CS 180 Problem Solving and OO Programming
Fall 2011
Recitation
Week 2 [August 29---Sept2]
Solutions: BY Anmer Daskin

Problem-1

The execution of a Java programs begins in the main() method. Even though a Java compiler will compile all classes without any main() method, the Java interpreter (JVM) requires a main() method to initiate program execution. So when we see an application we should start tracing program from the main method, and trace the flow of the program. Note that, Java is case sensitive: Main() is different from main().

The text between "/*" and "*/" is treated as comment.

```
/*
 * This program computes the distance travelled by
 * a moving object given its speed and travel duration.
 * Uses the Scanner class for input.
 * Author: Aditya Mathur. Date: July 20, 2010.
 */
```

Java predefined library classes: import all classes from the packages java.util and javax.swing, and import the class String from the package java.lang

```
import java.util.*;
import java.lang.String;
import javax.swing.*;
```

The following line initiates the class definition for the class ScannerExample

```
public class ScannerExample {
```

The text coming after "//" defines the comments at a line.

```
    // Create an object named source that will
    // help us input data for solving this problem.
```

The object source is created from the class Scanner. source has the type Scanner. System.in is given as an argument to the Scanner's constructor: System.in is input stream and supplies input data. The object source will use System.in to get input data.

```
    Scanner source = new Scanner(System.in); /*@\label{scannerObject}@*/
```

Constructors are special methods having the same name as their classes, and they are invoked to create objects from the class blueprint. The following constructor will be invoked, when one creates an object from the class ScannerExample. See that the constroctur does not have return type and has the same name as the class.

```
        public ScannerExample() {
```

Variables are used to do the the required computation. While speed, duration, and distance have the primitive type double, enters and enterD are objects whose types are String which is a class.

```
        double speed, duration;// Speed and duration.
/*@\label{useOfDouble}@*/
        double distance; // Distance to be computed.
```

```
        // Declare user prompt strings.
        String enterS = "Enter the speed.";
        String enterD = "Enter duration.";
// Solution to sub-problem 1: Read input.
        // Prompt the user and get speed and duration.
        System.out.println(enterS);
/*@\label{solutionSubProblemOneScanner}@*/
        speed = source.nextDouble();/*@\label{getSpeed}@*/
        System.out.println(enterD);
        duration = source.nextDouble();/*@\label{getDuration}@*/
// Solution to sub-problem 2: Compute distance travelled.
        distance = speed*duration;/*@\label{solutionSubProblemTwoScanner}@*/
// Solution to sub-problem 3: Display output.
        System.out.println("Distance travelled: "+ distance+"
miles.");/*@\label{distanceTravelled}@*/
    }
```

The public keyword is an access specifier that allows us to control the visibility of class members. The keyword static allows main( ) to be called without creating a particular instance of the class. (String[] args) indicates that our program can accept arguments at run time.

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

"new" returns an object reference which refers to the class ScannerExample. Since ScannerExample() constructor is invoked by this expression, the code inside the constructor is executed next.

```
        new ScannerExample(); // Create ScannerExample object
    }
}
```

Problem-2

```
import java.util.Scanner;
/**
 *
 * @author adaskin
 */
public class ChemicalElement {
   String name;
   String symbol;
   double atomic_mass;
   public ChemicalElement(String nm, String sym,double am){
      name=nm;
      symbol=sym;
      atomic_mass=am;
   }
   public String getName(){
      return name;
   }
   public String getSymbol(){
```

```
    return symbol;
  }
  public double getAtomicMass(){
    return atomic_mass;
  }
  public static void main(String [] args){
    /*Temporary variables to store the inputs from the user*/
    String name;
    String symbol;
    double atomic_mass;

    /*To get input from the console*/
    Scanner sc=new Scanner(System.in);

    /*Prompt user to input name, atomic mass, and symbol of the first element*/
    System.out.printf("Enter the first element's name\n");
    name=sc.nextLine();
    System.out.printf("Enter the first element's symbol\n");
    symbol=sc.nextLine();
    System.out.printf("Enter the first element's atomic mass\n");
    atomic_mass=sc.nextDouble();
    ChemicalElement e1=new ChemicalElement (name, symbol, atomic_mass);

   /*Prompt user to input name, atomic mass, and symbol of the second element*/
    System.out.printf("Enter the second element's name\n");
    name=sc.nextLine();
    System.out.printf("Enter the second element's symbol\n");
    symbol=sc.nextLine();
    System.out.printf("Enter the second element's atomic mass\n");
    atomic_mass=sc.nextDouble();
    ChemicalElement e2=new ChemicalElement (name, symbol, atomic_mass);

    /*extract the attributes*/
    System.out.printf("Element-1\n\t Name:%s\n\t Symbol:%s\n\t Atomic Mass:%s\n",
        e1.getName(),e1.getSymbol(),e1.getAtomicMass());
    System.out.printf("Element-2\n\t Name:%s\n\t Symbol:%s\n\t Atomic Mass:%s\n",
        e2.getName(),e2.getSymbol(),e2.getAtomicMass());
  }
}
```