KK14203
OBJECT ORIENTED PROGRAMMING

PROJECT 2

LECTURER

DR. SAMRY@MOHD SHAMRIE SAININ

PREPARED BY

NAME	MATRIC NO.
SALMAH BINTI ASMIN	BI19110102

DATE OF SUBMISSION

6TH AUGUST 2020

TABLE OF CONTENT

JAVA CODE

```java
//NAME : SALMAH BINTI ASMIN
//MATRIC NO. : BI19110102
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;

public class CourseGrade extends JFrame{

  JFrame f1 = new JFrame ("Course Grade System");
  JFrame f2 = new JFrame();
  double grade, png;
  String grading, status;

  JLabel lblcourse = new JLabel ("KK14203 : Object-Oriented Programming");
  JLabel lbl1 = new JLabel ("Enter your name");
  JLabel lbl2 = new JLabel ("Enter your matric no.");
  JLabel lbl3 = new JLabel ("Quizzes (20%)");
  JLabel lbl4 = new JLabel ("Assignments (10%)");
  JLabel lbl5 = new JLabel ("Projects (20%)");
  JLabel lbl6 = new JLabel ("Participation (10%)");
  JLabel lbl7 = new JLabel ("Attendance (10%)");
  JLabel lbl8 = new JLabel ("Examination (30%)");
  JLabel lbl9 = new JLabel ("Universiti Malaysia Sabah");
  JLabel lbl10 = new JLabel ("Final Grade");
  JLabel lbl11 = new JLabel ("Preliminary");
  JLabel lbl12 = new JLabel ("Midterm");
  JLabel lbl13 = new JLabel ("Prefinal");
  JLabel lbl14 = new JLabel ("Final");
  JLabel lbl15 = new JLabel ("FINAL GRADE");
  JLabel lbl16 = new JLabel ("GRADE POINT");
  JLabel lbl17 = new JLabel ("GRADE");
  JLabel lbl18 = new JLabel ("STUDENT STATUS");

  JTextField txt1 = new JTextField (15);
  JTextField txt2 = new JTextField (15);
  JTextField txt3 = new JTextField (2);
  JTextField txt4 = new JTextField (2);
  JTextField txt5 = new JTextField (2);
  JTextField txt6 = new JTextField (2);
  JTextField finalgrade = new JTextField (5);
  JTextField finalrating = new JTextField (5);
  JTextField finalgrading = new JTextField (5);
  JTextField finalstatus = new JTextField (10);

  String output1 = "";
```

```
String output2 = "";
String output3 = "";
String output4 = "";
String output5 = "";
String output6 = "";

JButton btn1 = new JButton ("OK");
JButton btn2 = new JButton ("Cancel");
JButton btn3 = new JButton ("Compute");
JButton btn4 = new JButton ("Compute");
JButton btn5 = new JButton ("Compute");
JButton btn6 = new JButton ("Compute");
JButton jcompute = new JButton ("Compute All");
JButton jclear = new JButton ("Clear All");
JButton jexit = new JButton ("Exit");
JButton finalbutton = new JButton ("OK");
JButton finalprint = new JButton ("Print");

//preliminary
JComboBox cb1 = new JComboBox();
JComboBox cb2 = new JComboBox();
JComboBox cb3 = new JComboBox();
JComboBox cb4 = new JComboBox();
JComboBox cb5 = new JComboBox();
JComboBox cb6 = new JComboBox();
//midterm
JComboBox cb7 = new JComboBox();
JComboBox cb8 = new JComboBox();
JComboBox cb9 = new JComboBox();
JComboBox cb10 = new JComboBox();
JComboBox cb11 = new JComboBox();
JComboBox cb12 = new JComboBox();
//prefinal
JComboBox cb13 = new JComboBox();
JComboBox cb14 = new JComboBox();
JComboBox cb15 = new JComboBox();
JComboBox cb16 = new JComboBox();
JComboBox cb17 = new JComboBox();
JComboBox cb18 = new JComboBox();
//finals
JComboBox cb19 = new JComboBox();
JComboBox cb20 = new JComboBox();
JComboBox cb21 = new JComboBox();
JComboBox cb22 = new JComboBox();
JComboBox cb23 = new JComboBox();
JComboBox cb24 = new JComboBox();

public CourseGrade(){
   f1.getContentPane().setLayout(null);
   f1.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   f1.setSize (500,300);
```

```
    f1.setResizable(false);
    f1.getContentPane().add(lbl9);
    f1.getContentPane().add(lblcourse);
    f1.getContentPane().add(lbl1);
    f1.getContentPane().add(lbl2);
    f1.getContentPane().add(txt1);
    f1.getContentPane().add(txt2);
    f1.getContentPane().add(btn1);
    f1.getContentPane().add(btn2);
    lbl9.setBounds(150,20,230,20);
    lblcourse.setBounds(110,25,300,50);
    lbl1.setBounds(30,70,100,75);
    lbl2.setBounds(30,120,170,75);
    txt1.setBounds(180,90,200,30);
    txt2.setBounds(180,140,150,30);
    btn1.setBounds(50,200,100,40);
    btn2.setBounds(300,200,100,40);

    btn1.addActionListener(new ActionListener(){
      public void actionPerformed (ActionEvent e){
        String x, y;
        x = txt1.getText();
        y = txt2.getText();

        if(x.isEmpty() || y.isEmpty()){
          JOptionPane.showMessageDialog(null, " Please enter your name/matric no.
");
        }else{
          f2.setTitle(x + "'s Grading");
          f2.getContentPane().setLayout(null);
          f2.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
          f2.setSize (850,600);
          f2.setResizable(false);

          f1.hide();
          f2.show();

          f2.getContentPane().add(lbl3);
          f2.getContentPane().add(lbl4);
          f2.getContentPane().add(lbl5);
          f2.getContentPane().add(lbl6);
          f2.getContentPane().add(lbl7);
          f2.getContentPane().add(lbl8);
          f2.getContentPane().add(lbl9);
          f2.getContentPane().add(lbl10);
          f2.getContentPane().add(lbl11);
          f2.getContentPane().add(lbl12);
          f2.getContentPane().add(lbl13);
          f2.getContentPane().add(lbl14);
          f2.getContentPane().add(lblcourse);
          f2.getContentPane().add(jcompute);
```

```
            f2.getContentPane().add(jclear);
            f2.getContentPane().add(jexit);

            lbl3.setBounds(30,120,110,70);
            lbl4.setBounds(30,170,110,70);
            lbl5.setBounds(30,220,90,70);
            lbl6.setBounds(30,270,120,70);
            lbl7.setBounds(30,320,100,70);
            lbl8.setBounds(30,370,110,70);
            lbl9.setBounds(280,20,230,20);
            lbl10.setBounds(30,420,80,70);
            lbl11.setBounds(185,60,100,100);
            lbl12.setBounds(315,60,100,100);
            lbl13.setBounds(440,60,100,100);
            lbl14.setBounds(575,60,100,100);
            lblcourse.setBounds(240,25,300,50);
            jcompute.setBounds(660,160,120, 60);
            jclear.setBounds(660,260,120, 60);
            jexit.setBounds(660,360,120,60);

            //PRELIMINARY
            f2.getContentPane().add(cb1);
            f2.getContentPane().add(cb2);
            f2.getContentPane().add(cb3);
            f2.getContentPane().add(cb4);
            f2.getContentPane().add(cb5);
            f2.getContentPane().add(cb6);
            f2.getContentPane().add(btn3);
            f2.getContentPane().add(txt3);
            txt3.setEditable(false);
            cb1.setBounds(190,140,50,30);
            cb2.setBounds(190,190,50,30);
            cb3.setBounds(190,240,50,30);
            cb4.setBounds(190,290,50,30);
            cb5.setBounds(190,340,50,30);
            cb6.setBounds(190,390,50,30);
            btn3.setBounds(170,490,95,40);
            txt3.setBounds(190,440,55,35);

            int addPrelim1 = 10;
            for (int i = 0; i <= addPrelim1; i++) {
               cb2.addItem(new Integer(i));
               cb4.addItem(new Integer(i));
               cb5.addItem(new Integer(i));
            }

            int addPrelim2 = 20;
            for (int i = 0; i <= addPrelim2; i++) {
               cb1.addItem(new Integer(i));
               cb3.addItem(new Integer(i));

