Chapter 7 - Inheritance

Chapter Objectives

- Derive new classes from existing ones.
- Explain how inheritance supports software reuse.
- Add and modify methods in child classes.
- Discuss how to design class hierarchies.
- Define polymorphism and how it can be done.

Chapter Overview: Inheritance is a very important concept in object oriented design and program writing. This chapter lays a solid foundation for you to understand the concepts behind inherited classes and interfaces. Polymorphism and inheritance are major concepts in this chapter.

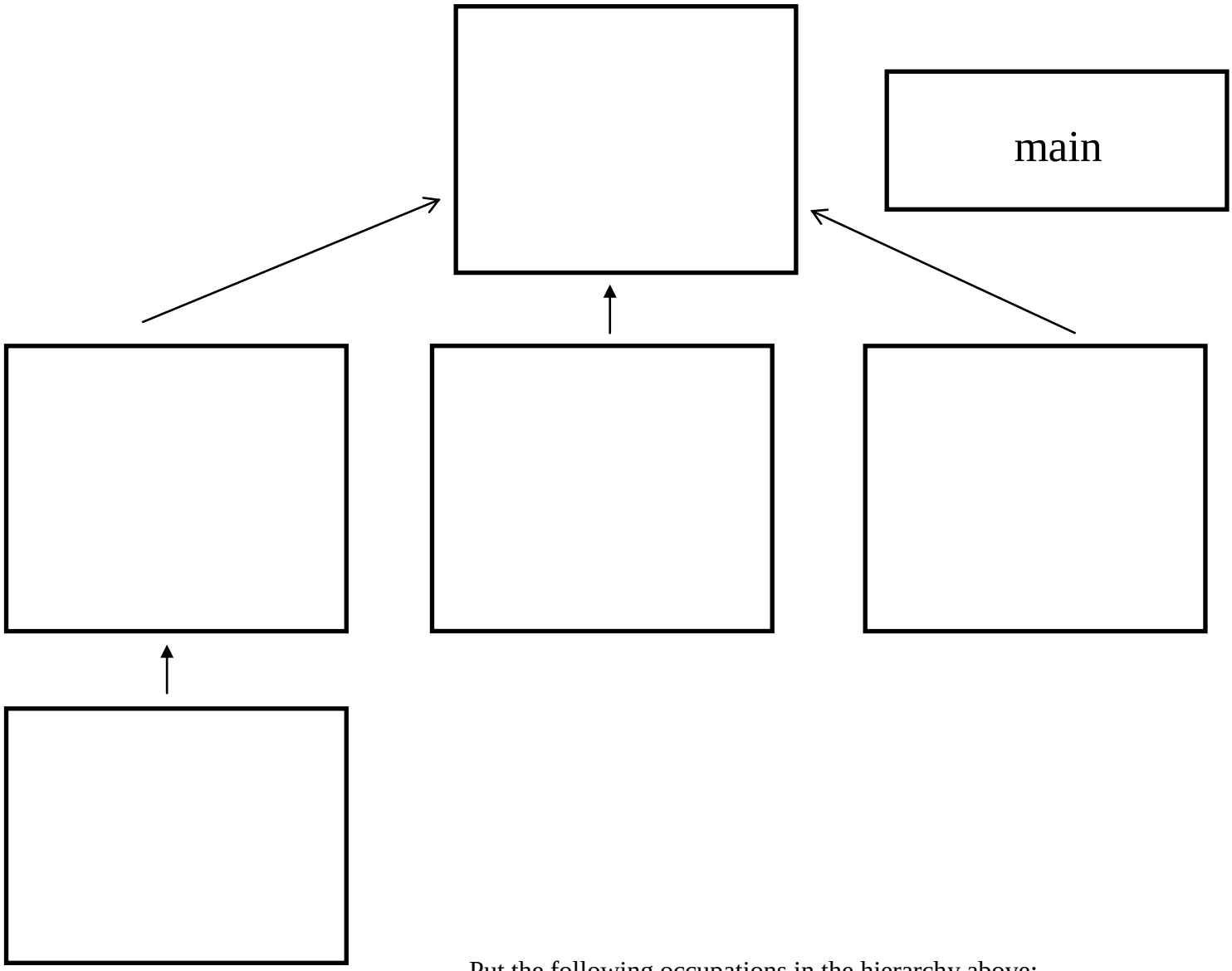
Multiple Choice: 7.1 – 7.10

True/False: 7.1 – 7.10

Short Answer: 7.3

AP Multiple Choice: 7.1 – 7.6

Programming Projects: 7.2, 7.2 again



Put the following occupations in the hierarchy above:
Hospital Employee, Doctor, Nurse, Custodian, Surgeon.
(Leave space to put the method names).

Put the following methods in the appropriate class: toString, getNumPatients, setNumPatients setLocation, getLocation, setIsRegistered, getIsRegistered, setName, getName, getID, setID, setType, getType, setAge, getAge, setArea, getArea

Project 7.2

Inheritance

```
public class Words
{

   public static void main (String[] args)
   {
      Dictionary webster = new Dictionary ();

      webster.pageMessage();
      webster.definitionMessage();
   }
}
```

```
public class Book
{
   public int pages = 1500;

   public void pageMessage ()
   {
      System.out.println ("Number of pages: " + pages);
   }
}
```

```
public class Dictionary extends Book
{
   private int definitions = 52500;

   public void definitionMessage ()
   {
      System.out.println ("Number of definitions: " + definitions);

      System.out.println ("Definitions per page: " + definitions/pages);
   }
}
```



```
public class Words2
{
   public static void main (String[] args)
   {
     Dictionary2 webster = new Dictionary2 (1500, 52500);

      webster.pageMessage();
      webster.definitionMessage();
   }
}
```

```
public class Book2
{
   public int pages;

   public Book2 (int numPages)
   {
      pages = numPages;
   }

   public void pageMessage ()
   {
      System.out.println ("Number of pages: " + pages);
   }
}
```

```
public class Dictionary2 extends Book2
