Philadelphia University
Lecturer: Dr. Nadia Y. Yousif
Coordinator: Dr. Nadia Y. Yousif
Internal Examiner: Dr. Murad Maouch

Faculty of Information Technology
Software Engineering Department

Examination Paper

Object-Oriented Paradigms (721112) - Section: 3 **First Exam**
Date: November 28, 2006 **First Semester - 2006/2007** **Time: 50 Minutes**

Information for Candidates

*1. This examination paper contains **3** questions totalling **15** marks*
*2. The marks for parts of questions are shown in square brackets: e.g. **[2 marks]**.*

Advice to Candidates

1. You should attempt ALL questions. You should write your answers clearly.

I. Basic Notions

Objectives: *The aim of the question in this part is to evaluate your required minimal knowledge and understanding skills in the fundamentals of object oriented programming.*

Question 1 (5 marks)

This question contains 5 multiple choice questions with each question worth 1 mark. You should write the letter A, B, C, or D in your answer sheet to indicate your correct answer.

1) One of the following is NOT TRUE about Object-Oriented Paradigms (OOPs):

A) OOP is a set of techniques and processes focusing on how to analyse and model a real world problem and to design a solution.
B) The intended benefits of OOP are to solve the "software" crisis, break complexity into small manageable chunks, and make software maintenance easier.
C) OOP allows reuse of components – plug and play
D) OOP solves the entire problem in one program.

2) Which point is FALSE from the following?

A) A class is an object
B) A class is a template or prototype that defines the composition and the behavior of all objects of certain kinds.
C) A class may have fields of composite types
D) From a class you may initiate an object.

3) Methods of a class are invoked from outside the class by

A) objects using the dot notation. e.g. circle_1.moveHorizontal (50)
B) using its name only e.g. moveHorizontal (50)
C) using the call statement e.g. CALL moveHorizontal (50)
D) A and B above

4) A method in a class that is used to change the values of some fields in that class is called:

A) A constructor
B) An accessor method
C) A mutator method
D) None of the above

5) If there are one or more constructors for a class then

A) Exactly one of the constructors will be called each time an object of that class is created
B) All of the constructors will be called each time an object of that class is created
C) A destructor must also be written.
D) None of the above, classes cannot have constructors

II. Familiar Problem Solving

Objectives: *The aim of the questions in this part is to evaluate that you have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP.*

Question 2 (5 marks)

Consider the following class definition that is written without methods. Instead, it has comments followed by blank areas denoted by …… to describe what each method has to do. Add method definitions in those blank areas as indicated in the comment. You will get **one mark** for each method definition.

```
/**
 * The VideoTape class holds information about a single television programme recorded on a video tape
 * and it is used in a video shop system. It holds the video tape details.
 */
public class VideoTape
{ private String title;              // the title of the programme
  private String classification;    // classification of the programme (comedy, drama, action, or romance)
  private int time;                 // the running time of the programme in minutes

   // Create a new video tape with a given title, classification, and time.

   public VideoTape (String fullTitle, String programClassification, int runningTime )
   {  title = fullTitle;
      classification = programClassification;
      time = runningTime;
   }

   // Return the title of this video tape.
        …….

  // Return the classification of this video tape.
        …….

  // Return the time of this video tape as a string in the following format:    2:06.
        …….

 // Set a new classification for this video tape.
        …….

 /* Print the details of the video tape to the output terminal in the following format:
  *  Adil Emam (COMEDY) 2:16
  */
        ……..
```

Question 3 (5 marks)

Design a class called **ISBN** to represent an International Standard Book Number, or ISBN for short.
The ISBN consists of 10 digits divided into 4 parts. For example, the ISBN 0 941831 39 6 represents the following information:

The first part: The first digit "0" signifies the book is from an English speaking country.
The second part: "941831" identifies the publisher.
The third part: "39" is the title number for the book.
The fourth part: "6" is a check digit to indicate that the sum of the ISBN digits is 10.

The class should have a constructor and methods to set and get the ISBN as a string.

Design a **Book** class that represents relevant information about a book, including the book's title, author, publisher, city and date of publication, and price. The class should also include the field **ISBN** isbnNum; where **ISBN** is the class defined above.

This class should include a **constructor** and the following methods:

- **setBookISBN**: to set the ISBN for the book.
- **getAuthor**: to return the author of the book.
- **getBookISBN**: to get the ISBN of the book.
- **printDetails**: to print the information of a book in the following form:

Book Title: Object First with Java
Book Author: David j. Barnes and Michael Kolling
Publisher: Prentice Hall
ISBN: 0 941831 39 6

GOOD LUCK!

Philadelphia University
Lecturer: Dr. Nadia Y. Yousif
Coordinator: Dr. Issa Shihabat
Internal Examiner: Dr. Amer Abu Ali

Faculty of Information Technology
Software Engineering Dept.

Examination Paper

……

Object-Oriented Paradigms (721112) - Section: 3 **First Exam**
Date: April 6, 2006 **Second Semester - 2005/2006** **Time: 50 Minutes**

……

Information for Candidates

1. *This examination paper contains **3** questions totaling **15** marks*
2. *The marks for parts of questions are shown in square brackets: e.g. **[2 marks]**.*

Advice to Candidates

1. You should attempt ALL questions. You should write your answers clearly.

……

I. Basic Notions

***Objectives:** The aim of the question in this part is to evaluate your required minimal knowledge and understanding skills in the fundamentals of object oriented programming.*

Question 1 (5 marks)

(a) Explain each of the following: **[2 marks]**

1) Abstraction
2) The state of an object
3) The method signature
4) Modularization

(b) Fill the blank in each of the following: **[3 marks]**

1) …..…….. methods return information about the state of an object.
2) ……. methods change the values of some fields.
3) Methods or fields that CANNOT be accessed directly from outside an object are declared ……
4) Variables of class type store ……. to objects.
5) An …….. is a type of flexible size collection that stores objects.
6) An object is created by …… statement.

……..……..……

II. Familiar Problem Solving

***Objectives**: The aim of the questions in this part is to evaluate that you have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP.*

Question 2 (6 marks)

Consider the following class definition that has comments. Add method definitions in the blank areas denoted by …… for each method as the comment that precedes it indicates.

```
/**
 * The Student class represents a student in a student administration system.
 * It holds the student details.
 */
public class Student
{  private String name;          // the student's full name
   private String id;            // the student ID
   private int credits;          // the amount of credits for study taken so far

   // Create a new student with a given name and ID number.

   public Student (String fullName, String studentID)
   {  name = fullName;
      id = studentID;
      credits = 0;
   }

   // Return the full name of this student.
       ……

   // Set a new name for this student.
       ……

   // Return the student ID of this student.
       ……

   // Add some credit points to the student's accumulated credits.
       ……

   /* Check the number of credit points this student has accumulated.
    * If it is less than 132, then print "Not yet grdauated", otherwise print "Graduated".
    */
       ……

   // Print the student's name and ID number to the output terminal.
       ……..
```

Question 3 (4 marks)

Design a class called **Meeting** to represent meetings in a diary. The **Meeting** class has the following fields:

- **time** of the meeting represented as string in hours and minutes,
- **location** of the meeting (such as "room 205"),
- **subject** to represent the meeting's subject (such as "Examiner's meeting").

Time, location and **subject** are stored as strings.

The class should include a **constructor** and the following methods:

- **setTime**: to set the time.
- **setLocation**: to set the location.
- **setSubject**: to set the subject.
- **getSubject**: to return the subject of the meeting.
- **printDetails**: to print all information of a meeting in the following form:

Meeting in room 205 at 12:30; Subject: Examiner's meeting.

GOOD LUCK !

Philadelphia University
Faulty of Information Technology
Department of Software Engineering

Marking Scheme and Outline Solutions of the First Exam

Module: Object-Oriented Paradigms (721112) **(Section 3)**
Exam Date: November 28, 2006
Lecturer: Dr. Nadia Y. Yousif

First Semester 2006 / 2007

……

There are two parts in this exam paper. Part I contains one question that carries 5 marks. Part II contains two questions each of which carries 5 marks.
The following is the marking scheme and the set of outline solutions for the questions. It includes breakdown of the marks to each part of the question and the steps of the solution. It also describes the type of answer required to gain the stated marks.

1- Question 1 (5 marks)

This question contains 5 multiple choice questions with each question worth 1 mark. You should write the letter A, B, C, or D in your answer sheet to indicate your correct answer.

1) One of the following is NOT TRUE about Object-Oriented Paradigms (OOPs):

A) OOP is a set of techniques and processes focusing on how to analyse and model a real world problem and to design a solution.
B) The intended benefits of OOP are to solve the "software" crisis, break complexity into small manageable chunks, and make software maintenance easier.
C) OOP allows reuse of components – plug and play
D) OOP solves the entire problem in one program.

2) Which point is FALSE from the following?

A) A class is an object
B) A class is a template or prototype that defines the composition and the behavior of all objects of certain kinds.
C) A class may have fields of composite types
D) From a class you may initiate an object.