```

```
            }

            int addPrelim3 = 30;
            for (int i = 0; i <= addPrelim3; i++) {
               cb6.addItem(new Integer(i));
            }

            //MIDTERM
            f2.getContentPane().add(cb7);
            f2.getContentPane().add(cb8);
            f2.getContentPane().add(cb9);
            f2.getContentPane().add(cb10);
            f2.getContentPane().add(cb11);
            f2.getContentPane().add(cb12);
            f2.getContentPane().add(btn4);
            f2.getContentPane().add(txt4);
            txt4.setEditable(false);
            cb7.setBounds(315,140,50,30);
            cb8.setBounds(315,190,50,30);
            cb9.setBounds(315,240,50,30);
            cb10.setBounds(315,290,50,30);
            cb11.setBounds(315,340,50,30);
            cb12.setBounds(315,390,50,30);
            btn4.setBounds(295,490,95,40);
            txt4.setBounds(315,440,55,35);

            int addMidterm1 = 10;
            for (int i = 0; i <= addMidterm1; i++) {
               cb8.addItem(new Integer(i));
               cb10.addItem(new Integer(i));
               cb11.addItem(new Integer(i));
            }

            int addMidterm2 = 20;
            for (int i = 0; i <= addMidterm2; i++) {
               cb7.addItem(new Integer(i));
               cb9.addItem(new Integer(i));
            }

            int addMidterm3 = 30;
            for (int i = 0; i <= addMidterm3; i++) {
               cb12.addItem(new Integer(i));
            }

            //PREFINAL
            f2.getContentPane().add(cb13);
            f2.getContentPane().add(cb14);
            f2.getContentPane().add(cb15);
            f2.getContentPane().add(cb16);
            f2.getContentPane().add(cb17);
            f2.getContentPane().add(cb18);
```

```
            f2.getContentPane().add(btn5);
            f2.getContentPane().add(txt5);
            txt5.setEditable(false);
            cb13.setBounds(440,140,50,30);
            cb14.setBounds(440,190,50,30);
            cb15.setBounds(440,240,50,30);
            cb16.setBounds(440,290,50,30);
            cb17.setBounds(440,340,50,30);
            cb18.setBounds(440,390,50,30);
            btn5.setBounds(420,490,95,40);
            txt5.setBounds(440,440,55,35);

            int addPrefinal1 = 10;
            for (int i = 0; i <= addPrefinal1; i++) {
               cb14.addItem(new Integer(i));
               cb16.addItem(new Integer(i));
               cb17.addItem(new Integer(i));
            }

            int addPrefinal2 = 20;
            for (int i = 0; i <= addPrefinal2; i++) {
               cb13.addItem(new Integer(i));
               cb15.addItem(new Integer(i));
            }

            int addPrefinal3 = 30;
            for (int i = 0; i <= addPrefinal3; i++) {
               cb18.addItem(new Integer(i));
            }


            //FINALS
            f2.getContentPane().add(cb19);
            f2.getContentPane().add(cb20);
            f2.getContentPane().add(cb21);
            f2.getContentPane().add(cb22);
            f2.getContentPane().add(cb23);
            f2.getContentPane().add(cb24);
            f2.getContentPane().add(btn6);
            f2.getContentPane().add(txt6);
            txt6.setEditable(false);
            cb19.setBounds(565,140,50,30);
            cb20.setBounds(565,190,50,30);
            cb21.setBounds(565,240,50,30);
            cb22.setBounds(565,290,50,30);
            cb23.setBounds(565,340,50,30);
            cb24.setBounds(565,390,50,30);
            btn6.setBounds(545,490,95,40);
            txt6.setBounds(565,440,55,35);

            int addFinal1 = 10;
```

```
            for (int i = 0; i <= addFinal1; i++) {
              cb20.addItem(new Integer(i));
              cb22.addItem(new Integer(i));
              cb23.addItem(new Integer(i));
            }

            int addFinal2 = 20;
            for (int i = 0; i <= addFinal2; i++) {
               cb19.addItem(new Integer(i));
               cb21.addItem(new Integer(i));
            }

            int addFinal3 = 30;
            for (int i = 0; i <= addFinal3; i++) {
               cb24.addItem(new Integer(i));
            }
         }
      }
   });

   //FOR 'CANCEL' BUTTON
   btn2.addActionListener(new ActionListener(){
      public void actionPerformed (ActionEvent e){
         System.exit(0);
      }
   });

   //TO COMPUTE PRELIMINARY MARKS
   btn3.addActionListener(new ActionListener(){
      public void actionPerformed (ActionEvent e){

         int cb1Int = Integer.parseInt(cb1.getSelectedItem().toString());
         int cb2Int = Integer.parseInt(cb2.getSelectedItem().toString());
         int cb3Int = Integer.parseInt(cb3.getSelectedItem().toString());
         int cb4Int = Integer.parseInt(cb4.getSelectedItem().toString());
         int cb5Int = Integer.parseInt(cb5.getSelectedItem().toString());
         int cb6Int = Integer.parseInt(cb6.getSelectedItem().toString());

         txt3.setText(String.valueOf(cb1Int + cb2Int + cb3Int + cb4Int + cb5Int +
cb6Int));
      }
   });

   //TO COMPUTE MIDTERM MARKS
   btn4.addActionListener(new ActionListener(){
      public void actionPerformed (ActionEvent e){

         int cb7Int = Integer.parseInt(cb7.getSelectedItem().toString());
         int cb8Int = Integer.parseInt(cb8.getSelectedItem().toString());
