name != null )**

**buf = new StringBuffer( name );**

**else**

**buf = new StringBuffer( 15 );**

**buf.setLength( 15 );**

**file.writeChars( buf.toString() );**

**if ( majorname != null )**

**buf = new StringBuffer( majorname );**

**else**

**buf = new StringBuffer( 15 );**

**buf.setLength( 15 );**

**file.writeChars( buf.toString() );**

**file.writeInt( earned );**

**file.writeDouble( majorscore);**

**file.writeDouble( majorscore2 );**

**file.writeDouble( optionalclass1);**

**file.writeDouble( optionalclass2);**

**}//write**

**public void setrecordxe( int r ) { recordxe = r; }**

**public int getrecordxe() { return recordxe; }**

**public void setname( String n ) { name= n; }**

**public String getname() { return name; }**

**public void setmajorname( String mn ) { majorname = mn; }**

**public String getmajorname() { return majorname; }**

**public void setearned( int ed ) {earned = ed; }**

**public int getearned() { return earned; }**

**public void setmajorscore( double ms ) { majorscore = ms; }**

**public double getmajorscore() { return majorscore;}**

**public void setmajorscore2( double m2 ) { majorscore2 = m2;}**

**public double getmajorscore2() { return majorscore2;}**

**public void setoptionalclass1( double oc ) { optionalclass1 = oc; }**

**public double getoptionalclass1() { return optionalclass1;}**

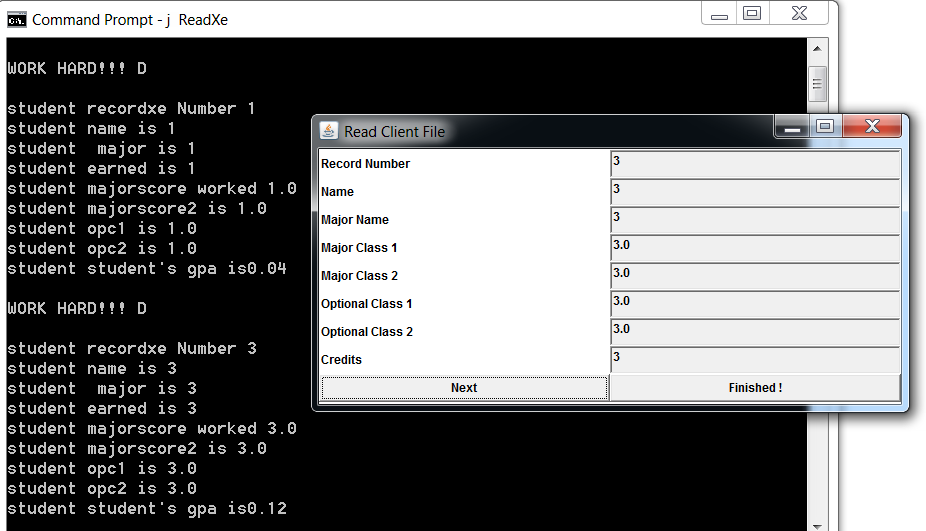
**public void setoptionalclass2( double oc2 ) { optionalclass2 = oc2; }**