3) Methods of a class are invoked from outside the class by

A) objects using the dot notation. e.g. circle_1.moveHorizontal (50)
B) using its name only e.g. moveHorizontal (50)
C) using the call statement e.g. CALL moveHorizontal (50)
D) A and B above

4) A method in a class that is used to change the values of some fields in that class is called:

A) A constructor
B) An accessor method
C) A mutator method
D) None of the above

5) If there are one or more constructors for a class then

A) Exactly one of the constructors will be called each time an object of that class is created
B) All of the constructors will be called each time an object of that class is created
C) A destructor must also be written.
D) None of the above, classes cannot have constructors

……

The aim of the question in this part is to evaluate student's required minimal knowledge and understanding skills in the fundamentals of object oriented programming. It is intended to students of intermediate level.

The answer is:

1) D 2) A 3) A 4) C 5) A	**1 mark each**

……

Question 2 (5 marks)

Consider the following class definition that is written without methods. Instead, it has comments followed by blank areas denoted by …… to describe what each method has to do. Add method definitions in those blank areas as indicated in the comment. You will get **one mark** for each method definition.

```
/**
 * The VideoTape class holds information about a single television programme recorded on a video tape
 * and it is used in a video shop system. It holds the video tape details.
 */
public class VideoTape
{ private String title;              // the title of the programme
  private String classification;    // classification of the programme (comedy, drama, action, or romance)
  private int time;                 // the running time of the programme in minutes

   // Create a new video tape with a given title, classification, and time.

   public VideoTape (String fullTitle, String programClassification, int runningTime )
   {  title = fullTitle;
      classification = programClassification;
      time = runningTime;
   }

   // Return the title of this video tape.
        …….

  // Return the classification of this video tape.
        …….

  // Return the time of this video tape as a string in the following format:    2:06.
        …….

 // Set a new classification for this video tape.
        …….

  /* Print the details of the video tape to the output terminal in the following format:
 *  Adil Emam (COMEDY) 2:16
  */
        ……..
```

……

This question is to test that students have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP. It is intended to students of intermediate level

One possible solution to these methods is as follows:

// Return the title of this video tape.
public String getTitle( )
{ return title; }
// Return the classification of this video tape.
public String getClassification ( )
{ return classification; }
	1 mark
	
	1 mark

// Return the time of this video tape as a string in the following format: 2:06.
public String getTime( )
{ int hour = time / 60;
int minute = time % 60;
if (minute > 9)
return hour + ":" + minute ;
else
return hour + ":0" + minute ;
}
	1 mark

// Set a new classification for this video tape. public void setClassification (String clas1) { classification = clas1; }	**1 mark**
/* Print the details of the video tape to the output terminal in the following format: * Adil Emam (COMEDY) 2:16 */ public void printDetails ( ) { System.out.println (title + "(" + classification + ")" + getTime( ); }	**1 mark**

……

This question is to test that students have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP to practice object interction. It is intended to students of intermediate level

Question 3 (5 marks)

Design a class called **ISBN** to represent an International Standard Book Number, or ISBN for short.
The ISBN consists of 10 digits divided into 4 parts. For example, the ISBN 0 941831 39 6 represents the following information:

The first part: The first digit "0" signifies the book is from an English speaking country.
The second part: "941831" identifies the publisher.
The third part: "39" is the title number for the book.
The fourth part: "6" is a check digit to indicate that the sum of the ISBN digits is 10.

The class should have a constructor and methods to set and get the ISBN as a string.
Design a **Book** class that represents relevant information about a book, including the book's title, author, publisher, city and date of publication, and price. The class should also include the field **ISBN** isbnNum; where **ISBN** is the class defined above.
This class should include a **constructor** and the following methods:

- **setBookISBN**: to set the ISBN for the book.
- **getAuthor**: to return the author of the book.
- **getBookISBN**: to get the ISBN of the book.
- **printDetails**: to print the information of a book in the following form:

Book Title: Object First with Java
Book Author: David j. Barnes and Michael Kolling
Publisher: Prentice Hall
ISBN: 0 941831 39 6

```
/**
 * This class demonstrates the use of object interaction.
 * Class Book has one of its fields as an object of type ISBN.
 *
 * @author (Dr. Nadia Y. Yousif)
 * @version (date 28/11/2006)
 */
public class Book
{
  private String title;                                                    0.5 mark
  private String author;
  private String publisher;
  private String city;
  private String date;
  private float price;
  private ISBN isbnNumber;      // object of class ISBN

  // Constructor
  public Book(String t, String au, String pu, String ci, String d, float pr)
  { title = t;                                                             0.5 mark
    author = au;
    publisher = pu;
    city = ci;
    date = d;
    price = pr;
    isbnNumber = new ISBN ( );
  }

  // Method to set the book's ISBN                                         0.5 mark
   public void setBookISBN (int n1, int n2, int n3, int n4)
   { isbnNumber.setISBN (n1, n2, n3, n4);
   }
   //Method to return the author
   public String getAuthor ()                                            0.5 mark
   { return author;  }

   //Method to return the book's ISBN as a string
    public String getBookISBN ( )                                        0.5 mark
   { return isbnNumber.getISBN ( ) ;
   }

   // Method to print detatils of the book                               0.5 mark
      public void printDetails ( )
   { System.out.println ("Book Title: " + title);
     System.out.println ("Book Author: " + author);
     System.out.println ("Publisher: " + publisher);
     System.out.println ("ISBN: " + getBookISBN ( ));
   }
}
```

```
/**
 * Class ISBN describes the Internation standard book numbering.
 * Each ISBN is 10 digits number dividined into four parts.
 * The first (1 digit only) represents the language of the book;
 * the second (6 digits) represents the publisher number;
 * the third (2 digits) represents the number of the book;
 * the fourth (1 digit) represents the check of the vvlidity of the number.
```

```
 *
 * @author (Dr Nadia Y. Yousif)
 * @version (date 28/11/2006)
 */
public class ISBN
{
        // instance variables
        private int countryNumber;
        private int publisherNumber;                                          0.5 mark
        private int titleNumber;
        private int checkDigit;
        /**
         * Constructor for objects of class ISBN
         */
        public ISBN()
        { countryNumber = 0;
          publisherNumber = 0;
          titleNumber = 0;                                                    0.5 mark
          checkDigit = 0;
        }

        //Method to return the ISBN as a string
        public String getISBN( )
        { return countryNumber + " " + publisherNumber + " " +titleNumber+ " "   0.5 mark
                        + checkDigit;
        }

        //Method to set the ISBN of book
        public void setISBN(int n1, int n2, int n3, int n4)
        {                                                                     0.5 mark
          countryNumber = n1;
          publisherNumber = n2;
          titleNumber = n3;
          checkDigit = n4;
        }
}
```

Philadelphia University
Lecturer: Dr. Nadia Y. Yousif
Coordinator: Dr. Issa Shihabat
Internal Examiner: Dr. Amer Abu Ali

Faculty of Information Technology
Software Engineering Dept.

Examination Paper

Object-Oriented Paradigms (721112) - Section: 3 **Second Exam**
Date: May 14, 2006 **Second Semester - 2005/2006** **Time: 50 Minutes**

Information for Candidates

1. *This examination paper contains **3** questions totaling **20** marks*
2. *The marks for parts of questions are shown in square brackets: e.g. **[2 marks]**.*

Advice to Candidates

1. You should attempt ALL questions. You should write your answers clearly.

I. Basic Notions

Objectives: *The aim of the question in this part is to evaluate your required minimal knowledge and understanding skills in the fundamentals of OOP and class library of Java.*

Question 1 (5 marks)

(a) What are the differences between instance variables and static variables? **[2 marks]**
(b) What does the method **nextInt** of class **Random** in java.util package return if it is called without parameter and if it is called with an integer parameter? **[1 mark]**
(c) List the advantages and disadvantages of using a collection class ArrayList in place of an array. **[2 marks]**

II. Familiar Problem Solving

Objectives: *The aim of the questions in this part is to evaluate that you have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP.*

Question 2 (5 marks)

Write a class that has a string field called **str** and a method that tokenizes **str** into tokens, keeps the tokens in an array of strings, then takes each token from the array and prints it to the screen.

Question 3 (10 marks)

(a) Suppose that class **Car** exists. What is wrong with the following code? Illustrate by giving an example of the use of this code where this would be an issue. **[2 marks]**
(b) How should you improve the code? **[4 marks]**
(c) Write class **Car** that has two fields: **model** that represents the model of the car; and **year** that represents the year of manufacturing. It has **accessor** methods and **toString** method that returns a string representing the information of a car. **[4 marks]**

```
import java.util.*;
public class myCarCollection
{       private ArrayList cars;

        // A method that returns information of all cars in the collection
        public String toString ()
        {  String result = "My Car Collection \n";
           Iterator iter = cars.iterator ();
           while (iter.hasNext () )
           {    Car car = (Car) iter.next ();
                result = result + " " + car.toString ();
                result = result + "\n";
            }
           return result;
         }
}
```

GOOD LUCK!

Philadelphia University
Lecturer: Dr. Nadia Y. Yousif
Coordinator: Dr. Nadia Y. Yousif
Internal Examiner: Dr. Murad Maouch

Faculty of Information Technology
Software Engineering Department

Examination Paper

……

Object-Oriented Paradigms (721112) - Section: 3 **Second Exam**
Date: January 7, 2007 **First Semester - 2006/2007** **Time: 50 Minutes**

……

Information for Candidates

*1. This examination paper contains **4** questions totalling **20** marks*
*2. The marks for parts of questions are shown in square brackets: e.g. **[2 marks]**.*

Advice to Candidates

1. You should attempt questions so that the total mark is 15 and Question 4 should one of them. You should write your answers clearly.