         int cb9Int = Integer.parseInt(cb9.getSelectedItem().toString());
         int cb10Int = Integer.parseInt(cb10.getSelectedItem().toString());
```

```
353            int cb11Int = Integer.parseInt(cb11.getSelectedItem().toString());
354            int cb12Int = Integer.parseInt(cb12.getSelectedItem().toString());
355
356            txt4.setText(String.valueOf(cb7Int + cb8Int + cb9Int + cb10Int + cb11Int +
cb12Int));
357         }
358      });
359
360      //TO COMPUTE PREFINAL MARKS
361      btn5.addActionListener(new ActionListener(){
362         public void actionPerformed (ActionEvent e){
363
364            int cb13Int = Integer.parseInt(cb13.getSelectedItem().toString());
365            int cb14Int = Integer.parseInt(cb14.getSelectedItem().toString());
366            int cb15Int = Integer.parseInt(cb15.getSelectedItem().toString());
367            int cb16Int = Integer.parseInt(cb16.getSelectedItem().toString());
368            int cb17Int = Integer.parseInt(cb17.getSelectedItem().toString());
369            int cb18Int = Integer.parseInt(cb18.getSelectedItem().toString());
370
371            txt5.setText(String.valueOf(cb13Int + cb14Int + cb15Int + cb16Int + cb17Int
+ cb18Int));
372         }
373      });
374
375
376      //TO COMPUTE FINAL MARKS
377      btn6.addActionListener(new ActionListener(){
378         public void actionPerformed (ActionEvent e){
379
380            int cb19Int = Integer.parseInt(cb19.getSelectedItem().toString());
381            int cb20Int = Integer.parseInt(cb20.getSelectedItem().toString());
382            int cb21Int = Integer.parseInt(cb21.getSelectedItem().toString());
383            int cb22Int = Integer.parseInt(cb22.getSelectedItem().toString());
384            int cb23Int = Integer.parseInt(cb23.getSelectedItem().toString());
385            int cb24Int = Integer.parseInt(cb24.getSelectedItem().toString());
386
387            txt6.setText(String.valueOf(cb19Int + cb20Int + cb21Int + cb22Int + cb23Int
+ cb24Int));
388         }
389      });
390
391      //TO CALCULATE FINAL GRADE, DETERMINE THE GRADE POINT AND STATUS
392      jcompute.addActionListener(new ActionListener(){
393         public void actionPerformed (ActionEvent e){
394
395            String prelim, midterm, prefinal, finals, total;
396            double a, b, c, d, totalgrade;
397
398            prelim = txt3.getText();
399            midterm = txt4.getText();
400            prefinal = txt5.getText();
```

```
401         finals = txt6.getText();
402
403         a = Double.parseDouble(prelim);
404         b = Double.parseDouble(midterm);
405         c = Double.parseDouble(prefinal);
406         d = Double.parseDouble(finals);
407
408         totalgrade = (a + b + c + d)/4;
409         total = Double.toString(totalgrade);
410         finalgrade.setText(total);
411
412         JFrame f3 = new JFrame ("Student Final Grading");
413
414         f3.getContentPane().setLayout(null);
415         f3.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
416         f3.setSize (350,400);
417         f3.setResizable(false);
418
419         f2.hide();
420         f3.show();
421
422         f3.getContentPane().add(lbl9);
423         f3.getContentPane().add(lblcourse);
424         f3.getContentPane().add(lbl15);
425         f3.getContentPane().add(lbl16);
426         f3.getContentPane().add(lbl17);
427         f3.getContentPane().add(lbl18);
428         f3.getContentPane().add(finalgrade);
429         finalgrade.setEditable(false);
430         f3.getContentPane().add(finalrating);
431         finalrating.setEditable(false);
432         f3.getContentPane().add(finalgrading);
433         finalgrading.setEditable(false);
434         f3.getContentPane().add(finalstatus);
435         finalstatus.setEditable(false);
436         f3.getContentPane().add(finalbutton);
437         f3.getContentPane().add(finalprint);
438
439         lbl9.setBounds(95,20,230,20);
440         lblcourse.setBounds(55,25,300,50);
441         lbl15.setBounds(90,80,150,30);
442         lbl16.setBounds(90,115,150,30);
443         lbl17.setBounds(145,160,50,30);
444         lbl18.setBounds(30,240,150,30);
445         finalgrade.setBounds(175,80,50,30);
446         finalrating.setBounds(175,115,50,30);
447         finalgrading.setBounds(140,190,50,30);
448         finalstatus.setBounds(140,240,150,30);
449         finalbutton.setBounds(70,310,70,30);
450         finalprint.setBounds(180,310,70,30);
451
```

```
452            //GRADE POINT, GRADE AND STATUS
453            grade = Double.parseDouble(finalgrade.getText());
454
455            if(grade<101 && grade>=80.00){
456              png = 4.00;
457              grading = "A";
458              status = "Pass with Distinction";
459            }else if(grade<80.00 && grade>=75.00){
460              png = 3.67;
461              grading = "A-";
462              status = "Pass with Distinction";
463            }else if(grade<75.00 && grade>=70.00){
464              png = 3.33;
465              grading = "B+";
466              status = "Pass with Credit";
467            }else if(grade<70.00 && grade>=65.00){
468              png = 3.00;
469              grading = "B";
470              status = "Pass with Credit";
471            }else if(grade<65.00 && grade>=60.00){
472              png = 2.67;