**public double getoptionalclass2() { return optionalclass2;}**

**public static int size() { return 100; } //SIZE OF RECORD IN BYTES!!!!**

**}**

**//========================================================================================**



**// Now Template 3 with FX**

**// Emprec.java**

**/\***

**\* Random Access File - JavaFX**

**\***

**\***

**\***

**\*/**

**//package pkg;**

**import java.text.DecimalFormat;**

**public class Emprec implements Emprec\_Iface{**

**//define properties**

**int record;**

**String jobTitle;**

**String name;**

**int age;**

**int dependents;**

**char gender;**

**double hours;**

**double rate;**

**double salary;**

**//constructor**

**public Emprec( int record, String jobTitle, String name, int age, int dependents, char gender,**

**double hours, double rate, double salary)**

**{**

**this.record=record;**

**this.jobTitle=jobTitle;**

**this.name=name;**

**this.age=age;**

**this.dependents=dependents;**

**this.gender = gender;**

**this.hours=hours;**

**this.rate=rate;**

**this.salary=salary;**

**}**

**//interface method implementation**

**public double calc\_gross\_pay()**

**{**

**return (this.hours \* this.rate);**

**}// calc\_gross\_pay**

**//interface method implementation**

**public double calc\_fed\_tax(double hours,double rate)**

**{**

**double yearly\_income;**

**yearly\_income = hours \* rate \* TOTAL\_WEEKS; //52 - weeks per year**

**if (yearly\_income < YEAR\_INCOME\_S) return (hours \* rate \* TAX\_RATE\_ONE);**

**else if ( yearly\_income < YEAR\_INCOME\_M) return (hours \* rate\* TAX\_RATE\_TWO);**

**else return (hours \* rate \* TAX\_RATE\_THREE);**

**}// calc\_fed\_tax**

**//interface method implementation**

**public double calc\_state\_tax(double hours, double rate)**

**{**

**return (hours \* rate \* STATE\_TAX\_RATE);**

**}// calc\_state\_tax**

**public String toString()**

**{**

**DecimalFormat twoDigits = new DecimalFormat( "0.00" );**

**DecimalFormat dollar = new DecimalFormat( "$.00" );**

**return**

**("\n Record: "+ this.record +**

**"\n Job title: "+ this.jobTitle +**

**"\n Name: "+ this.name +**

**"\n Age: "+ this.age +**

**"\n Dependents: " + this.dependents +**

**"\n Gender: " + this.gender +**

**"\n Hours: " + twoDigits.format(this.hours) +**

**"\n Rate: " + dollar.format(this.rate) +**

**"\n Salary: " + dollar.format(this.salary) +**

**"\n Gross Pay: " + dollar.format(calc\_gross\_pay()) +**

**"\n State Tax: " + dollar.format(calc\_state\_tax(this.hours, this.rate)) +**

**"\n Federal Tax: " + dollar.format(calc\_fed\_tax(this.hours, this.rate)) );**

**}//toString**

**}**

**//Emprec\_Iface.java**

**public interface Emprec\_Iface{**

**//define and initialize taxRate**

**public double TAX\_RATE\_ONE = 0.28;**

**public double TAX\_RATE\_TWO = 0.32;**

**public double TAX\_RATE\_THREE = 0.38;**

**public double YEAR\_INCOME\_S = 30000.00;**

**public double YEAR\_INCOME\_M = 50000.00;**

**public double STATE\_TAX\_RATE = 0.0561;**

**public int TOTAL\_WEEKS = 52;**

**public double calc\_gross\_pay();**

**public double calc\_fed\_tax(double hours,double rate);**

**public double calc\_state\_tax(double hours, double rate);**

**}**

**//=========================================================**

**// Employee\_Record.java**

**/\***

**\* Random Access File - JavaFX**

**\***

**\***

**\***

**\*/**

**//package pkg;**

**import java.io.\*;**

**import java.io.RandomAccessFile;**

**import java.util.Date;**

**public class EmployeeRecord{**

**//define properties**

**private int record;**

**private String jobTitle;**

**private String name;**

**private int age;**

**private int dependents;**

**private char gender;**

**private double hours;**

**private double rate;**

**private double salary;**

**private Date date;**

**//default constructor**

**public EmployeeRecord(){**

**record = 0;**

**jobTitle = "";**

**name = "";**

**age = 0;**

**dependents = 0;**

**gender = ' ';**

**hours = 0.0;**

**rate = 0.0;**

**salary = 0.0;**

**this.date = new Date();**

**}**

**//define write method - to write data to a file**

**public void write(RandomAccessFile