King Saud University
College of Computer & Information Science
CSC111 – Lab08
Objectss – II –
All Sections

Instructions

Web-CAT submission URL:
http://10.131.240.28:8080/Web-CAT/WebObjects/Web-CAT.woa/wa/assignments/eclipse

Objectives:

- To apply class abstraction to develop software.
- To design programs using the object-oriented paradigm.
- To use UML graphical notation to describe classes and objects.
- To demonstrate how to define classes and create objects.
- To create objects using constructors.
- To access objects via object reference variables.
- To define a reference variable using a reference type.
- To access an object's data and methods using the object member access operator (.).
- To define data fields of reference types and assign default values for an object's data fields.
- To distinguish between instance and static variables and methods.
- To learn how to use static constant data members.
- To define private data fields with appropriate getter and setter methods.
- To learn when and how to define private methods.
- To encapsulate data fields to make classes easy to maintain

- To develop methods with object arguments and differentiate between primitive-type arguments and object-type arguments
- To use the keyword ***this*** to refer to the calling object itself.
- To combine logic (conditionals and loops) with objects.
- To learn that most of program logic in OOP goes inside classes.
- To learn how to write private helper methods.
- To learn how a method of an object calls another method of the same object.

Lab Exercise 1

Design a class named `LinearEquation` for a 2 x 2 system of linear equations:

$$\boldsymbol{ax + by = e}$$

$$cx + dy = f$$

A system of linear equations can be solved using Cramer's rule as following:

$$x = \frac{ed - bf}{ad - bc} \quad y = \frac{af - ec}{ad - bc}$$

The class contains:

- Private data fields `a, b, c, d, e`, and `f`.
- A constructor with the arguments for `a, b, c, d, e`, and `f`.
- A method named `isSolvable()` that returns true if ad - bc is not 0.
- Methods `solveX()` and `solveY()` that return the solution for the equation.
- A method `isEquivalent(LinearEquation leq)` that returns true if linear system `leq` is equivalent to linear system represented by current object used to call the method. Two 2 x 2 linear systems are equivalent if they have the same solution for both x and y.

Draw the UML diagram for the class and then implement the class. Write a test program that prompts the user to enter `a, b, c, d, e`, and `f` and displays the solution. If `ad - bc` is `0`, report that `"The system has no solution."` Then if the first equation is solvable, it allows the user to enter another 2 x 2 system to check if it is equivalent to first one otherwise program exits (use `System.exit(0)` to exit the program).
(***Additional exercise to solve at home***:
Write setter and getter methods for all attributes in class `LinearEquation`.)

Sample Run 1

```
Enter a, b, c, d, e, f: 9 4 3 -5 -6 -21 ↵
x is -2.0 and y is 3.0
Do you want to check equivalence with another system (Yes/No)?
yes ↵
Enter a, b, c, d, e, f: 2 3 4 1 5 -5
The two systems have the same answers.
```