473              grading = "B-";
474              status = "Pass with Credit";
475            }else if(grade<60.00 && grade>=55.00){
476              png = 2.33;
477              grading = "C+";
478              status = "Pass";
479            }else if(grade<55.0 && grade>=50.00){
480              png = 2.00;
481              grading = "C";
482              status = "Pass";
483            }else if(grade<50.00 && grade>=45.00){
484              png = 1.67;
485              grading = "C-";
486              status = "Pass";
487            }else if(grade<44.00 && grade>=40.00){
488              png = 1.33;
489              grading = "D+";
490              status = "Pass";
491            }else if(grade<40.00 && grade>=35.00){
492              png = 1.00;
493              grading = "D";
494              status = "Pass";
495            }else{
496              png = 0.00;
497              grading = "E";
498              status = "Fail";
499            }
500
501            finalrating.setText("" + png);
502            finalrating.setEditable(false);
```

```
            finalgrading.setText(grading);
            finalgrading.setEditable(false);
            finalstatus.setText(status);
            finalstatus.setEditable(false);
          }
        });

        finalbutton.addActionListener(new ActionListener() {
          @Override
          public void actionPerformed(ActionEvent e) {
            JOptionPane.showMessageDialog(null, "" + png);

            if (png  >=1.50 && png <=4.0) JOptionPane.showMessageDialog(null, " STUDENT PASSED ");
            else if (png >=0.0 && png <=1.49) JOptionPane.showMessageDialog(null, " STUDENT FAILED ");
          }
        });

        finalprint.addActionListener(new ActionListener() {
          public void actionPerformed(ActionEvent e) {
          JOptionPane.showMessageDialog(rootPane, "File Saved as 'Course Grade.txt' in Dekstop");
          String finalname, finalmatric, printfinal, printrating, printgrading, printstatus;

          finalname = txt1.getText();
          finalmatric = txt2.getText();
          printfinal = finalgrade.getText();
          printrating = finalrating.getText();
          printgrading = finalgrading.getText();
          printstatus = finalstatus.getText();

          output1 = "Name          : " + finalname + System.lineSeparator();
          output2 = "Matric No.     : " + finalmatric + System.lineSeparator();
          output3 = "Final Grade    : " + printfinal + System.lineSeparator();
          output4 = "Grade Point    : " + printrating + System.lineSeparator();
          output5 = "Grade          : " + printgrading + System.lineSeparator();
          output6 = "Student Status : " + printstatus + System.lineSeparator();

          if(e.getSource() == finalprint){
            try {
              PrintWriter out = new PrintWriter(new FileWriter("C:\\Users\\USER\\Desktop\\Course Grade.txt"));

              out.write(output1);
              out.write(output2);
              out.write(output3);
              out.write(output4);
              out.write(output5);
              out.write(output6);
```

```
549             out.flush();
550             out.close();
551
552           } catch (IOException e1) {
553             System.err.println("Error occurred");
554             e1.printStackTrace();
555             }
556           }
557         }
558       });
559
560       jclear.addActionListener(new ActionListener(){
561         public void actionPerformed (ActionEvent e){
562           cb1.setSelectedItem(Integer.valueOf(0));
563           cb2.setSelectedItem(Integer.valueOf(0));
564           cb3.setSelectedItem(Integer.valueOf(0));
565           cb4.setSelectedItem(Integer.valueOf(0));
566           cb5.setSelectedItem(Integer.valueOf(0));
567           cb6.setSelectedItem(Integer.valueOf(0));
568           cb7.setSelectedItem(Integer.valueOf(0));
569           cb8.setSelectedItem(Integer.valueOf(0));
570           cb9.setSelectedItem(Integer.valueOf(0));
571           cb10.setSelectedItem(Integer.valueOf(0));
572           cb11.setSelectedItem(Integer.valueOf(0));
573           cb12.setSelectedItem(Integer.valueOf(0));
574           cb13.setSelectedItem(Integer.valueOf(0));
575           cb14.setSelectedItem(Integer.valueOf(0));
576           cb15.setSelectedItem(Integer.valueOf(0));
577           cb16.setSelectedItem(Integer.valueOf(0));
578           cb17.setSelectedItem(Integer.valueOf(0));
579           cb18.setSelectedItem(Integer.valueOf(0));
580           cb19.setSelectedItem(Integer.valueOf(0));
581           cb20.setSelectedItem(Integer.valueOf(0));
582           cb21.setSelectedItem(Integer.valueOf(0));
583           cb22.setSelectedItem(Integer.valueOf(0));
584           cb23.setSelectedItem(Integer.valueOf(0));
585           cb24.setSelectedItem(Integer.valueOf(0));
586
587           txt3.setText(" ");
588           txt4.setText(" ");
589           txt5.setText(" ");
590           txt6.setText(" ");
591
592         }
593       });
594
595       jexit.addActionListener(new ActionListener(){
596         public void actionPerformed (ActionEvent e){
597           System.exit(0);
598         }
599       });
```

```
    f1.show();
  }

  public static void main (String args []){
    CourseGrade xx = new CourseGrade();
  }
}
```