file) throws IOException{**

**file.writeInt(record);**

**StringBuffer buff;**

**if(jobTitle != null){**

**buff = new StringBuffer(jobTitle);**

**}else{**

**buff = new StringBuffer(15);**

**}**

**buff.setLength(15); //set length**

**file.writeChars(buff.toString()); //write string as sequense of characters**

**if(name != null){**

**buff = new StringBuffer(name);**

**}else{**

**buff = new StringBuffer(15);**

**}**

**buff.setLength(15);**

**file.writeChars(buff.toString());**

**file.writeInt(age);**

**file.writeInt(dependents);**

**file.writeChar(gender);**

**file.writeDouble(hours);**

**file.writeDouble(rate);**

**file.writeDouble(salary);**

**}**

**//define read method to read a record from a file**

**public void read(RandomAccessFile file) throws IOException{**

**record = file.readInt();**

**char[] arr = new char[15];**

**for(int i=0; i<arr.length; i++){**

**arr[i] = file.readChar();//read chars from a file**

**}**

**jobTitle = new String(arr);**

**char[] arr2 = new char[15];**

**for(int i=0; i<arr2.length; i++){**

**arr2[i] = file.readChar();//read chars from a file**

**}**

**name = new String(arr2);**

**age = file.readInt();**

**dependents = file.readInt();**

**gender = file.readChar();**

**hours = file.readDouble();**

**rate = file.readDouble();**

**salary = file.readDouble();**

**}**

**//define mutator methods**

**public void setRecord(int record){**

**this.record = record;**

**}**

**public void setJobTitle(String jobTitle){**

**this.jobTitle = jobTitle;**

**}**

**public void setName(String name){**

**this.name = name;**

**}**

**public void setAge(int age){**

**this.age = age;**

**}**

**public void setDependents(int dependents){**

**this.dependents = dependents;**

**}**

**public void setGender(char gender){**

**this.gender = gender;**

**}**

**public void setHours(double hours){**

**this.hours = hours;**

**}**

**public void setRate(double rate){**

**this.rate = rate;**

**}**

**public void setSalary(double salary){**

**this.salary = salary;**

**}**

**//define accessor methods**

**public int getRecord(){**

**return this.record;**

**}**

**public String getJobTitle(){**

**return this.jobTitle;**

**}**

**public String getName(){**

**return this.name;**

**}**

**public int getAge(){**

**return this.age;**

**}**

**public int getDependents(){**

**return this.dependents;**

**}**

**public char getGender(){**

**return this.gender;**

**}**

**public double getHours(){**

**return this.hours;**

**}**

**public double getRate(){**

**return this.rate;**

**}**

**public double getSalary(){**

**return this.salary;**

**}**

**public Date getDate(){**

**return this.date;**

**}**

**//define size() method which return size of a record**

**public static int size(){**

**return 98; //12+60+2+24=98**

**}**

**}**

**//=============================================================================================**

**// WriteEmployeeRecord.java**

**/\***

**\* Random Access File - JavaFX**

**\***

**\***

**\***

**\*/**

**//import pkg.EmployeeRecord;**

**import javafx.application.Application;**

**import javafx.scene.Scene;**

**import javafx.scene.control.Button;**

**import javafx.stage.Stage;**

**import javafx.scene.layout.StackPane;//place nodes on top on each other**

**import javafx.scene.control.Label;**

**import javafx.scene.control.TextField;**

**import javafx.scene.layout.FlowPane; //place horiz and vert nodes**

**import javafx.scene.layout.VBox;//place nodes in a single column**

**import javafx.scene.layout.HBox;//place nodes in a single row**

**import javafx.scene.layout.GridPane; //grid**

**import javafx.geometry.Pos;**

**import javafx.geometry.HPos;**

**import javafx.geometry.Insets;**

**import javafx.scene.text.Font;**

**import javafx.event.ActionEvent;**

**import javafx.event.EventHandler;**

**import java.io.RandomAccessFile;**

**import java.io.\*;**

**public class WriteEmployeeRecord extends Application{**

**//define text fields**

**private TextField RecordField, JobTitleField, NameField, AgeField, DependentsField, GenderField, HoursField, RateField, SalaryField;**