……

I. Basic Notions

Objectives: *The aim of the question in this part is to evaluate your required minimal knowledge and understanding skills in the fundamentals of object oriented programming.*

Question 1 (5 marks)

Fill each of the following blanks with one of the given words: {type, next, Map, overloading, class}

(a) ---------- means that a class may contain more than one constructor, or more than one method of the same name, as long as each has a different set of parameters type
(b) A class name can be used as the ---------- for a variable.
(c) A type of a flexible-size collection that permits to store a pair of information (the key and the value) is called a ---------- .
(d) Iterator method that gets the next unprocessed item in the collection is called ---------- .
(e) A ---------- variable (or a static variable) is a variable that is shared by all objects of a class

……

II. Familiar Problem Solving

Objectives: *The aim of the questions in this part is to evaluate that you have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP.*

Question 2 (5 marks)

a) What might go wrong with the following code, what would you need to do to fix it. **[2 marks]**

```
public class Test
{  int x;
   int y;
  public Test  (int x, int y)
  {   x = x;
      y = y;
  }
}
```

b) Write a method that generates 100 integer random numbers between 0 and 10 and counts the even numbers from those generated numbers. **[3 marks]**

……

Question 3 (5 marks)

a) Write a declaration for an array variable **people** that could be used to refer to an array of **Person** objects and write a statement to create such array. **[2 marks]**
b) Write a declaration for an ArrayList variable **library** and any possible statements that could be used to create the array list and to store **Book** objects in library array list. **[3 marks]**

……

III. Unfamiliar Problems Solving

Objectives*: The aim of the question in this part is to evaluate that you can solve familiar problems with ease and can make progress towards the solution of unfamiliar problems in object interaction and in String manipulation.*

Question 4 (5 marks)

Write class **Employee** that has fields for employee's name, ID, and salary. The **Employee** class has also another field called address which is of type class **Address**. Class **Address** has fields for the street name, house number, and city.

Include in class **Employee** any suitable accessor and mutator methods and write a method called **nameTokenizing** that tokenizes the name of the employee and prints the tokens.

GOOD LUCK!

Philadelphia University
Faulty of Information Technology
Department of Software Engineering

Marking Scheme and Outline Solutions of the Second Exam

Module: Object-Oriented Paradigms (721112) **(Section 2)**
Exam Date: May 14, 2006
Lecturer: Dr. Nadia Y. Yousif

Second Semester 2005 / 2006

……

There are two parts in this exam paper. Part I contains one question that carries 5 marks and contains 3 sections. Part II contains two questions: Question (2) carries 5 marks and Questions (3) carries 10 marks and three sections.

The following is the marking scheme and the set of outline solutions for the questions. It includes breakdown of the marks to each part of the question and the steps of the solution. It also describes the type of answer required to gain the stated marks.

1- Question 1 (5 marks)

(a) What are the differences between instance variables and static variables? **[2 marks]**
(b) What does the method **nextInt** of class **Random** in java.util package return if it is called without parameter and if it is called with an integer parameter? **[1 mark]**
(c) List the advantages and disadvantages of using a collection class ArrayList in place of an array. **[2 marks]**

……

This question is to test students understanding of the fundamentals of OOP and class library of Java. It is intended to students of intermediate level.

The student's answer is preferred to look like the following:

(a) Instance variable: it is the one that is associated with an object of a given type. It is called by writing the name of the object followed by a period and then the name of the variable. It is part of the object's state. (b) Static variable is a class variable that is instantiated by the class itself not by an instance of the class.	**1 mark** **1 mark**
(b) nextInt method without parameter returns any random integer between - lower and + upper, where lower and upper are the possible range of integers in the computer. nextInt method with parameter (n) returns any random integer between 0 (inclusive) and n (exclusive).	**0.5 mark** **0.5 mark**
(c) Advantages of using ArrayList in place of an array: 1- No limit on its size (the array has a fixed size) 2- It is a flexible structure that can grow or shrink dynamically. The array cannot do that. 3- It can store any type of objects; the arrays store one type of objects. 4- It has methods to add, remove, get an element from the list, where as arrays don't have such operations. Disadvantages of using ArrayList in place of an array: 1- When removing an element from the list, it takes time to readjust the locations of the remaining elements.	**1.5 marks** **0.5 mark**

……

Question 2 (5 marks)

Write a class that has a string field called **str** and a method that tokenizes **str** into tokens, keeps the tokens in an array of strings, then takes each token from the array and prints it to the screen.

……

This question is to test that students have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems that practice StringTokenizer class. It is intended to students of intermediate level

import java.util.StringTokenizer;	**0.5 mark**
/**	
* class Tokens to practice StringTokenizer.	
*/	
public class Tokens	
{ // instance variables	**0.5 mark**
private String str;	
// Constructor for objects of class Tokens	**0.5 mark**
public Tokens(String s)	
{ str = s; }	
// tokenize Method that tokenizes a string and stores the tokens in an array	
public void tokenize()	
{ String [] arrayTokens;	**0.5 mark**
arrayTokens = new String[10];	
int index=0;	
String token="";	
StringTokenizer st = new StringTokenizer(str);	**1 mark**
// Keep tokenizing the string until there are no more tokens	
while (st.hasMoreTokens())	
{ token = st.nextToken();	**1 mark**
arrayTokens[index] = token;	
index++;	
}	
// Get elements from the array and print them	
for (int i = 0; i<index; i++)	**1 mark**
{ System.out.println ("Token" + i +": " + arrayTokens[i]);	
}	
}	
}	

……

This question is to test that students have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP. It is intended to students of intermediate level

Question 3 (10 marks)

(a) Suppose that class **Car** exists. What is wrong with the following code? Illustrate by giving an example of the use of this code where this would be an issue. **[2 marks]**

(b) How should you improve the code? **[4 marks]**

(c) Write class **Car** that has two fields: **model** that represents the model of the car; and **year** that represents the year of manufacturing. It has **accessor** methods and **toString** method that returns a string representing the information of a car. **[4 marks]**

```
import java.util.*;
public class myCarCollection
{       private ArrayList cars;
        // A method that returns information of all cars in the collection
        public String toString ()
        {  String result = "My Car Collection \n";
           Iterator iter = cars.iterator ();
           while (iter.hasNext () )
           {    Car car = (Car) iter.next ();
                result = result + " " + car.toString ();
                result = result + "\n";
            }
           return result;
        } }
```

(a) 1- This class has no constructor to initialize the ArrayList cars. 2- There is no method to create the cars list. Therefore, the iterator method does not work on empty list. For example, if we call the toString method, it will give Null exception because list cars has no elements to iterate on.	**2 marks**
(b) We write the constructor as follows: public myCarCollection ( ) {cars = new ArrayList ( ); } We write a method to fill the list with car objects: public void fillArray (Car c) { cars.add (c); }	**2 marks** **2 marks**

(c) Car class could be like this:
/ * Car class stores information about the car. Its model and year of manufacturing
*
* @author (Nadia)
* @version (Version 1- 10/5/2006)
*/
public class Car
{ private String model;
private int year;
// Constructor for objects of class Car
public Car (String m, int y)
{ model =m ;
year = y;
}
 	
// to return the model of the car
public String getModel()
{ return model;
}
// to return the year of manufacturing
public int getYear()
{ return year;
}
// to print all informtion about this car
public String toString()
{ return "model= "+model+" Year = "+year;
}
}
	1 mark
	
	1 mark
	
	0.5 mark
	
	0.5 mark
	
	1 mark

Philadelphia University
Faulty of Information Technology
Department of Software Engineering

Marking Scheme and Outline Solutions of the Second Exam

Module: Object-Oriented Paradigms (721112) **(Section 3)**
Exam Date: January 7, 2007
Lecturer: Dr. Nadia Y. Yousif

First Semester 2006 / 2007

……

There are three parts in this exam paper. Part I contains one question that carries 5 marks. Part II contains two questions each of which carries 5 marks. Part III contains one question that carries 5 marks.
The following is the marking scheme and the set of outline solutions for the questions. It includes breakdown of the marks to each part of the question and the steps of the solution. It also describes the type of answer required to gain the stated marks.

1- Question 1 (5 marks)

Fill each of the following blanks with one of the given words: {type, next, Map, overloading, class}

(a) ---------- means that a class may contain more than one constructor, or more than one method of the same name, as long as each has a different set of parameters type
(b) A class name can be used as the ---------- for a variable.
(c) A type of a flexible-size collection that permits to store a pair of information (the key and the value) is called a ---------- .
(d) Iterator method that gets the next unprocessed item in the collection is called ---------- .
(e) A ---------- variable (or a static variable) is a variable that is shared by all objects of a class

……

The aim of the question in this part is to evaluate student's required minimal knowledge and understanding skills in the fundamentals of object oriented programming. It is intended to students of intermediate level.