Also available at :

https://github.com/salmahasmin/KK14203/blob/master/Project%202

OBJECT ORIENTED CONCEPT IMPLEMENTATION

Object means a real-world entity such as a pen, chair, table, computer, watch, etc. **Object-Oriented Programming (OOP)** is a methodology or paradigm to design a program using classes and objects. It simplifies software development and maintenance by providing some concepts. In my program, there are a few OOP concepts involved:

- Object
- Class
- Inheritance
- Polymorphism
- Inner Classes
- Modifiers
- Constructors
- Exceptions

1. Object

Any entity that has state and behavior is known as an object. For example, a chair, pen, table, keyboard, bike, etc. It can be physical or logical.

An Object can be defined as an instance of a class. An object contains an address and takes up some space in memory. Objects can communicate without knowing the details of each other's data or code. The only necessary thing is the type of message accepted and the type of response returned by the objects.

To create an object, specify the class name, followed by the object name, and use the keyword new:

Example : `CourseGrade xx = new CourseGrade(); //line 605`

You can create multiple objects of one class:

```
JFrame f1 = new JFrame ("Course Grade System"); //line 12
JFrame f2 = new JFrame(); //line 13
```

You can also create an object of a class and access it in another class. This is often used for better organization of classes (one class has all the attributes and methods, while the other class holds the main() method (code to be executed)).

2. Class

Collection of objects is called class. It is a logical entity.

A class can also be defined as a blueprint from which you can create an individual object. Class doesn't consume any space.

To create a class, use the keyword class and specify the class name:

Example : `public class CourseGrade extends JFrame{ //line 10`

3. Inheritance

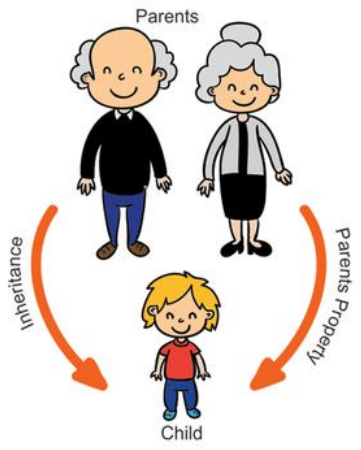
When one object acquires all the properties and behaviors of a parent object, it is known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism.

In Java, it is possible to inherit attributes and methods from one class to another. We group the "inheritance concept" into two categories:

- **subclass** (child) - the class that inherits from another class
- **superclass** (parent) - the class being inherited from

To inherit from a class, use the extends keyword. If you don't want other classes to inherit from a class, use the final keyword

Example : `public class CourseGrade extends JFrame{ //line 10`



4. Polymorphism

Polymorphism means "many forms", and it occurs when we have many classes that are related to each other by inheritance.

Like we specified previously; **Inheritance** lets us inherit attributes and methods from another class. **Polymorphism** uses those methods to perform different tasks. This allows us to perform a single action in different ways.

If one task is performed in different ways, it is known as polymorphism. For example: to convince the customer differently, to draw something, for example, shape, triangle, rectangle, etc.

In Java, we use method overloading and method overriding to achieve polymorphism.

Example : `line 392 - line 508 //TO CALCULATE FINAL GRADE, DETERMINE THE RATING AND STATUS`

5. Inner Classes

In Java, it is also possible to nest classes (a class within a class). The purpose of nested classes is to group classes that belong together, which makes your code more readable and maintainable.

To access the inner class, create an object of the outer class, and then create an object of the inner class:

Static Inner Class

An inner class can also be static, which means that you can access it without creating an object of the outer class. **Example :**

```
public static void main (String args []){ //line 604
    CourseGrade xx = new CourseGrade(); //line 605
} //line 606
```

6. Modifiers

By now, you are quite familiar with the public keyword that appears in almost all of the examples: `public class CourseGrade extends JFrame{ //line 10`

The public keyword is an **access modifier**, meaning that it is used to set the access level for classes, attributes, methods and constructors.

We divide modifiers into two groups:

- **Access Modifiers** - controls the access level
- **Non-Access Modifiers** - do not control access level, but provides other functionality

For **classes**, you can use either public or *default*:

A static method means that it can be accessed without creating an object of the class, unlike public:

Example : `public static void main (String args []){ //line 604`

7. Constructors

A constructor in Java is a **special method** that is used to initialize objects. The constructor is called when an object of a class is created. It can be used to set initial values for object attributes:

Note that the constructor name must **match the class name**, and it cannot have a **return type** (like void).

Also note that the constructor is called when the object is created.

All classes have constructors by default: if you do not create a class constructor yourself, Java creates one for you. However, then you are not able to set initial values for object attributes.

Constructors can also take parameters, which is used to initialize attributes.

Example : `public CourseGrade(){ //line 96 - class constructor for the CourseGrade class`

8. Exceptions

When executing Java code, different errors can occur: coding errors made by the programmer, errors due to wrong input, or other unforeseeable things.

When an error occurs, Java will normally stop and generate an error message. The technical term for this is: Java will throw an **exception** (throw an error).

Java try and catch

The try statement allows you to define a block of code to be tested for errors while it is being executed.

The catch statement allows you to define a block of code to be executed, if an error occurs in the try block.