Sample Run 2

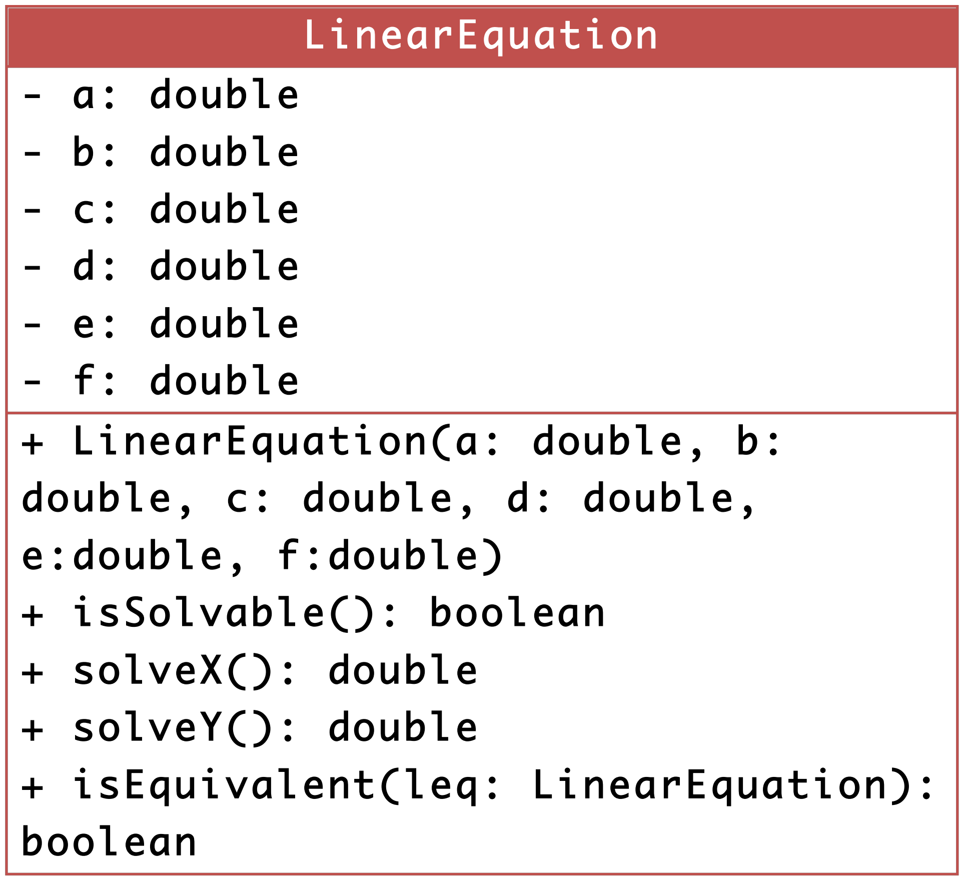
```
Enter a, b, c, d, e, f: 9 4 3 -5 -6 -21 ↵
x is -2.0 and y is 3.0
Do you want to check equivalence with another system (Yes/No)?
No↵
```

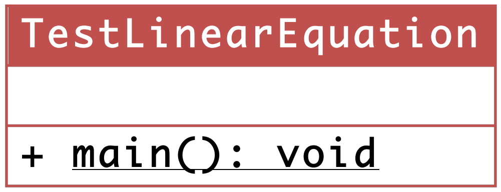
Sample Run 3

```
Enter a, b, c, d, e, f: 1 2 2 4 5 5 ↵
The system has no solution
```

Solution

1- First phase is to design your program as an OOP program. Draw UML diagrams for the two classes, `LinearEquation` and `TestLinearEquation`.





2- Create a new eclipse project and name it `lab08`

3- Create a new class and name it `TestLinearEquation`.

   Make sure you choose the `public static void main` option.

   We will write both classes into this file, which means

`LinearEquation` class, will not be public. Another option is to create two files one for each class.

4- Write the program as shown in next page (you can ignore comments)

5- When you are done, save your program and run it. Make sure it prints the output as shown above.
6- Submit your program to WebCAT through. Ask your TA for help.

```java
class LinearEquation {
  private double a;
  private double b;
  private double c;
  private double d;
  private double e;
  private double f;

  public LinearEquation(double a, double b, double c,
      double d, double e, double f) {
    this.a = a;
    this.b = b;
    this.c = c;
    this.d = d;
    this.e = e;
    this.f = f;
  }

  public boolean isSolvable() {
    return a * d - b * c != 0;
  }

  public double solveX() {
    double x = (e * d - b * f) / (a * d - b * c);
    return x;
  }

  public double solveY() {
    double y = (a * f - e * c) / (a * d - b * c);
    return y;
  }

  public boolean isEquivalent(LinearEquation leq){
      return this.solveX() == leq.solveX() && this.solveY() == leq.solveY();
  }
}
```

```
import java.util.Scanner;

public class TestLinearEquation {
  public static void main(String[] args) {
    Scanner input = new Scanner(System.in);
    System.out.print("Enter a, b, c, d, e, f: ");
    double a = input.nextDouble();
    double b = input.nextDouble();
    double c = input.nextDouble();
    double d = input.nextDouble();
    double e = input.nextDouble();
    double f = input.nextDouble();

    LinearEquation equation = new LinearEquation(a, b, c, d, e, f);

    if (equation.isSolvable()) {
      System.out.println("x is " +
        equation.solveX() + " and y is " + equation.solveY());
    }
    else {
      System.out.println("The system has no solution");
      System.exit(0);
    }

    System.out.print("Do you want to check equivalence with another system (Yes/No)? ");
    String answer = input.next();
    if (answer.equalsIgnoreCase("Yes")){
        System.out.print("Enter a, b, c, d, e, f: ");
        a = input.nextDouble();
        b = input.nextDouble();
        c = input.nextDouble();
        d = input.nextDouble();
        e = input.nextDouble();
        f = input.nextDouble();

        LinearEquation equation2 = new LinearEquation(a, b, c, d, e, f);
        if (equation.isEquivalent(equation2))
            System.out.println("The two systems have the same answers.");
        else
            System.out.println("The two systems have different answers.");
    }
  }
}
```