The answer is:

(a) overloading	(b) type	(c) Map	(d) next	(e) class	**1 mark each**

……

Question 2 (5 marks)

a) What might go wrong with the following code, what would you need to do to fix it. **[2 marks]**

```
public class Test
{  int x;
   int y;
  public Test  (int x, int y)
   {   x = x;
       y = y;
   }
}
```

b) Write a method that generates 100 integer random numbers between 0 and 10 and counts the even numbers from those generated numbers. **[3 marks]**

……

This question is to test that students have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP. It is intended to students of intermediate level

One possible solution is as follows:

a) The error is in the constructor where x = x; y = y; and the correction is
public Test (int x1, int y1)
{ x = x1;
y = y1;
}
	1 mark
	
	1 mark

b) public void generateNumbers ( )
{ int count = 0;
Random rand = new Random ( ) ;
for (int i = 0; i < 100; i++)
{ int number = rand.nextInt (10);
if (number % 2 == 0)
count ++;
}
System.out.println ("Count of Even = " + count);
}
	0.5 mark
	
	0.5 mark
	0.5 mark
	0.5 mark
	1 mark

……

Question 3 (5 marks)

a) Write a declaration for an array variable **people** that could be used to refer to an array of **Person** objects and write a statement to create such array. **[2 marks]**

b) Write a declaration for an ArrayList variable **library** and any possible statements that could be used to create the array list and to store **Book** objects in the library array list. **[3 marks]**

……

This question is to test that students have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP to practice using arrays and array lists. It is intended to students of intermediate level

a) Person people [ ]; // array declaration
people = new Person [10]; // creating the array people that holds 10 objects of type Person.
	1 mark
	1 mark

b) ArrayList library; // array list declaration
library = new ArrayList ( ); // creating the array list library
library.add (book1); // adding an object of type Book into library array list
	1 mark
	1 mark
	1 mark

……

Question 4 (5 marks)

Write class **Employee** that has fields for employee's name, ID, and salary. The **Employee** class has also another field called address which is of type class **Address**. Class **Address** has fields for the street name, house number, and city.

Include in class **Employee** any suitable accessor and mutator methods and write a method called **nameTokenizing** that tokenizes the name of the employee and prints the tokens.

……

The aim of this question is to evaluate that students can solve familiar problems with ease and can make progress towards the solution of unfamiliar problems in object interaction and in String manipulation.

import java.util.StringTokenizer; **public class Employee** **{ // instance variables** **private String name;** **private int ID;** **private double salary;** **private Address address;** **// Constructor for objects of class Employee** **public Employee(String na, int id, double sal)** **{ name = na;** **ID = id;**	**0.5 mark** **0.5 mark**

salary = sal;
address = new Address ();
}
public String getName()
{ return name; }
public int getID()
{ return ID; }
public double getSalary()
{ return salary; }
public void setAddress(String st, int num, String cty)
{ address.changeAddress(st, num, cty);
}
public String getAddress()
{ return address.getStreet()+ address.getHouseNumber()+ address.getCity();
}
public void nameTokenizer()
{ StringTokenizer st = new StringTokenizer(name);
while (st.hasMoreTokens())
{ String token = st.nextToken ();
System.out.println ("Token is " + token);
}
}
}
	0.5 mark
	
	1 mark

public class Address
{// instance variables
private String street;
private int hnumber;
private String city;
 	
// Constructor for objects of class Address
public Address()
{ street = "Unknown";
hnumber = 0;
city = "Unknown";
}
public String getStreet()
{ return street; }
public String getCity()
{ return city; }
public int getHouseNumber()
{ return hnumber; }
public void changeAddress(String st, int hnum, String cty)
{ street = st;
hnumber = hnum;
city = cty;
}
}
	0.5 mark
	
	0.5 mark
	
	0.5 mark
	
	1 mark

Philadelphia University
Lecturer: Dr. Nadia Y. Yousif
Coordinator: Dr. Nadia Y. Yousif
Internal Examiner: Dr. Amer Abu Ali

Faculty of Information Technology
Software Engineering Department

Examination Paper

Object-Oriented Paradigms (721112) - Section: 3 **Final Exam**
Date: June 14, 2006 **Second Semester - 2005/2006** **Time: 2 Hours**

Information for Candidates

1. *This examination paper contains **6** questions totaling **50** marks*
2. *The marks for parts of questions are shown in square brackets: e.g. **[2 marks]**.*

Advice to Candidates

1. You should attempt ALL questions. You should write your answers clearly.

I. Basic Notions

***Objectives:** The aim of the question in this part is to evaluate your required minimal knowledge and understanding skills in the fundamentals of OOP.*

Question 1 (5 marks)

For each of the following sections select the correct answer. **[1 mark each]**

1. A method that is called automatically each time an object is created is a
   a) constructor
   b) accessor function
   c) mutator method
   d) None of the above

2- Which of the following **is not** one of the major support of object-oriented programming?
a) Encapsulation
b) Data Hiding
c) Inheritance
d) Structured Programming

3- Encapsulation provides
a) inheritance
b) information hiding
c) polymorphism
d) none of the above

4- The static variable declared in a class is called
a) Global variable
b) Local Variable
c) Class variable
d) Instance variable

5- In an Interface
a) Only one of its methods is abstract method
b) All of its methods are abstract
c) Some methods are abstract and some are concrete
d) There are no methods

Question 2 (7 marks)

NOTE: Answer **EITHER** section (a) or section (b).

(a)

i) The following are examples of good programming practice: **[1.5 marks each]**

1- Do not use public instance variables.
2- Use appropriate class, method and variable names.
3- Always initialize instance variables in a constructor.

For each example, explain why.

ii) What is wrong with each of the following class definitions? How would you fix them? **[2.5 marks]**

```
class Point
{ private int x, y;
  public Point (int x, int y)
   {this.x = x; this.y = y;}
  public int getX ( )        { return x; }
  public int getY ( )        { return y; }
 }
class ScaledPoint extends Point
{ private int c;
  public ScaledPoint (int x, int y, int c)
   { this.x = x; this.y = y; this.c = c;}
  public void getC ( )   { return c; }
}
```

(b) Show the trace and the output of the following Java program. Choose any appropriate sample data for your tracing. **[7 marks]**

```
import java.util.Random;
// A class to find the sum and count of the even and odd numbers generated randomly.
public class RandomTest
{   private Random rand;
    private int evenSum, evenCnt;
    private int oddSum, oddCnt;
    // Constructor for objects of class RandomTest
    public RandomTest()
    {   rand = new Random();
        evenSum = 0;   evenCnt = 0;
        oddSum = 0;     oddCnt = 0;
    }
    //A method to find the sum and the count of even and odd numbers separately.
    public void summing()
    {  int x;
       for (int i = 1; i<6; i++)
       { x = (int) rand.nextInt(50);
         if ( x % 2 == 0 )
            { evenSum += x ;  evenCnt++ ; }
         else
            { oddSum += x;    oddCnt++ ; }
        System.out.print (x + "   ");
       }
       System.out.println ( );
       System.out.println ("Even Sum = "+evenSum + " Odd Sum = "+oddSum);
       System.out.println ("Even Count = "+ evenCnt + " Odd Count = "+oddCnt);
    }
}
```

……..……..……..……

Question 3 (10 marks)

From the following given list:

(toString, override, polymorphism, inheritance, protected, public, private, dynamic, static, set, Integer, Float, duplication, super, ArrayList, Map)

select the appropriate word to fill the blank in each of the following: **[1 mark each]**

1- The ---------- type of a variable is the type of the object that is currently stored in the variable.
2- When the same method call may at different times invoke different methods based on the dynamic type, this is called method ---------- .
3- A subclass can ---------- a superclass method by declaring a method with the same signature as the superclass method.
4- Every object in Java has a ---------- method that can be used to return a String representation of it.
5- The ---------- type of a variable is the type as declared in the source code in the variable declaration statement.
6- The name of the int wrapper class is ---------- .
7- ---------- allows us to define one class as an extension of another.
8- The keyword used to call methods in the superclass is ---------- .
9- Code ---------- (having the same segment of code in an application more than once) is a sign of bad design and should be avoided.
10- A collection that stores each element at most once and does not maintain any specific order is called a ----------.

……..……..……..……

II. Familiar Problem Solving

Objectives: *The aim of the questions in this part is to evaluate that you have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP.*

Question 4 (6 marks)

Consider the following class definitions and answer the questions that follow.

```
public class Kid
{ String name;
  int friends;
  public Kid(String name)
  { this.name = name;
    friends = 0;
  }
  public void getFriend(Kid k)
  {  friends++;  }
  public int numFriends()
  {  return friends;  }
  public String playsWith()
  { return "toys";   }
  public String toString()
  { return "a kid named "+name;}
}
```

```
public class Girl extends Kid
{  public Girl(String name)
   {  super(name);
      friends = 1; // has self esteem
   }
   public void getFriend(Kid k)
   {  if (k instanceof Girl)
      // her friends come along too
      friends += k.numFriends( );
      else
           friends++;
    }
   public String toString( )
   { return "a girl named "+ name;}
 }
```

```
public class Boy extends Kid
{ public Boy(String name)
  { super(name); }
  public String playsWith( )
  { return (super.playsWith()
                + " frogs");
   }
   public String toString()
   { return "a boy named " +
              name; }
 }
```

Given the above class definitions, look at the following code and indicate which line of code that would result in a compiler error and those lines that would compile, but would result in an execution error. In considering each line, assume that all the correct lines above it have executed.

```
Kid   snoopy, woodstock;
Boy  ali, ahmed;
Girl  saba, sally;
snoopy = new Kid("snoopy");
ali = new Boy("ali");
ahmed = new Kid ("ahmed");
ahmed = snoopy;
ahmed = (Boy) snoopy;
woodstock = new Boy("pigpen");
ahmed = woodstock;
ali = (Boy) woodstock;
saba = new Girl("saba");
sally = saba;
ahmed = (Boy) saba;
```

……..……..……..……

Question 5 (10 marks)

A hospital wants to create a database regarding its indoor patients. The information to store include

- Name of the patient
- Age of the patient
- Disease
- Date of admission
- Date of discharge

Create a class called **Patient** to store the above information. The member methods should include methods to enter information and display the patient's information.