The try and catch keywords come in pairs:

Example :

```
try { //line 540

}catch (IOException e1) { //line 552

}
```

READ AND WRITE IMPLEMENTATION

Read Data

In my program, there are two panels where the program has to read the user input. The first panel requires user to input the name and matric no. before going to the next panel. If the user did not enter the name or matric no. and click 'OK' button, a message will appear telling them to enter the name or matric no. first before going to the next panel.

```
105     f1.getContentPane().add(txt1);
106     f1.getContentPane().add(txt2);
 :
118     btn1.addActionListener(new ActionListener(){
119       public void actionPerformed (ActionEvent e){
120         String x, y;
121         x = txt1.getText();
122         y = txt2.getText();
123
124         if(x.isEmpty() || y.isEmpty()){
125           JOptionPane.showMessageDialog(null, " Please enter your name/matric no. ");
126         }else{
 :
319         }
320       }
321     });
```

Variable `x` and `y` (line 121 and 122 respectively) will get the user input from `txt1` and `txt2` (line 105 and 106 respectively). Then, the if else statement (line 124 until 319) will check the condition if the user has input the name and matric number. If the user did not enter the name, matric no. or both, a message dialog will appear stating that they have to enter the name and matric no. (line 124 until 126). If the user did, they can go to the next panel, which is the Grading panel where they have to input all of the marks according to the given assessments. Then, the user have to click 'Compute' button to calculate all of the marks according to the types of examinations, which is Preliminary, Midterm, Prefinal and Finals.

```
170             //PRELIMINARY
 :
189             int addPrelim1 = 10;
190             for (int i = 0; i <= addPrelim1; i++) {
191               cb2.addItem(new Integer(i));
192               cb4.addItem(new Integer(i));
193               cb5.addItem(new Integer(i));
 :
318             }
```

Line 170 until 318 is the code for the user to input the marks by choosing from the combo box provided according to the types of examinations and assessments.

```
330     //TO COMPUTE PRELIMINARY MARKS
331     btn3.addActionListener(new ActionListener(){
:
345     //TO COMPUTE MIDTERM MARKS
346     btn4.addActionListener(new ActionListener(){
:
360     //TO COMPUTE PREFINAL MARKS
361     btn5.addActionListener(new ActionListener(){
:
376     //TO COMPUTE FINAL MARKS
377     btn6.addActionListener(new ActionListener(){
:
387 txt6.setText(String.valueOf(cb19Int + cb20Int + cb21Int + cb22Int + cb23Int + cb24Int));
388       }
389     });
```

Line 330 until 389 is the code to compute all of the assessment marks according to the types of examinations.

```
391     //TO CALCULATE FINAL GRADE, DETERMINE THE GRADE POINT AND STATUS
392     jcompute.addActionListener(new ActionListener(){
:
408         totalgrade = (a + b + c + d)/4;
409         total = Double.toString(totalgrade);
410         finalgrade.setText(total);
411
412         JFrame f3 = new JFrame ("Student Final Grading");
:
452         //GRADE POINT, GRADE AND STATUS
453         grade = Double.parseDouble(finalgrade.getText());
454
455         if(grade<101 && grade>=80.00){
456            png = 4.00;
457            grading = "A";
458            status = "Pass with Distinction";
:
495         }else{
496            png = 0.00;
497            grading = "E";
498            status = "Fail";
499         }
:
507       }
508     });
```

Line 391 until 508 is the code to calculate the final grade and determine the grade point, grade and status. The user have to click 'Compute All' button for the final grade computation. To determine the grade point, grade and status, the program has to calculate the final grade first. As you can see in line 408 until 410, the final grade will be calculated by using the formula in line 408. Then, the final grade is set to the `finalgrade` variable in line 410. After that, the program will determine the grade point, grade and status in line 452 until 507. The `grade` variable will get the value of final grade from `finalgrade` variable. Then, by using if else statement in line 455 until 499 , the program will determine the grade point, grade and status.

The result of the final grade, grade point, grade and status will be shown on the next panel which is the Student Final Grading panel. Then, the user can click 'OK' button, 'Print' button or close the panel to end the program. If the user clicked the 'OK' button, a message dialog will appear showing the student's grade point. If the user clicked the 'OK' button on the message dialog, another message dialog appears showing the student status. If the user clicked the 'OK' button on the message dialog, it will go to the previous panel, which is the Student Final Grading panel. The code for the 'OK' button and message dialog shown below.

```
510      finalbutton.addActionListener(new ActionListener() {
511        @Override
512        public void actionPerformed(ActionEvent e) {
513          JOptionPane.showMessageDialog(null, "" + png);
514
515          if (png  >=1.50 && png <=4.0) JOptionPane.showMessageDialog(null, " STUDENT
PASSED ");
516          else if (png >=0.0 && png <=1.49) JOptionPane.showMessageDialog(null, "
STUDENT FAILED ");
517        }
518      });
```

Write Data

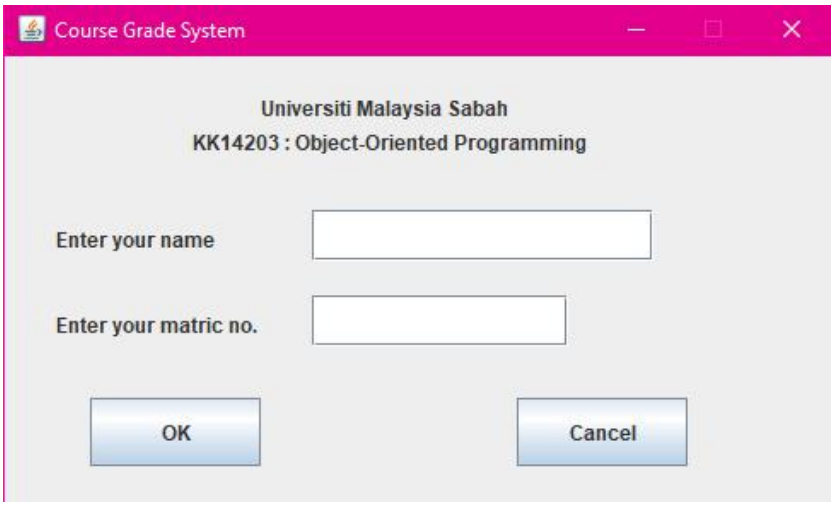
In the program, the user can save the data by clicking the 'Print' button on the Student Final Grading panel. The destination of the saved data had been set, as shown in the coding below.

```
520      finalprint.addActionListener(new ActionListener() {
521        public void actionPerformed(ActionEvent e) {
522        JOptionPane.showMessageDialog(rootPane, "File Saved as 'Course Grade.txt' in
Dekstop");
 :
531
532        output1 = "Name          : " + finalname + System.lineSeparator();
533        output2 = "Matric No.     : " + finalmatric + System.lineSeparator();
534        output3 = "Final Grade    : " + printfinal + System.lineSeparator();
535        output4 = "Grade Point    : " + printrating + System.lineSeparator();
536        output5 = "Grade         : " + printgrading + System.lineSeparator();
537        output6 = "Student Status : " + printstatus + System.lineSeparator();
538
539        if(e.getSource() == finalprint){
540          try {
541              PrintWriter out = new PrintWriter(new
FileWriter("C:\\Users\\USER\\Desktop\\Course Grade.txt"));
542
 :
558      });
```