Lab Exercise 2

You have been asked by Saudi Wildlife Authority to design a class named `Species` to manage endangered species. Class details are as following:

- Three data fields (properties) which are:
  - `name` of type `String` which stores name of species,
  - `population` of type `int` which stores the current population of the species and it can not be negative, and
  - `growthRate` of type `double` which stores the growth rate of the current population.
- A method `readInput()` that reads the values of data members. The method must not allow the user to enter incorrect population value. If this happens, it should keep asking for correct value until the user enters it.
- A method `writeOutput()` that prints the data of the species.
- A method `predictPopulation(int years)` that returns the projected population of the species after the specified number of years (which must be a non-negative number).
- A method `setSpecies(String newName, int newPopulation, double newGrowthRate)` that sets values of receiving object to new data sent through parameters. The method should print an error message and exit the whole program if `newPopulation` is a negative number.
- Getter methods for `name, population` and `growthRate`.

- A method `equals(Species otherObject)` that compares this object to `otherObject`. Two species objects are equal if they have the same name ignoring the letter case.
- A method `isPopulationLargerThan(Species otherSpecies)` that returns true if population of this object is greater than the population of `otherSpecies`.
- A method `isExtinct()` that returns true if the population of this object is 0, otherwise returns false.

Draw the UML diagram for the class and then implement the class. Write a test program that checks if the population of an input species chosen by the user could one day exceed that of Arabian Oryx.

- The program first reads the information of some species X.
- Then it checks if species X is extinct and if so it prints the message ”`The species that you entered is extinct`” and exits.
- If species X is not extinct, then it checks if, given the current population and growth rate of species X, its population will surpass that of Arabian Oryx, given that the current population of Arabian Oryx is 1000 and its growth rate is 0.25. If this could happen then it prints after how many years this will happen.
- If user enters Arabian Oryx then you should keep asking him to enter a different species.

Name your classes `Species` and `TestSpecies`. Use two separate files for each of the two classes.

Sample Run 1

```
What is the species' name?
Dinosaur ↲
What is the population of the species?
0 ↲
Enter growth rate (% increase per year):
0 ↲
The species that you entered is extinct.
```

Sample Run 2

```
What is the species' name?
Houbara Bustard ↲
What is the population of the species?
900 ↲
Enter growth rate (% increase per year):
2 ↲
Species Houbara Bustard has slower growth rate than Arabian
Oryx.
```

Sample Run 3

```
What is the species' name?
African Lion ↲
What is the population of the species?
16500 ↲
Enter growth rate (% increase per year):
5 ↲
Population of species Lion is already larger than
population of Arabian Oryx.
```

Sample Run 4

```
What is the species' name?
Arabian Oryx ↲
What is the population of the species?
1000 ↲
Enter growth rate (% increase per year):
25 ↲
You entered the Arabian Oryx species, please enter another
one:
What is the species' name?
Arabian Oryx ↲
What is the population of the species?
1000 ↲
Enter growth rate (% increase per year):
25 ↲
You entered the Arabian Oryx species, please enter another
one:
What is the species' name?
Blue Whale ↲
What is the population of the species?
500 ↲
Enter growth rate (% increase per year):
38 ↲
After 8 years, population of Blue Whale will surpass that
of Arabian Oryx.
```

Solution

1- First phase is to design your program as an OOP program. Draw UML diagrams for the two classes, `Species` and `TestSpecies`.