Create class **Date** to have the date (year, month and day as its fields) and a method to display the date.

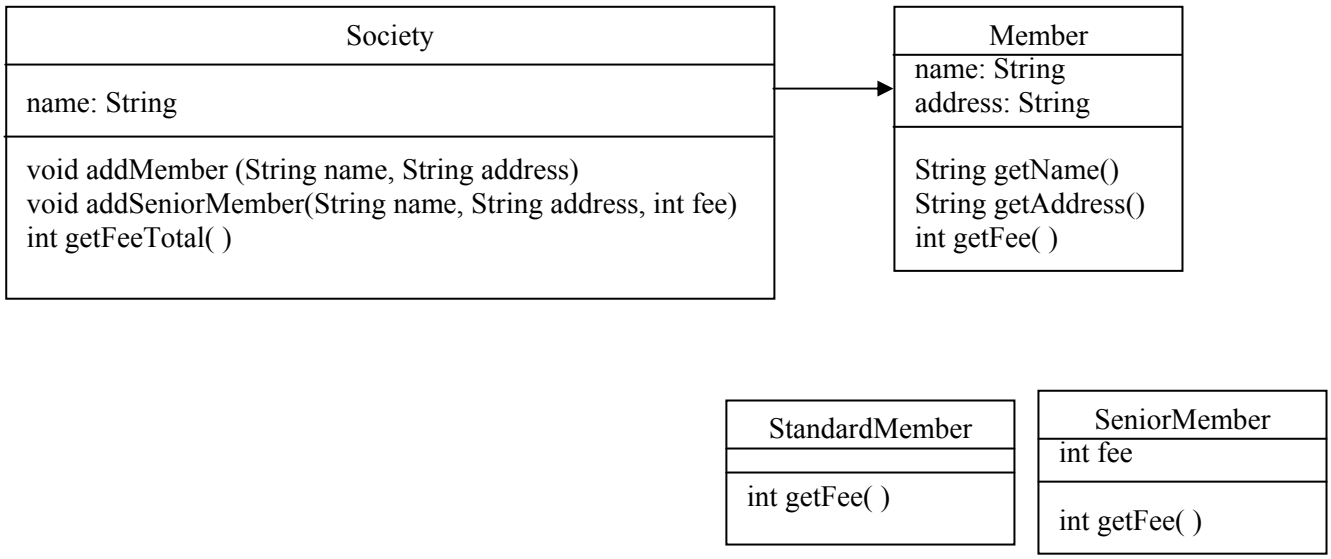
Create class **Hospital** to have an array list to store all patients. It has methods to add a patient to the list and to delete a patient from the list. It also has a method to display a list of all the patients in the hospital and a method to display only the patients whose age is less than 12.

III. Unfamiliar Problems Solving

***Objectives**: The aim of the question in this part is to evaluate that you can solve familiar problems with ease and can make progress towards the solution of unfamiliar problems, and can set out reasoning and explanation in a clear and coherent manner.*

Question 6 (12 marks)

Consider the following class diagram showing part of a program to manage the membership information for a professional society:



a) Class **Member** is an abstract class. Explain the role of an abstract class. **[1 marks]**

b) Write a Java version of class **Member** assuming it has this constructor:

```
public Member(String name, String address)
```

and that the method **getFee( )** is abstract. **[3 marks]**

c) Write a Java version of class **StandardMember** assuming it has this constructor:

```
public StandardMember (String name, String address)
```

and the standard membership fee is fixed at 30 JD. **[2 marks]**

d) Write a Java version of class **SeniorMember** assuming it has this constructor:

```
public SeniorMember(String name, String address, int fee)
```

where the membership fee is set when a **SeniorMember** object is created. **[2 marks]**

e) Write a Java version of class **Society** assuming it has this constructor:

```
public Society(String societyName)
```

[4 marks]

where **Society** has members of different types stored in an arraylist and **getFeeTotal** method that returns the total fees of all members in the society.

GOOD LUCK!

Philadelphia University
Lecturer: Dr. Nadia Y. Yousif
Coordinator: Dr. Nadia Y. Yousif
Internal Examiner: Dr. Murad Maouch

Faculty of Information Technology
Software Engineering Department

Examination Paper

……

Object-Oriented Paradigms (721112) - Section: 3 Final Exam
Date: February 5, 2007 First Semester - 2006/2007 Time: 2 Hours

……

Information for Candidates

*1. This examination paper contains **5** questions totalling **50** marks*
*2. The marks for parts of questions are shown in square brackets: e.g. **[2 marks]**.*

Advice to Candidates

1. You should attempt all question. You should write your answers clearly.

……

I. Basic Notions

***Objectives:** The aim of the question in this part is to evaluate your required minimal knowledge and understanding skills in the fundamentals of object oriented programming.*

Question 1 (10 marks)

Fill each of the following blanks with one of the words given in the list below: **[1 mark each]**

{dynamic, polymorphism, protected, interface, override, objects, inheritance, static, tokenizing, super}

1- ……. are used to model things from a problem domain.
2- The term …… refers to the fact that a variable can hold objects of different types.
3- In an ……. all methods are abstract.
4- The keyword …… is used to call methods in the superclass.
5- …….. is an access control keyword that permits access from the subclass, but denies access from anywhere not in a class or its subclass.
6- To cut up an input string into separate words is called ……..
7- …….. allows us to define one class as an extension of another.
8- A subclass can …….. a superclass method by declaring a method with the same signature as the superclass method.
9- The class variable is declared as ……..
10- The …… type of a variable is the type of the object that is currently stored in the variable.

……..……..……

Question 2 (10 marks)

(a) What is wrong with the following class definitions? How would you fix them? **[3 marks]**

```
class Employee
{ private String name;
  private int id;
  public Employee (String name, int id)
  {  this.name = name;        this.id = id;     }
  public String getName ( )          { return name; }
  public int getId ( )               { return id; }
 }
class  Manager extends Employee
{ private String department;
  public Manager (String name, int id, String department)
  { this.name = name; this.id = id; this.department = department;}
  public void getDeprtment ( )   { return department; }
}
```

(b) What does this piece of code do? **[3 marks]**

```
Random rand = new Random ( );
String table = "";
String row = "";
for (int i = 1; i < 4; i++)
{   for (int j = 1; j < 4; j++)
    {   int number = rand.nextInt(50));
        row += number + " ";
    }
    table += row + "\n";
    row = "";
}
System.out.println(table);
```

(c) What is wrong with the following interface? Correct it. **[2 marks]**

```
public interface Figures
{   public void printMessage (String s)
    { System.out.println ("This figure is   " + s );    }
    public double area ( ) ;          // to calculate the area of the figure
}
```

(d) Design a class **Rectangle** that implements the corrected interface of section (c). **[2 marks]**

……..……..……

Question 3 (8 marks)

(a) What is meant by polymorphism? Why do Java languages which support inheritance also support polymorphism? **[4 marks]**

(b) Assume we have three classes: **Person, Teacher,** and **Student. Teacher** and **Student** are both subclasses of **Person**.
Which of the following assignments are legal, and why? **[4 marks]**

```
Person p;
Teacher t;
Student s;

t = new Teacher ( );
p = t;
s = (Student) t;
s = (Student) p;
p = new Student ( );
t = new Person ( );
t = p;
```

……..……..……

II. Familiar Problem Solving

Objectives: *The aim of the questions in this part is to evaluate that you have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in object interaction.*

Question 4 (9 marks)

Write the definition and implementation of the class **Robot** that has one field called ***position*** of type class **Point** and three methods. The methods are:

- **moveLeft** that moves the robot from the current position to the left for d distance.
- **moveRight** that moves the robot from the current position to the right for d distance.
- **moveForward** that moves the robot forward for d distance.

Class **Point** has two fields: x and y of type int, which represent the coordinates of a point in x-y plane and two accessor methods to get the values of x and y.