**private Button done, next;**

**private RandomAccessFile output;**

**private EmployeeRecord employeeRecord;**

**//override start() - to start the application**

**public void start(Stage primaryStage){**

**employeeRecord = new EmployeeRecord();//create employee record object**

**//open a file**

**try{**

**output = new RandomAccessFile("employeeRecord.dat", "rw");**

**}catch(IOException er){**

**System.err.println("Could not opened a file: " + er.toString());**

**System.exit(1);**

**}**

**//create a pane**

**GridPane pane = new GridPane();**

**pane.setAlignment(Pos.CENTER);**

**pane.setHgap(8);//set horizontal gap**

**pane.setVgap(5);//set vertical gap**

**//set padding - inside offset**

**pane.setPadding(new Insets(10, 10, 10, 10));**

**//create form**

**pane.add(new Label("Record #: "), 0, 0); //column, row**

**RecordField = new TextField();**

**pane.add(RecordField, 1, 0);**

**pane.add(new Label("Job Title: "), 0, 1);**

**JobTitleField = new TextField();**

**pane.add(JobTitleField, 1, 1);**

**pane.add(new Label("Employee Name: "), 0, 2);**

**NameField = new TextField();**

**pane.add(NameField, 1, 2);**

**pane.add(new Label("Employee Age: "), 0, 3);**

**AgeField = new TextField();**

**pane.add(AgeField, 1, 3);**

**pane.add(new Label("Dependents #: "), 0, 4);**

**DependentsField = new TextField();**

**pane.add(DependentsField, 1, 4);**

**pane.add(new Label("Gender: "), 0, 5);**

**GenderField = new TextField();**

**pane.add(GenderField, 1, 5);**

**pane.add(new Label("Hours: "), 0, 6);**

**HoursField = new TextField();**

**pane.add(HoursField, 1, 6);**

**pane.add(new Label("Rate: "), 0, 7);**

**RateField = new TextField();**

**pane.add(RateField, 1, 7);**

**pane.add(new Label("Salary: "), 0, 8);**

**SalaryField = new TextField();**

**pane.add(SalaryField, 1, 8);**

**next = new Button("Next");**

**next.setMaxWidth(Double.MAX\_VALUE);**

**NextButton handler1 = new NextButton();//create handler**

**next.setOnAction(handler1); //register handler**

**pane.add(next, 0, 9);**

**GridPane.setHalignment(next, HPos.LEFT);**

**done = new Button("Done");**

**done.setMaxWidth(Double.MAX\_VALUE);**

**DoneButton handler2 = new DoneButton(); //create handler**

**done.setOnAction(handler2);//register handler**

**pane.add(done, 1, 9);**

**GridPane.setHalignment(done, HPos.RIGHT);**

**//create scene**

**Scene scene = new Scene(pane);**

**//set app title**

**primaryStage.setTitle("Employee Record - Write");**

**primaryStage.setWidth(400);**

**primaryStage.setHeight(400);**

**//set scene**

**primaryStage.setScene(scene);**

**//display the stage**

**primaryStage.show();**

**}//start**

**void addEmployeeRecord(){**

**//define temporary variables**

**int recordTemp = 0;**

**int ageTemp = 0;**

**int depenTemp = 0;**

**double hoursTemp, rateTemp, salaryTemp;**

**char genderTemp;**

**if(!RecordField.getText().equals("")){**

**try{**

**recordTemp = Integer.parseInt(RecordField.getText());**

**if(recordTemp >= 1){**

**employeeRecord.setRecord(recordTemp); //initialize a record**

**}**

**}catch(NumberFormatException nfe){**

**System.err.println("Record number must be entered as an Integer");**

**}**

**employeeRecord.setJobTitle(JobTitleField.getText());//initialize a job title**

**employeeRecord.setName(NameField.getText());//initialize a name**

**try{**

**ageTemp = Integer.parseInt(AgeField.getText());**

**employeeRecord.setAge(ageTemp);//initialize a number of dependents**

**}catch(NumberFormatException nfe){**

**System.err.println("Number must be entered as an Integer");**

**}**

**try{**

**depenTemp = Integer.parseInt(DependentsField.getText());**

**employeeRecord.setDependents(depenTemp);//initialize a number of dependents**

**}catch(NumberFormatException nfe){**

**System.err.println("Number must be entered as an Integer");**

**}**

**try{**

**//initialize gender**

**genderTemp = GenderField.getText().charAt(0);**

**employeeRecord.setGender(genderTemp);**

**}catch(IndexOutOfBoundsException nfe){**

**System.err.println( "Gender must be entered as a one character" );**

**}**

**try{**

**hoursTemp = new Double(HoursField.getText());**

**employeeRecord.setHours(hoursTemp);**

**}catch(NumberFormatException nfe){**

**System.err.println("Number must be entered as a Double");**

**}**

**try{**

**rateTemp = new Double(RateField.getText());**

**employeeRecord.setRate(rateTemp);**

**}catch(NumberFormatException nfe){**

**System.err.println("Number must be entered as a Double");**

**}**

**try{**

**salaryTemp = new Double(SalaryField.getText());**

**employeeRecord.setSalary(salaryTemp);**

**}catch(NumberFormatException nfe){**

**System.err.println("Number must be entered as a Double");**

**}**

**//write information to a file**

**try{**

**output.seek((long)(recordTemp-1) \* EmployeeRecord.size());//set pointer**

**employeeRecord.write(output);**

**}catch(IOException er){**

**System.err.println("Error during write to a file: " + er.toString());**

**System.exit(1);**

**}**

**}//if statement**

**//clean text fields**

**RecordField.setText("");**

**JobTitleField.setText("");**

**NameField.setText("");**

**AgeField.setText("");**

**DependentsField.setText("");**

**GenderField.setText("");**

**HoursField.setText("");**

**RateField.setText("");**

**SalaryField.setText("");**

**}//addEmployeeRecord**

**//Inner classes - useful for defining handler classes**

**class NextButton implements EventHandler<ActionEvent>{**

**public void handle(ActionEvent e){**

**addEmployeeRecord();**

**}**

**}//NextButton**

**class DoneButton implements EventHandler<ActionEvent>{**

**public void handle(ActionEvent e){**

**try{**

**output.close(); //close a file**

**}catch(IOException er){**

**System.err.println("File not closed properly: " + er.toString());**

**System.exit(1);**

**}**

**System.exit(0);**

**}**

**}//DoneButton**

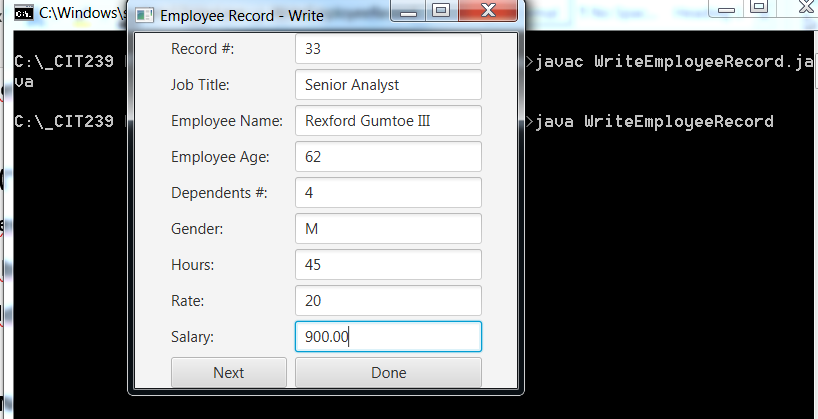
**public static void main(String[] args){**