Species
- name: String - population: int - growthRate: double
+ readInput(): void + wrtieOutput(): void + predictPopulation(years: int): int + setSpecies(newName: String, newPopulation: int, newGrowthRate: double): void + getName(): String + getPopulation(): int + getGrowthRate(): double + equals(otherObject: Species): boolean + isPopulationLargerThan(otherSpecies: Species): boolean + isExtinct(): boolean

TestSpecies
+ main(): void

2- Use previously created project `lab08`

3- Unlike in previous exercise, we will create two separate files for the two classes. Create a new class and name it `Species`.

4- Create a new class and name it `TestSpecies`. Make sure you choose the `public static void main` option.
5- Write the two program classes as shown in next pages (you can ignore comments)
6- When you are done, save your program and run it. Make sure it prints the output as shown above.
7- Submit your program to WebCAT through. Ask your TA for help.

```java
import java.util.Scanner;

public class Species {
    private String name;
    private int population;
    private double growthRate;

    public void readInput() {
        Scanner keyboard = new Scanner(System.in);
        System.out.println("What is the species' name?");
        name = keyboard.nextLine();
        System.out.println("What is the population of the species?");
        population = keyboard.nextInt();
        while (population < 0) {
            System.out.println("Population cannot be negative.");
            System.out.println("Reenter population:");
            population = keyboard.nextInt();
        }
        System.out.println("Enter growth rate (% increase per year):");
        growthRate = keyboard.nextDouble();
    }

    public void writeOutput() {
        System.out.println("Name = " + name);
        System.out.println("Population = " + population);
        System.out.println("Growth rate = " + growthRate + "%");
    }

    /**
     * Precondition: years is a nonnegative number. Returns the projected
     * population of the receiving object after the specified number of years.
     */
    public int predictPopulation(int years) {
        int result = 0;
        double populationAmount = population;
        int count = years;
        while ((count > 0) && (populationAmount > 0)) {
            populationAmount = (populationAmount + (growthRate / 100)
                    * populationAmount);
            count--;
        }
        if (populationAmount > 0)
            result = (int) populationAmount;
        return result;
    }
```

Continue code of this class on next page

```
    public void setSpecies(String newName, int newPopulation,
            double newGrowthRate) {
        name = newName;
        if (newPopulation >= 0)
            population = newPopulation;
        else {
            System.out.println("ERROR: using a negative " + "population.");
            System.exit(0);
        }
        growthRate = newGrowthRate;
    }

    public String getName() {
        return name;
    }

    public int getPopulation() {
        return population;
    }

    public double getGrowthRate() {
        return growthRate;
    }

    public boolean equals(Species otherObject) {
        return (name.equalsIgnoreCase(otherObject.name));
    }

    /**
     * Precondition: This object and the argument otherSpecies both have values
     * for their population. Returns true if the population of this object is
     * greater than the population of otherSpecies; otherwise, returns false.
     */
    public boolean isPopulationLargerThan(Species otherSpecies) {
        return population > otherSpecies.population;
    }

    /**
     * Precondition: This object has a value for its population. Returns true if
     * the population of this object is zero; otherwise, returns false.
     */
    public Boolean isExtinct() {
        return population == 0;
    }

}
```

Class with main is on next page

```java
public class TestSpecies {

    public static void main(String[] args) {
        Species speciesX = new Species();
        speciesX.readInput();
        if (speciesX.isExtinct()){
            System.out.println("The species that you entered is extinct.");
            System.exit(0);
        }
        Species arabianOryx = new Species();
        arabianOryx.setSpecies("Arabian Oryx", 1000, 25);
        while (speciesX.equals(arabianOryx)){
            System.out.println("You entered the Arabian Oryx species, " +
                    "please enter another one:");
            speciesX.readInput();
        }
        if (speciesX.isPopulationLargerThan(arabianOryx))
            System.out.println("Population of species " + speciesX.getName() +
                    " is already larger than population of Arabian Oryx.");
        else {
            if (speciesX.getGrowthRate() <= arabianOryx.getGrowthRate())
                System.out.println("Species " + speciesX.getName() +
                        " has slower growth rate than Arabian Oryx.");
            else {
                int years = 1;
                while (speciesX.predictPopulation(years) < arabianOryx.predictPopulation(years))
                    years++;
                System.out.println("After " + years + " years, population of "
                    + speciesX.getName() + " will surpass" +
                    " that of Arabian Oryx.");
            }
        }
```