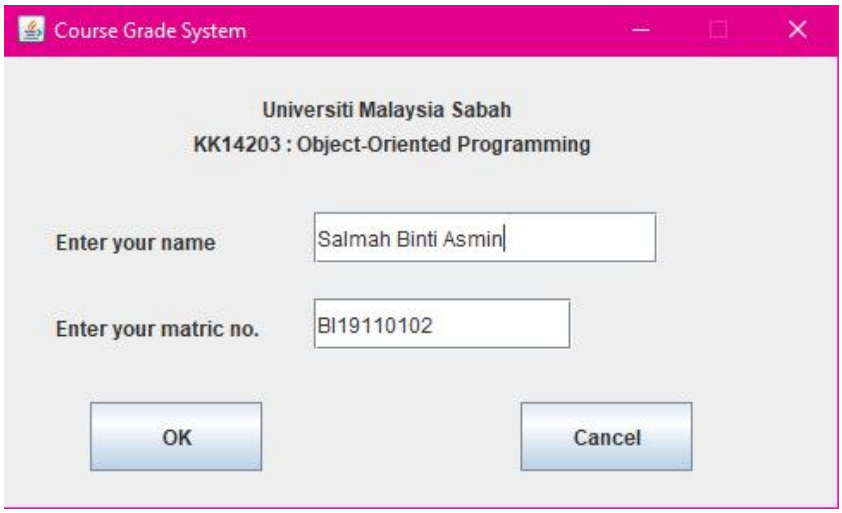
When the user clicked the 'Print' button on the Student Final Grading panel, a message dialog will appear stating that the data is saved as a text file at the set location, as shown in line 522. The data will be saved as shown in line 532 until 537, containing the name, matric no., final grade, grade point, grade and student status. The location of the saved data had been set, as shown in line 541. The user can find the text file named Course Grade.txt in Dekstop.

USER MANUAL

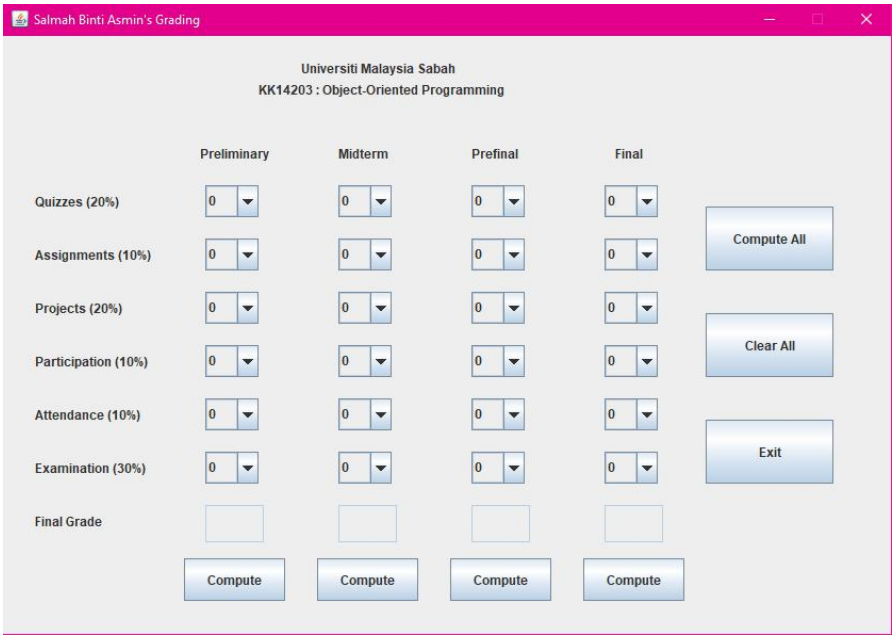
Step 1



The user have to enter the name and matric no. before clicking the 'OK' button. If the user did not fill the name, matric no. or both, a message dialog below will appear. The user can click the 'OK' button and enter the name and matric no. again.