**launch(args);**

**}**

**}//WriteEmployeeRecord**

**//===================================================================================================**



**//ReadEmployeeRecord.java**

**/\***

**\* Random Access File - JavaFX**

**\***

**\***

**\***

**\*/**

**//import pkg.Emprec;**

**//import pkg.EmployeeRecord;**

**import javafx.application.Application;**

**import javafx.scene.Scene;**

**import javafx.scene.control.Button;**

**import javafx.stage.Stage;**

**import javafx.scene.layout.StackPane;//place nodes on top on each other**

**import javafx.scene.control.Label;**

**import javafx.scene.control.TextField;**

**import javafx.scene.layout.FlowPane; //place horiz and vert nodes**

**import javafx.scene.layout.VBox;//place nodes in a single column**

**import javafx.scene.layout.HBox;//place nodes in a single row**

**import javafx.scene.layout.GridPane; //grid**

**import javafx.geometry.Pos;**

**import javafx.geometry.HPos;**

**import javafx.geometry.Insets;**

**import javafx.scene.text.Font;**

**import javafx.event.ActionEvent;**

**import javafx.event.EventHandler;**

**import java.io.RandomAccessFile;**

**import java.io.\*;**

**import java.text.DecimalFormat;**

**import java.text.NumberFormat;**

**import java.text.SimpleDateFormat;**

**import java.util.Date;**

**public class ReadEmployeeRecord extends Application{**

**//define text fields**

**private TextField DateField, RecordField, JobTitleField, NameField, AgeField, DependentsField,**

**GenderField, HoursField, RateField, SalaryField, GrossPayField, StateTaxField, FedTaxField;**

**private Button done, next;**

**private RandomAccessFile input;**

**private EmployeeRecord employeeRecord;**

**private Emprec emprec;**

**//override start() - to start the application**

**public void start(Stage primaryStage){**

**employeeRecord = new EmployeeRecord();//create employee record object**

**//open a file**

**try{**

**input = new RandomAccessFile("employeeRecord.dat", "r");**

**}catch(IOException er){**

**System.err.println("Could not opened a file: " + er.toString());**

**System.exit(1);**

**}**

**employeeRecord = new EmployeeRecord(); //initialize object**

**//create a pane**

**GridPane pane = new GridPane();**

**pane.setAlignment(Pos.CENTER);**

**pane.setHgap(8);//set horizontal gap**

**pane.setVgap(5);//set vertical gap**

**//set padding - inside offset**

**pane.setPadding(new Insets(10, 10, 10, 10));**

**//create form**

**pane.add(new Label("Date: "), 0, 0);**

**DateField = new TextField();**

**DateField.setEditable(false);**

**pane.add(DateField, 1, 0);**

**pane.add(new Label("Record #: "), 0, 1); //column, row**

**RecordField = new TextField();**

**RecordField.setEditable(false);**

**pane.add(RecordField, 1, 1);**

**pane.add(new Label("Job Title: "), 0, 2);**

**JobTitleField = new