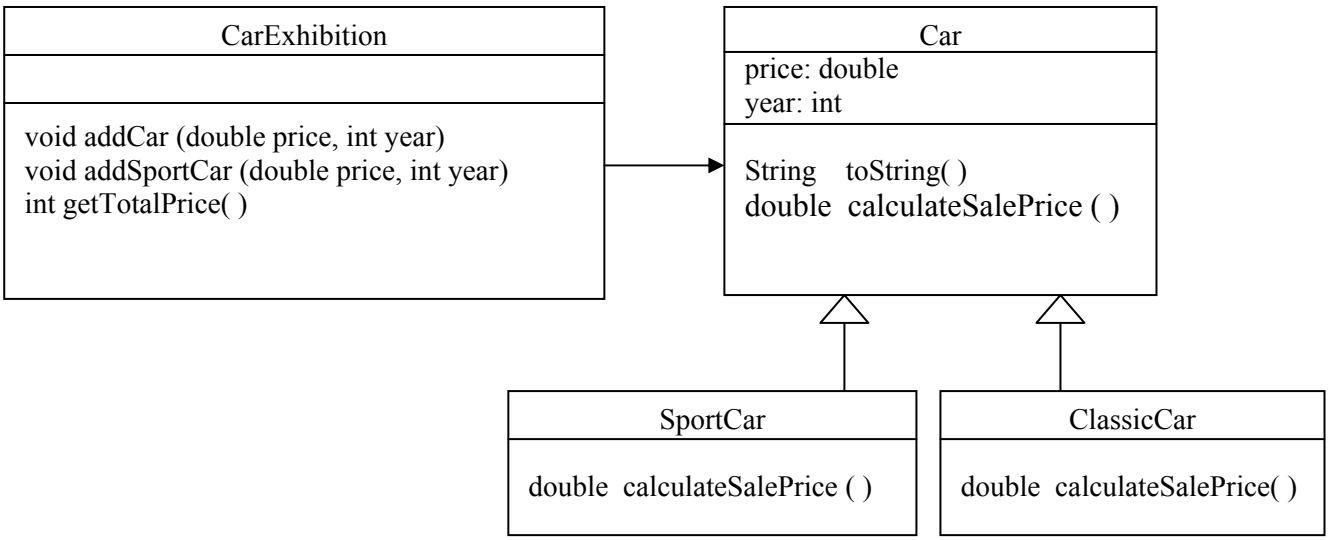
……..……..……

III. Unfamiliar Problems Solving

***Objectives**: The aim of the question in this part is to evaluate that you can solve familiar problems with ease and can make progress towards the solution of unfamiliar problems in inheritancee.*

Question 5 (13 marks)

Consider the following class hierarchy where Class **Car** is the supper class and the classes **ClassicCar** and **SportCar** are two subclasses derived from **Car**.
Class **CarExhibition** contains a filed of type ArrayList that stores objects of type **Car**.



(a) Class **Car** is an abstract class. Explain the role of an abstract class. **[1 marks]**

(b) Write a Java version of class **Car** assuming it has this constructor:

```
public Car(double price, int year)
```

and that the method **calculateSalePrice ( )** is abstract. **[3 marks]**

(c) Write a Java version of class **ClassicCar** assuming it has this constructor:

```
public ClassicCar (double price, int year)
```

[2 marks]

and that the method **calculateSalePrice ( )** returns 10,000 as the sale price of the car.

d) Write a Java version of class **SportCar** assuming it has this constructor:

```
public SportCar(double price, int year)
```

[3 marks]

and that the method **calculateSalePrice ( )** calculates the sale price of the car as follow:
if year > 2000 then the sale price is 0.75 * its original price; if year > 1995 then the sale price is 0.5 * its original price; otherwise the sale price is 0.25 * its original price

e) Write a Java version of class **CarExhibition** assuming it has this constructor:

```
public CarExhibition( )
```

[4 marks]

where **CarExhibition** has cars of different types stored in an arraylist and **getTotalPrice** method that returns the total prices of all cars in the exhibition.

GOOD LUCK!

Philadelphia University
Faulty of Information Technology
Department of Software Engineering

Marking Scheme and Outline Solutions of the Final Exam

Module: Object-Oriented Paradigms (721112) **(Section 3)**
Exam Date: Jun 14, 2006
Lecturer: Dr. Nadia Y. Yousif

Second Semester 2005 / 2006

……

There are three parts in this exam paper. Part I contains three questions: Question (1) carries 5 marks and contains 5 sections; Question (2) carries (7) marks and contains 2 sections from which the student will answer only one of them; Question (3) carries 10 marks. Part II contains two questions: Question (4) carries 6 marks and Questions (5) carries 10 marks. Part III contains only one Question that carries 12 marks.

The following is the marking scheme and the set of outline solutions for the questions. It includes breakdown of the marks to each part of the question and the steps of the solution. It also describes the type of answer required to gain the stated marks.

1- Question 1 (5 marks)

For each of the following sections select the correct answer. **[1 mark each]**

1. A method that is called automatically each time an object is created is a
   a) constructor
   b) accessor function
   c) mutator method
   d) None of the above

2- Which of the following **is not** one of the major support of object-oriented programming?
a) Encapsulation
b) Data Hiding
c) Inheritance
d) Structured Programming

3- Encapsulation provides
a) inheritance
b) information hiding
c) polymorphism
d) none of the above

4- The static variable declared in a class is called
a) Global variable
b) Local Variable
c) Class variable
d) Instance variable

5- In an Interface
a) Only one of its methods is abstract method
b) All of its methods are abstract
c) Some methods are abstract and some are concrete
d) There are no methods

……

This question is to test students understanding of the fundamentals of OOP. It is intended to students of intermediate level.

The correct answers are:

1-	(a)	**1 mark**
2-	(d)	**1 mark**
3-	(b)	**1 mark**
4-	(c)	**1 mark**
5-	(b)	**1 mark**

2- Question 2 (7 marks)

NOTE: Answer **EITHER** section (a) or section (b).

(a)

i) The following are examples of good programming practice: **[1.5 marks each]**

1- Do not use public instance variables.
2- Use appropriate class, method and variable names.
3- Always initialize instance variables in a constructor.

For each example, explain why.

ii) What is wrong with the following class definitions? How would you fix them?**[2.5 marks]**

```
class Point
{ private int x, y;
  public Point (int x, int y)
   {this.x = x; this.y = y;}
  public int getX ( )        { return x; }
  public int getY ( )        { return y; }
 }
class ScaledPoint extends Point
{ private int c;
  public ScaledPoint (int x, int y, int c)
   { this.x = x; this.y = y; this.c = c;}
  public void getC ( )   { return c; }
}
```

(b) Show the trace and the output of the following Java program. Choose any appropriate sample data for your tracing. **[7 marks]**

```
import java.util.Random;
// A class to find the sum and count of the even and odd numbers generated randomly.
public class RandomTest
{   private Random rand;
    private int evenSum, evenCnt;
    private int oddSum, oddCnt;
    // Constructor for objects of class RandomTest
    public RandomTest()
    {   rand = new Random();
        evenSum = 0;   evenCnt = 0;
        oddSum = 0;     oddCnt = 0;
    }
    //A method to find the sum and the count of even and odd numbers separately.
    public void summing()
    {  int x;
       for (int i = 1; i<6; i++)
       { x = (int) rand.nextInt(50);
         if ( x % 2 == 0 )
            { evenSum += x ;  evenCnt++ ; }
         else
            { oddSum += x;    oddCnt++ ; }
        System.out.print (x + "   ");
       }
       System.out.println ( );
       System.out.println ("Even Sum = "+evenSum + " Odd Sum = "+oddSum);
       System.out.println ("Even Count = "+ evenCnt + " Odd Count = "+oddCnt);
    }
}
```

……

This question is to test that students have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems that practice inheritance and Random class. It is intended to students of intermediate level Students should answer only one section.

(a) (i) 1- Use private instance variables to satisfy the concept of information hiding. 2- To be self documented class and easy to maintain and test. 3- If instance variables are not initialized in the constructor and an attempt was made to print them, then the result will be any value stored previously in these variables which may cause error in later processing.	**1.5 mark** **1.5 mark** **1.5 mark**

(a) (ii)

First error is in class ScaledPoint where the constructor tries to initialize the private variables of Point (x and y). **1.25 mark**

To fix it, we write: super (x, y) instead of this.x=x; this.y=y;

Second error is in getC method that returns an integer value an it is declared void. **1.25 mark**

To fix it, we write: public int getC( )

(b) **Tracing and Output** **1 mark**

The random number that are generated by rand object should be between 0 & 50 (50 excluded).

Suppose the 5 numbers that are generated are 15, 23, 12, 40, 7 then we trace the program on these data.

evenSum	evenCnt	oddSum	oddCnt	i	x	(x%2==0)	output	
0	0	0	0	1	15	false	15	
		15	1					**1 mark**
				2	23	false	15 23	
		38	2					**1 mark**
				3	12	true	15 23 12	
12	1							**1 mark**
				4	40	true	15 23 12 40	
52	2							**1 mark**
				5	7	false	15 23 12 40 7	
		45	3					
				6 (stop)				**1 mark**

Even Sum= 52 Odd Sum= 45
Even Count = 2 Odd Count = 3

……

3- Question 3 (10 marks)

From the following given list:

(toString, override, polymorphism, inheritance, protected, public, private, dynamic, static, set, Integer, Float, duplication, super, ArrayList, Map)

select the appropriate word to fill the blank in each of the following: **[1 mark each]**

1- The ---------- type of a variable is the type of the object that is currently stored in the variable.
2- When the same method call may at different times invoke different methods based on the dynamic type, this is called method ---------- .
3- A subclass can ---------- a superclass method by declaring a method with the same signature as the superclass method.
4- Every object in Java has a ---------- method that can be used to return a String representation of it.
5- The ---------- type of a variable is the type as declared in the source code in the variable declaration statement.
6- The name of the int wrapper class is ---------- .
7- ---------- allows us to define one class as an extension of another.
8- The keyword used to call methods in the superclass is ---------- .
9- Code ---------- (having the same segment of code in an application more than once) is a sign of bad design and should be avoided.
10- A collection that stores each element at most once and does not maintain any specific order is called a ----------.