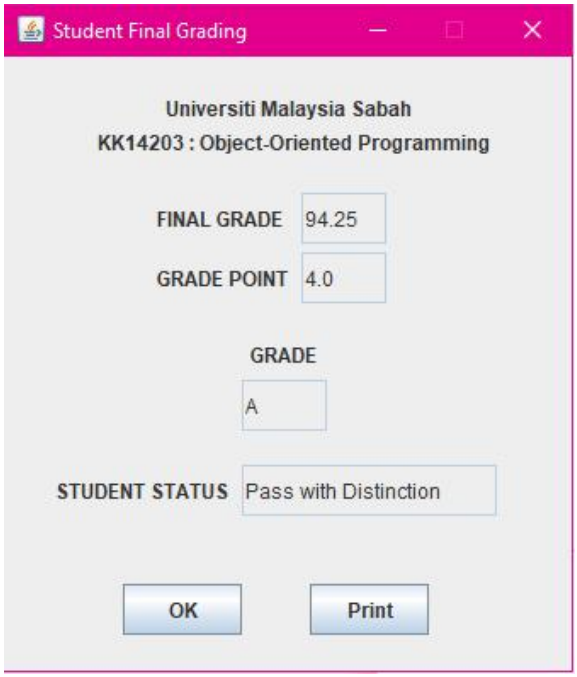


Step 2



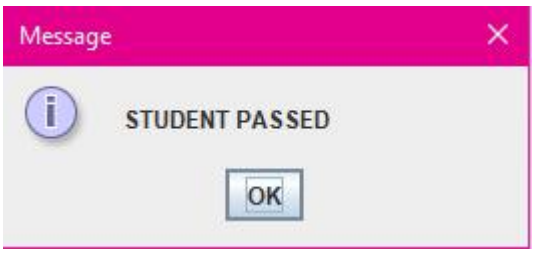
After clicking the 'OK' button, the user will go to the next panel which is shown above. The user have to choose the marks from the combo box provided according to the assessments (Quizzes, Assignments, Projects, Participation, Attendance and Examination) and examinations (Preliminary, Midterm, Prefinal and Final). After choosing the marks, user have to click the 'Compute' button for each examination before clicking the 'Compute All' button for the final grade computation.

Step 3



After the user clicked the 'Compute All' button, the user will go to the next panel which is shown above. The 'Student Final Grading' panel shows the calculated final grading of student which is the Final Grade, Grade Point, Grade and Student Status.

The 'OK' button is just for a message dialog that shows the grade point and status of the student. The message dialogs are shown below respectively :

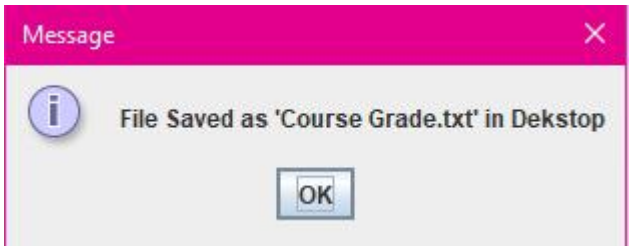


After clicking the 'OK' button for both message dialog, the user will go back to the previous panel, which is the 'Student Final Grading' panel.

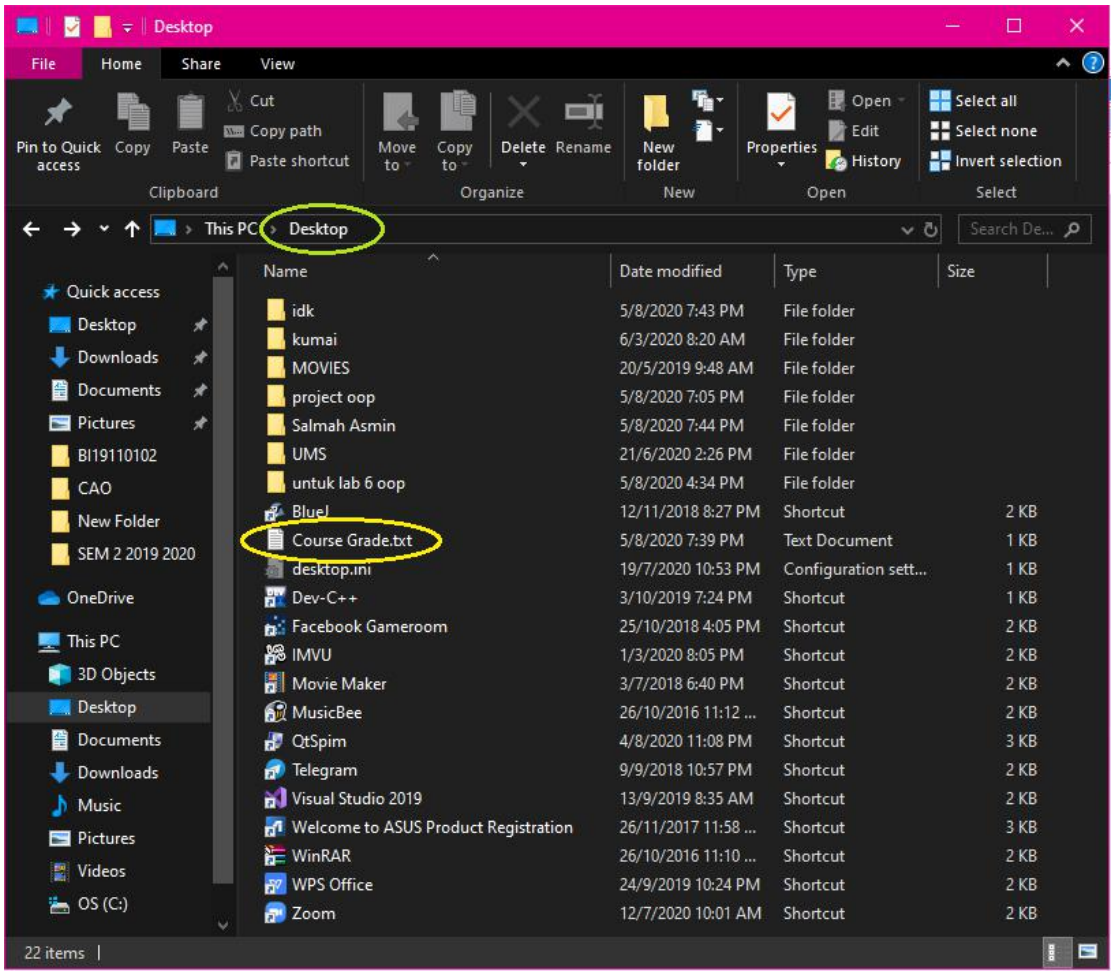
If the user wants to end the session, the user can simply clicked the Exit button at the upper right of the 'Student Final Grading' panel. If the user wants to save the data, the user can go to the next step.

Step 4

The ‘Print’ button on the ‘Student Final Grading’ panel is used to save the student’s data to a text file to the location that had been set by the program. After clicking the ‘Print’ button, a message dialog will appear telling the user that the data has been saved to a text file, as shown below.



The user can find the text file by referring to the message dialog and open it.



The figure below shows the data that has been saved to ‘Course Grade.txt’ file.