Lab Exercise 3

(This exercise is extra and student should complete it at home)

We would like you to program a simple game called Oracle. The oracle pretends that it knows the answer to any question. It starts by letting the user ask a question. Then, it asks the user for an advice on how to answer his question. Finally, it gives the user an answer to his question based on the advice he gave to previous question. Let me give you a sample run and then explain the program in detail.

Sample Run

```
I am the oracle. I will answer any one-line question.
What is your question?
What time is it? ↵
Hmm, I need some help on that.
Please give me one line of advice.
Seek and you shall find the answer ↵
Thank you. That helped a lot.
You asked the question:
 What time is it?
Now, here is my answer:
 The answer is in your heart.
Do you wish to ask another question?
yes ↵
What is your question?
Am I gonna pass this course :(? ↵
Hmm, I need some help on that.
Please give me one line of advice.
Ask the professor ↵
Thank you. That helped a lot.
You asked the question:
 Am I gonna pass this course :(?
Now, here is my answer:
 Seek and you shall find the answer
Do you wish to ask another question?
No ↵
The oracle will now rest.
```

Design a class called `Oracle` that has the following members:

- Three instance data fields (we will see how to use these when we explain the method members):
  - `oldAnswer` of type `String` which keeps the old answer given by the user.
  - `newAnswer` of type `String` where we store new answer given by the user.
  - `question` of type `String` where we store the question given by the user
- Four static data fields:
  - `WELCOME_MSG` of type `String` that stores the intro string given to user. The message is `“I am the oracle. I will answer any one-line question.”`
  - `ADVICE_SEEKING_MSG` of type `String`, which stores the advice-seeking message given to user after each question, he asks. The message is `“Hmm, I need some help on that.\nPlease give me one line of advice.”`.
  - `THANKS_YOU_MSG` of type `String` that stores the thank you message given to the user after he gives his advice. The value is `“Thank you. That helped a lot.”`.
  - `GOODBYE_MSG` of type `String` that stores the goodbye message given to the user at the end of the program. The value is `“The oracle will now rest.”`.
- A method `chat()` which is the main method that runs the game logic. In this method, we start by giving the welcome message

then we repeat the following tasks: (1) ask the user for his new question, (2) do the game trick to answer the question and then (3) check if the user wants to ask another question (i.e., repeat the game) quit. To ask, process and answer a new question (i.e., (1) and (2)), Method `chat()` delegates this task to method `answer()`. Finally, the method prints the goodbye message.

- A method `answer()` that receives the user's question and tries to answer it. Method `answer()` first asks the user to enter his question then calls method `seekAdvice()` to get an advice from the user on how to answer the question. After that it uses the old advice (i.e., old answer) given by the user to previous question to answer the current question. At the first run, old answer is initialized to the string `"The answer is in your heart."`. After that it calls the method `update()` to update the old answer with the advice given by the user when method `seekAdvice()` was called.
- A method `seekAdvice()` that asks the user for an advice on how to answer his question. It then stores this advice as the new answer and prints the thank you message. This advice/answer will be used later on to answer the user's next question. You notice that this will achieve the game trick which is to answer the user's question using his previous answer.
- A method `update()` which stores the advice given by the user as the old answer to be used later on to answer the next question.

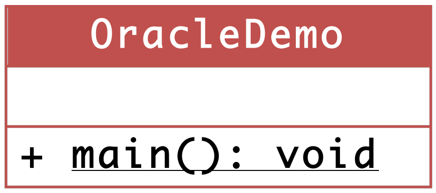
Write a program called `OracleDemo` that runs the Oracle game by creating an object of type `Oracle` and then calling the `chat()` method.