……

This question is to test that students have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP. It is intended to students of intermediate level

1- dynamic	2- polymorphism	3- override	4- toString	5- static	**5 marks**
6- Integer	7- inheritance	8- supper	9- duplication	10- set	**5 marks**

4- ***Question 4 (6 marks)***

Consider the following class definitions and answer the questions that follow.

```
public class Kid
{ String name;
  int friends;
  public Kid(String name)
  { this.name = name;
    friends = 0;
  }
  public void getFriend(Kid k)
  {  friends++;  }
  public int numFriends()
  {  return friends;  }
  public String playsWith()
  { return "toys";   }
  public String toString()
  { return "a kid named "+name;}
}
```

```
public class Girl extends Kid
{  public Girl(String name)
   {  super(name);
      friends = 1; // has self esteem
   }
   public void getFriend(Kid k)
   {  if (k instanceof Girl)
      // her friends come along too
      friends += k.numFriends( );
      else
           friends++;
   }
   public String toString( )
   { return "a girl named "+ name;}
}
```

```
public class Boy extends Kid
{ public Boy(String name)
  { super(name); }
  public String playsWith( )
  { return (super.playsWith()
                + " frogs");
  }
  public String toString()
  { return "a boy named " +
              name; }
}
```

Given the above class definitions, look at the following code and indicate which line of code that would result in a compiler error and those lines that would compile, but would result in an execution error. In considering each line, assume that all the correct lines above it have executed.

```
Kid   snoopy, woodstock;
Boy  ali, ahmed;
Girl  saba, sally;
snoopy = new Kid("snoopy");
ali = new Boy("ali");
ahmed = new Kid ("ahmed");
ahmed = snoopy;
ahmed = (Boy) snoopy;
woodstock = new Boy("pigpen");
ahmed = woodstock;
ali = (Boy) woodstock;
saba = new Girl("saba");
sally = saba;
ahmed = (Boy) saba;
```

……..……..……..……

This question is to test that students have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP especially in Inheritance and Polymorphism. It is intended to students of intermediate level

The errors are as follows:

1- In the assignment: ahmed = new Kid ("ahmed"); //error: incompatible type	**1.5 mark**
2- In the assignment: ahmed = snoopy; //error: incompatible type	**1.5 mark**
3- In the assignment: ahmed = (Boy) snoopy; //error: class cast exception	**1.5 mark**
4- In the assignment: ahmed = woodstock; //error: incompatible type	**1.5 mark**

……..……..……..……

5- Question 5 (10 marks)

A hospital wants to create a database regarding its indoor patients. The information to store include

- Name of the patient - Age of the patient - Disease - Date of admission - Date of discharge

Create a class called **Patient** to store the above information. The member methods should include methods to enter information and display the patient's information.
Create class **Date** to have the date (year, month and day as its fields) and a method to display the date.
Create class **Hospital** to have an array list to store all patients. It has methods to add a patient to the list and to delete a patient from the list. It also has a method to display a list of all the patients in the hospital and a method to display only the patients whose age is less than 12.

……..……..……..……

This question is to test that students have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP especially in Inheritance and object composition. It is intended to students of more than intermediate level.

One possible solution could be as follows:

1- Class Hospital — **0.5 mark**

```
import java.util.*;
/**
 * A system that demonstrates the management of a Hospital.
 */
public class Hospital
{    private ArrayList patients;
     public Hospital ( )                                          1 mark
     {       patients = new ArrayList ();   }
     public void addPatient(Patient  p)                           0.5 mark
     {       patients.add (p) ;   }
    public void deletePatient(Patient p)                          0.5 mark
     {     patients.remove (p) ;  }
    public void displayAllPatient()                               1 mark
    {  Iterator it = patients.iterator ( ) ;
       System.out.println(" ALl Patients in the hospital   ...") ;
       while (it.hasNext () )
        {  Patient  pat = (Patient) it.next ( ) ;
            pat.display ( ) ;
        }
    }                                                             1.5 mark
   public void printKidPatients()
   {  Iterator it = patients.iterator ( );
      System.out.println(" Patients less than 12 years old ... " );
       while (it.hasNext () )
      {  Patient  pat = (Patient) it.next ( );
         if (pat.getAge() < 12)
            pat.display ( );
      }
   }
}
```

2- Class Patient

```
/**
 * A system that demostrates the management of a Hospital.
 * @author (Nadia)
 * @version (version 1  - 10/5/2006)
 */
public class Patient
{     // instance variables
      private String  name;
      int age;
      Date adminDate;                                             0.5 mark
      String disease;
      Date discharge;
```

```
      /**
       * Constructor for objects of class Hospital
       */
      public Patient()
      {     // initialise instance variables
            name = "";
            age = 0;
            disease = "";
            adminDate = new Date (3, 12, 2003);
            discharge = new Date (4, 2, 2003);
            }

      // methods to set and get patient's information

      public void setName(String na)
      {     name = na;  }
      public String getName()
      {     return name;      }
      public void setAge(int a)
      {     age = a;}
      public int getAge()
      { return age; }
      public void setDisease( String dis)
      {     disease = dis;}
      public void setAdimDate( Date d)
      {     adminDate = d;     }
      public Date getAdminDate()
      { return adminDate;}
      public void setDischarge( Date d)
      {     discharge= d;    }
      public Date getDischarge()
      { return discharge;}

      public void display()
      { System.out.println("Name = "+name+" age = "+age+
                           " Disease = "+disease);
      System.out.print(" Admission date= ");
      adminDate.printDate();
      System.out.print("   Discharge Date = ");
      discharge.printDate();
    }
}
```

0.5 mark

1.5 mark

1 mark

3- Class Date

```
/** class Date to set the date in month, year, day
 * @author (Nadia)
 * @version ( version 1 - 10/5/2006)
 */
public class Date
{    // instance variables
      private int month;
      private int year;
      private int day;
      // Constructor for objects of class Date
      public Date(int m, int y, int d)
      { month = m;
        year = y;
        day = d;
      }
      //a method is to print the date in the format; mm/dd/yyyy
      public void printDate()
      {  System.out.println (month+"/" +day + "/" + year);  }
}
```

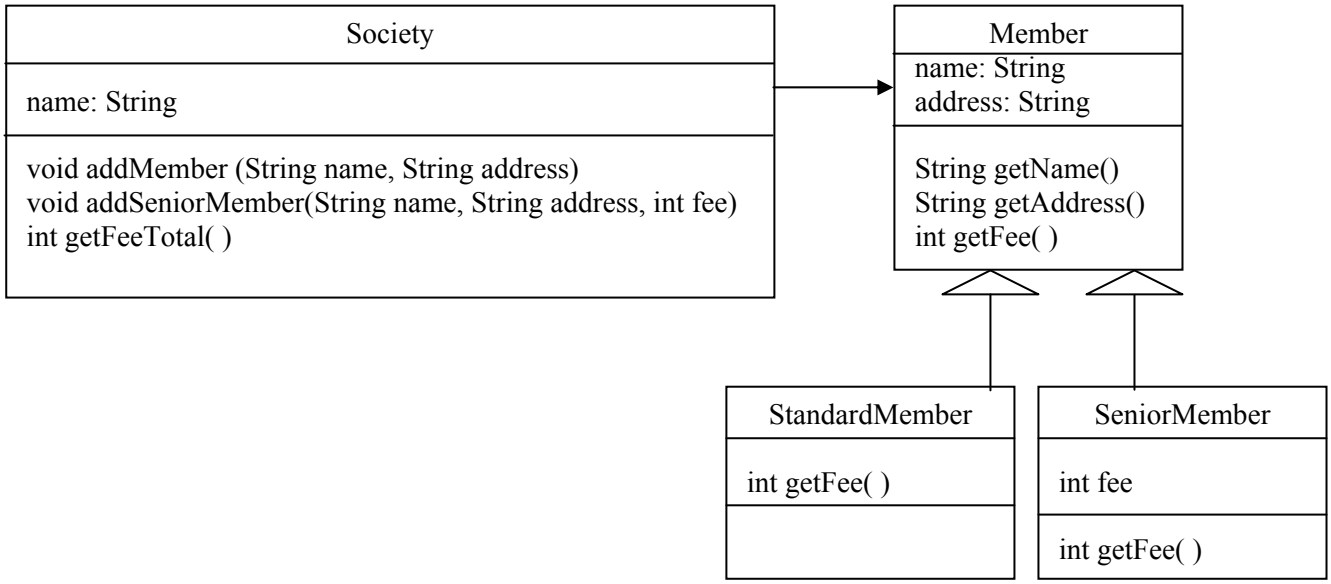
0.5 mark

1 mark

0.5 mark

6- Question 6 (12 marks)

Consider the following class diagram showing part of a program to manage the membership information for a professional society:



a) Class **Member** is an abstract class. Explain the role of an abstract class. **[1 marks]**

b) Write a Java version of class **Member** assuming it has this constructor:

```
public Member(String name, String address)
```

and that the method **getFee( )** is abstract. **[3 marks]**

c) Write a Java version of class **StandardMember** assuming it has this constructor:

```
public StandardMember (String name, String address)
```

and the standard membership fee is fixed at 30 JD. **[2 marks]**

d) Write a Java version of class **SeniorMember** assuming it has this constructor:

```
public SeniorMember(String name, String address, int fee)
```

where the membership fee is set when a **SeniorMember** object is created. **[2 marks]**

e) Write a Java version of class **Society** assuming it has this constructor:

```
public Society(String societyName)
```

[4 marks]

where **Society** has members of different types stored in an arraylist and **getFeeTotal** method that returns the total fees of all members in the society.