TextField();**

**JobTitleField.setEditable(false);**

**pane.add(JobTitleField, 1, 2);**

**pane.add(new Label("Employee Name: "), 0, 3);**

**NameField = new TextField();**

**NameField.setEditable(false);**

**pane.add(NameField, 1, 3);**

**pane.add(new Label("Employee Age: "), 0, 4);**

**AgeField = new TextField();**

**AgeField.setEditable(false);**

**pane.add(AgeField, 1, 4);**

**pane.add(new Label("Dependents #: "), 0, 5);**

**DependentsField = new TextField();**

**DependentsField.setEditable(false);**

**pane.add(DependentsField, 1, 5);**

**pane.add(new Label("Gender: "), 0, 6);**

**GenderField = new TextField();**

**GenderField.setEditable(false);**

**pane.add(GenderField, 1, 6);**

**pane.add(new Label("Hours: "), 0, 7);**

**HoursField = new TextField();**

**HoursField.setEditable(false);**

**pane.add(HoursField, 1, 7);**

**pane.add(new Label("Rate: "), 0, 8);**

**RateField = new TextField();**

**RateField.setEditable(false);**

**pane.add(RateField, 1, 8);**

**pane.add(new Label("Salary: "), 0, 9);**

**SalaryField = new TextField();**

**SalaryField.setEditable(false);**

**pane.add(SalaryField, 1, 9);**

**pane.add(new Label("Gross Pay: "), 0, 10);**

**GrossPayField = new TextField();**

**GrossPayField.setEditable(false);**

**pane.add(GrossPayField, 1, 10);**

**pane.add(new Label("State Tax: "), 0, 11);**

**StateTaxField = new TextField();**

**StateTaxField.setEditable(false);**

**pane.add(StateTaxField, 1, 11);**

**pane.add(new Label("Federal Tax: "), 0, 12);**

**FedTaxField = new TextField();**

**FedTaxField.setEditable(false);**

**pane.add(FedTaxField, 1, 12);**

**next = new Button("Next");**

**next.setMaxWidth(Double.MAX\_VALUE);**

**NextButton handler1 = new NextButton();//create handler**

**next.setOnAction(handler1); //register handler**

**pane.add(next, 0, 13);**

**GridPane.setHalignment(next, HPos.LEFT);**

**done = new Button("Done");**

**done.setMaxWidth(Double.MAX\_VALUE);**

**DoneButton handler2 = new DoneButton(); //create handler**

**done.setOnAction(handler2);//register handler**

**pane.add(done, 1, 13);**

**GridPane.setHalignment(done, HPos.RIGHT);**

**//create scene**

**Scene scene = new Scene(pane);**

**//set app title**

**primaryStage.setTitle("Employee Record - Read");**

**primaryStage.setWidth(400);**

**primaryStage.setHeight(600);**

**//set scene**

**primaryStage.setScene(scene);**

**//display the stage**

**primaryStage.show();**

**}//start**

**void readEmployeeRecord(){**

**DecimalFormat twoDigits = new DecimalFormat( "0.00" );**

**//DecimalFormat dollar = new DecimalFormat( "$.00" );**

**NumberFormat dollar = NumberFormat.getCurrencyInstance();**

**//define temporary variables**

**int recordTemp = 0;**

**int ageTemp = 0;**

**int depenTemp = 0;**

**double hoursTemp, rateTemp, salaryTemp;**

**char genderTemp;**

**String