Notice that the program does not need to call any of the methods except method `chat()`. This means that, except for method `chat()`, all other methods should be `private` since they are called internally only. This is a clear example that shows how, in Object Oriented Programming, most of the program logic is implemented in the classes not the main method.

Solution

1- First phase is to design your program as an OOP program. Draw UML diagrams for the two classes, `Oracle` and `OracleDemo`.

Oracle
- oldAnswer: String - newAnswer: String - question: String - WELCOME_MSG: String - ADVICE_SEEKING_MSG: String - THANK_YOU_MSG: String - GOODBYE_MSG: String
+ chat(): void - answer(): void - seekAdvice(): void - update(): void



2- Use previously created eclipse project `lab08`

3- Similar to previous exercise, we will create two separate files for the two classes. Create a new class and name it `Oracle`.

4- Create a new class and name it `OracleDemo`. Make sure you choose the `public static void main` option.

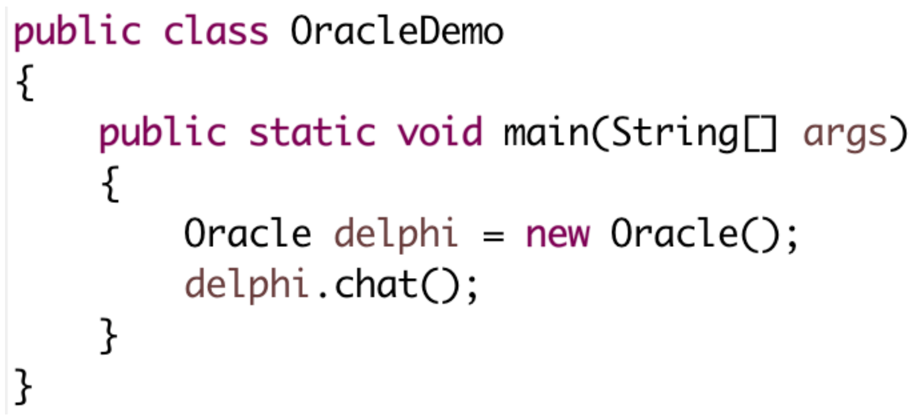
5- Write the two program classes as shown in next pages (you can ignore comments)

6- When you are done, save your program and run it. Make sure it prints the output as shown above.

7- Submit your program to WebCAT through. Ask your TA for help.

```java
import java.util.Scanner;

public class Oracle {
    private String oldAnswer = "The answer is in your heart.";
    private String newAnswer;
    private String question;
    private static final String WELCOME_MSG = "I am the oracle." +
            "I will answer any one-line question.";
    private static final String ADVICE_SEEKING_MSG  = "Hmm, I need" +
            "some help on that.\nPlease give me one line of advice.";
    private static final String THANKS_YOU_MSG  = "Thank you. "+
            "That helped a lot.";
    private static final String GOODBYE_MSG  = "The oracle will "
            + "now rest.";

    public void chat() {
        System.out.println(WELCOME_MSG);
        Scanner keyboard = new Scanner(System.in);
        String response;
        do {
            answer();
            System.out.println("Do you wish to ask " + "another question?");
            response = keyboard.next();
        } while (response.equalsIgnoreCase("yes"));
        System.out.println(GOODBYE_MSG);
    }

    private void answer() {
        System.out.println("What is your question?");
        Scanner keyboard = new Scanner(System.in);
        question = keyboard.nextLine();
        seekAdvice();
        System.out.println("You asked the question:");
        System.out.println("  " + question);
        System.out.println("Now, here is my answer:");
        System.out.println("  " + oldAnswer);
        update();
    }

    private void seekAdvice() {
        System.out.println(ADVICE_SEEKING_MSG);
        Scanner keyboard = new Scanner(System.in);
        newAnswer = keyboard.nextLine();
        System.out.println(THANKS_YOU_MSG);
    }

    private void update() {
        oldAnswer = newAnswer;
    }
}
```

Class with main is on next page



```java
public class OracleDemo
{
    public static void main(String[] args)
    {
        Oracle delphi = new Oracle();
        delphi.chat();
    }
}
```

Done…