{
   private int definitions;

   public Dictionary2 (int numPages, int numDefinitions)
   {
      super (numPages);
      definitions = numDefinitions;
   }

   public void definitionMessage ()
   {
      System.out.println ("Number of definitions: " + definitions);

      System.out.println ("Definitions per page: " + definitions/pages);
   }
}
```



```
public class Messages
{
   public static void main (String[] args)
   {
      Thought parked = new Thought();
      Advice dates = new Advice();

      parked.message();
      dates.message();  // overridden
   }
}
```

```
public class Thought
{

   public void message()
   {
      System.out.println ("I feel like I'm diagonally parked in a " +
                          "parallel universe.");

      System.out.println();
   }
}
```

```
public class Advice extends Thought
{

   public void message()
   {
      System.out.println ("Warning: Dates in calendar are closer " +
                          "than they appear.");

      System.out.println();

      super.message();
   }
}
```



```
public class Academia
{
   public static void main (String[] args)
   {
      Student Frank = new Student ("Frank", 5);
      StudentAthlete Suki = new StudentAthlete ("Suki", 4, "Soccer");

      System.out.println (Frank);
      System.out.println ();
      System.out.println (Suki);
      System.out.println ();

      if (! Frank.equals(Suki))
         System.out.println ("These are two different students.");
   }
}


public class Student
{
   private String name;
   private int numCourses;

   public Student (String studentName, int courses)
   {
      name = studentName;
      numCourses = courses;
   }

   public String toString()
   {
      String result = "Student name: " + name + "\n";
      result += "Number of courses: " + numCourses;
      return result;
   }
}

public class StudentAthlete extends Student
{
   private String sport;

   public StudentAthlete (String studentName, int courses,
                          String sportName)
   {
     super (studentName, courses);
     sport = sportName;
   }

   public String toString()
   {
      String result = super.toString();
      result += "\nSport: " + sport;
      return result;
   }
}
```