jobTemp, nameTemp;**

**try{**

**do{**

**//read from a file**

**employeeRecord.read(input);**

**recordTemp = employeeRecord.getRecord();**

**jobTemp = employeeRecord.getJobTitle();**

**nameTemp = employeeRecord.getName();**

**ageTemp = employeeRecord.getAge();**

**depenTemp = employeeRecord.getDependents();**

**genderTemp = employeeRecord.getGender();**

**hoursTemp = employeeRecord.getHours();**

**rateTemp = employeeRecord.getRate();**

**salaryTemp = employeeRecord.getSalary();**

**emprec = new Emprec(recordTemp, jobTemp, nameTemp, ageTemp, depenTemp, genderTemp, hoursTemp, rateTemp, salaryTemp);**

**}while(employeeRecord.getRecord() == 0);//?????**

**System.out.println(emprec.toString());**

**//get Date field**

**Date newDate = employeeRecord.getDate();**

**SimpleDateFormat dateformat = new SimpleDateFormat("MM/dd/YYYY, hh:mm");**

**//write employee information to a form**

**DateField.setText(String.valueOf(dateformat.format(newDate)));**

**RecordField.setText(String.valueOf(recordTemp));**

**JobTitleField.setText(jobTemp);**

**NameField.setText(nameTemp);**

**AgeField.setText(String.valueOf(ageTemp));**

**DependentsField.setText(String.valueOf(depenTemp));**

**GenderField.setText(String.valueOf(genderTemp));**

**HoursField.setText(String.valueOf(twoDigits.format(hoursTemp)));**

**RateField.setText(String.valueOf(dollar.format(rateTemp)));**

**SalaryField.setText(String.valueOf(dollar.format(salaryTemp)));**

**GrossPayField.setText(String.valueOf(dollar.format(emprec.calc\_gross\_pay())));**

**StateTaxField.setText(String.valueOf(dollar.format(emprec.calc\_state\_tax(hoursTemp, rateTemp))));**

**FedTaxField.setText(String.valueOf(dollar.format(emprec.calc\_fed\_tax(hoursTemp, rateTemp))));**

**}catch(EOFException er){**

**closeFile();**

**}catch(IOException er){**

**System.err.println( "Error during read from file: " + er.toString() );**

**System.exit( 1 );**

**}**

**}//readEmployeeRecord**

**void closeFile(){**

**try{**

**input.close(); //close a file**

**}catch(IOException er){**

**System.err.println("File not closed properly: " + er.toString());**

**System.exit(1);**

**}**

**System.exit(0);**

**}**

**//Inner classes - useful for defining handler classes**

**class NextButton implements EventHandler<ActionEvent>{**

**public void handle(ActionEvent e){**

**readEmployeeRecord();**

**}**

**}//NextButton**

**class DoneButton implements EventHandler<ActionEvent>{**

**public void handle(ActionEvent e){**

**closeFile();**

**}**

**}//DoneButton**

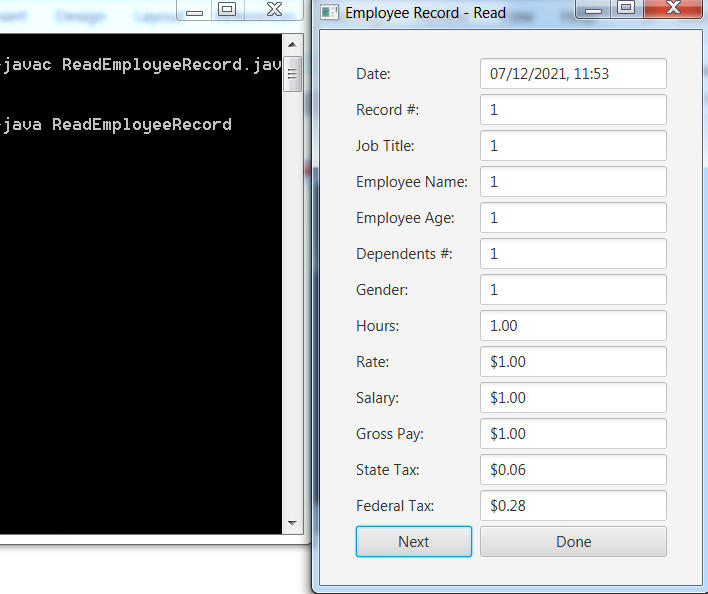
**public static void main(String[] args){**

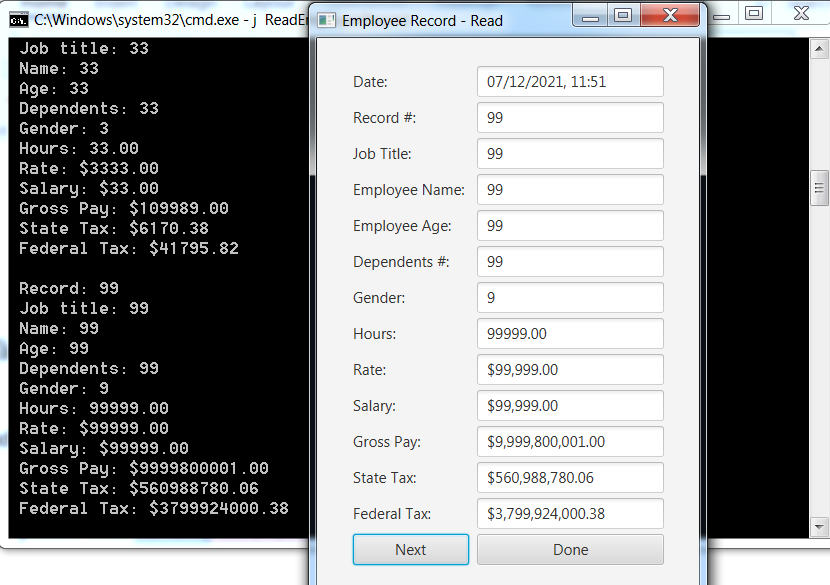
**launch(args);**

**}**

**}//ReadEmployeeRecord**

**//==================================================================================================**





C’est Finis !!!!!!