……

The aim of this question is to evaluate that students can solve familiar problems with ease and can make progress towards the solution of unfamiliar problems, and can set out reasoning and explanation in a clear and coherent manner. It is intended to students of more than intermediate level

One possible solution could be as follows:

(a) Abstract class is useful for making generalization. That is, one abstract method can be implemented in different views.	**1 mark**

(b) Abstract Class Member
`/**`
`  * Abstract class Member`
`  */`
`public abstract class Member`
`{  private String name;`
`  private String address;`
`  public String getName()`
`  { return name;  }`
	0.5 mark
	0.5 mark
	0.5 mark

`public String getAddress()`
`{ return address; }`
 	
`// abstract method`
`abstract public int getFee();`
`}`
	0.5 mark
	
	1 mark

(c) Class StandardMember
 	
`public class StandardMember extends Member`
`{ // Constructor for objects of class StandardMember`
`public StandardMember(String name, String address)`
`{ super (name, address); }`
`// Implement the abstract method`
`public int getFee()`
`{ return 30; }`
`}`
	0.5 mark
	
	1 mark
	
	0.5 mark

(d) Class SeniorMember
`//Class SeniorMemebr`
`public class SeniorMember extends Member`
`{ private int fee;`
`// Constructor for objects of class SeniorMemebr`
`public SeniorMember(String name1, String address1, int fees)`
`{ name = name1;`
`address = address1;`
`fee = fees;`
`}`
`public int getFee()`
`{ return fee; }`
`}`
	0.5 mark
	
	1 mark
	
	0.5 mark

(e) Class Society
`import java.util.ArrayList;`
`import java.util.Iterator;`
`public class Society`
`{ private String name;`
`private ArrayList society;`
`public Society``(String societyName)`
`{ name = societyName;`
`society = new ArrayList();`
`}`
`public void addMember``(String name, String address)`
`{ Member member = new StandardMember (name, address);`
`society.add(member);`
`}`
`public void addSeniorMemberString` `name, String address, int fee)`
`{ Member member = new SeniorMember (name, address, fee);`
`society.add(member); }`
`public void printAllMembers()`
`{ for (int i=0; i<society.size(); i++)`
`{ Member mb = (Member) society.get(i);`
`System.out.println("i = "+i+" Name= "+mb.getName()+`
`" Address = "+mb.getAddress()+" Fees = "+mb.getFee());`
`System.out.println("*********");`
`}`
`}`
`public int getFeeTotal()`
`{ int totalFees=0;`
`for (int i=0; i<society.size(); i++)`
`{ Member mb = (Member) society.get(i);`
`totalFees += mb.getFee(); }`
`return totalFees;`
`}}`
	0.5 mark
	
	1 mark
	
	0.5 mark
	
	0.5 mark
	
	0.5 mark
	
	1 mark

Philadelphia University
Faulty of Information Technology
Department of Software Engineering

Marking Scheme and Outline Solutions of the Final Exam

Module: Object-Oriented Paradigms (721112) **(Section 3)**
Exam Date: February 5, 2007
Lecturer: Dr. Nadia Y. Yousif

First Semester 2006 / 2007

……

There are three parts with five questions in this exam paper.
The following is the marking scheme and the set of outline solutions for the questions. It includes breakdown of the marks to each part of the question and the steps of the solution. It also describes the type of answer required to gain the stated marks.

1- Question 1 (10 marks)

Fill each of the following blanks with one of the words given in the list below: **[1 mark each]**
{dynamic, polymorphism, protected, interface, override, objects, inheritance, static, tokenizing, super}

1- ……. are used to model things from a problem domain.
2- The term …… refers to the fact that a variable can hold objects of different types.
3- In an ……. all methods are abstract.
4- The keyword …… is used to call methods in the superclass.
5- …….. is an access control keyword that permits access from the subclass, but denies access from anywhere not in a class or its subclass.
6- To cut up an input string into separate words is called ……..
7- …….. allows us to define one class as an extension of another.
8- A subclass can …….. a superclass method by declaring a method with the same signature as the superclass method.
9- The class variable is declared as ……..
10- The …… type of a variable is the type of the object that is currently stored in the variable.

……

The aim of the question in this part is to evaluate student's required minimal knowledge and understanding skills in the fundamentals of object oriented programming. It is intended to students of intermediate level.

The answer is:

1- objects	2- polymorphism	3- interface	4- super	5- protected	**1 mark each**
6- tokenizing	7- inheritance	8- override	9- static	10- dynamic	

……

Question 2 (10 marks)

(a) What is wrong with the following class definitions? How would you fix them? **[3 marks]**

```
class Employee
{ private String name;
  private int id;
  public Employee (String name, int id)
  {  this.name = name;        this.id = id;    }
  public String getName ( )         { return name; }
  public int getId ( )              { return id; }
 }
```

```
class  Manager extends Employee
{ private String department;
  public Manager (String name, int id, String department)
  { this.name = name; this.id = id; this.department = department;}
  public void getDepartment ( )   { return department; }
}
```

(b) What does this piece of code do? **[3 marks]**

```
Random rand = new Random ( );
String table = "";
String row = "";
for (int i = 1; i < 4; i++)
{   for (int j = 1; j < 4; j++)
    {   int number = rand.nextInt(50));
        row += number + " ";
    }
    table += row + "\n";
    row = "";
}
System.out.println(table);
```

(c) What is wrong with the following interface? Correct it. **[2 marks]**

```
public interface Figures
{   public void printMessage (String s)
     { System.out.println ("This figure is   " + s );    }
    public double area ( ) ;          // to calculate the area of the figure
}
```

(d) Design a class **Rectangle** that implements the corrected interface of section (c). **[2 marks]**

……

The aim of the question in this part is to evaluate student's required minimal knowledge and understanding skills in the fundamentals of object oriented programming. It is intended to students of intermediate level.

One possible solution is as follows:

(a) 1- The constructor in class Manager is initializing name and id which are private fields in the superclass. To correct this error, we could make the fields name and id protected or we define the constructor of Manager as follows: public Manager (String name, int id, String department) { super (name, id); this.department = department;} 2- Method getDepartment should return a String value not void.	**1.5 marks** **1.5 marks**
(b) The code will print a 3 x 3 table with random integer values between 0 and 50 e.g. 4 35 18 25 16 20 1 0 13	**3 mark**

(c) It has one method with implementation (i.e. not abstract) and in the interface all methods should be abstract.
The correction:
public interface Figures
{ public void printMessage (String s); // abstract method to print a message
public double area ( ) ; // to calculate the area of the figure
}
	1 mark
	
	1 mark

(d) public class Rectangle implements Figures { private double length; private double width;	**0.5 mark**

public Rectangle (int len, int wid)
{ length = len; width = wid; }
// Implementation of the abstract methods
public double area ( )
{ return (length * width) ; }
public void printMessage (String s)
{ System.out.println ("Rectangle " + s); }
}
	0.5 mark
	
	0.5 mark
	
	0.5 mark

……

Question 3 (8 marks)

(a) What is meant by polymorphism? Why do Java languages which support inheritance also support polymorphism? **[4 marks]**

(b) Assume we have three classes: **Person, Teacher,** and **Student. Teacher** and **Student** are both subclasses of **Person**.

Which of the following assignments are legal, and why? **[4 marks]**

Person p;
Teacher t;
Student s;
t = new Teacher ( );
p = t;
s = (Student) t;
s = (Student) p;
p = new Student ( );
t = new Person ( );
t = p;

……

This question is to test that students have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP to practice using inheritance and polymorphism. It is intended to students of intermediate level.

Solution:

(a) Polymorphism means that variables can hold objects of more than one type, i.e. they can hold objects of the declared type, or of subtypes of the declared type. Methods also can be polymorphic since the superclass and subclass can have a method with the same signature and the object type knows which method to call. Besides inheritance, Java also supports polymorphism since it allows dynamic typing. Inheritance, however avoids code duplication, allows code reuse, simplifies the code, and simplifies maintenance and extending. Whereas polymorphism allows variables to hold subtype objects and subtypes can be used wherever supertype objects are expected (substitution) to give more flexible implementation.	**2 mark** **2 mark**

(b)
t = new Teacher ( ); // legal because t is object of type Teacher and declared as Teacher
p = t; //legal – superclass object can refer to subclass object
 	
s = (Student) t; //illegal-compile-time error-incomputable type: Student is not a subclass of Teacher
 	
s = (Student) p; // illegal- run-time error: since p is a reference to t (object of Teacher) and there is no subclass/subclass relationship
 	
p = new Student ( ); //legal – now p becomes an instance of Student
t = new Person ( ); //illegal – class casting exception
t = (Teacher) p; //illegal – since p became of type Student and no subclass/subclass relationship
	1 mark
	
	1 mark
	
	1 mark
	1 mark

Question 4 (9 marks)

Write the definition and implementation of the class **Robot** that has one field called ***position*** of type class **Point** and three methods. The methods are:

- **moveLeft** that moves the robot from the current position to the left for d distance.
- **moveRight** that moves the robot from the current position to the right for d distance.
- **moveForward** that moves the robot forward for d distance.

Class **Point** has two fields: x and y of type int, which represent the coordinates of a point in x-y plane and two accessor methods to get the values of x and y.

……

This question is to test that students have some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar problems in OOP to practice using inheritance and polymorphism. It is intended to students of intermediate level.

One possible solution is:

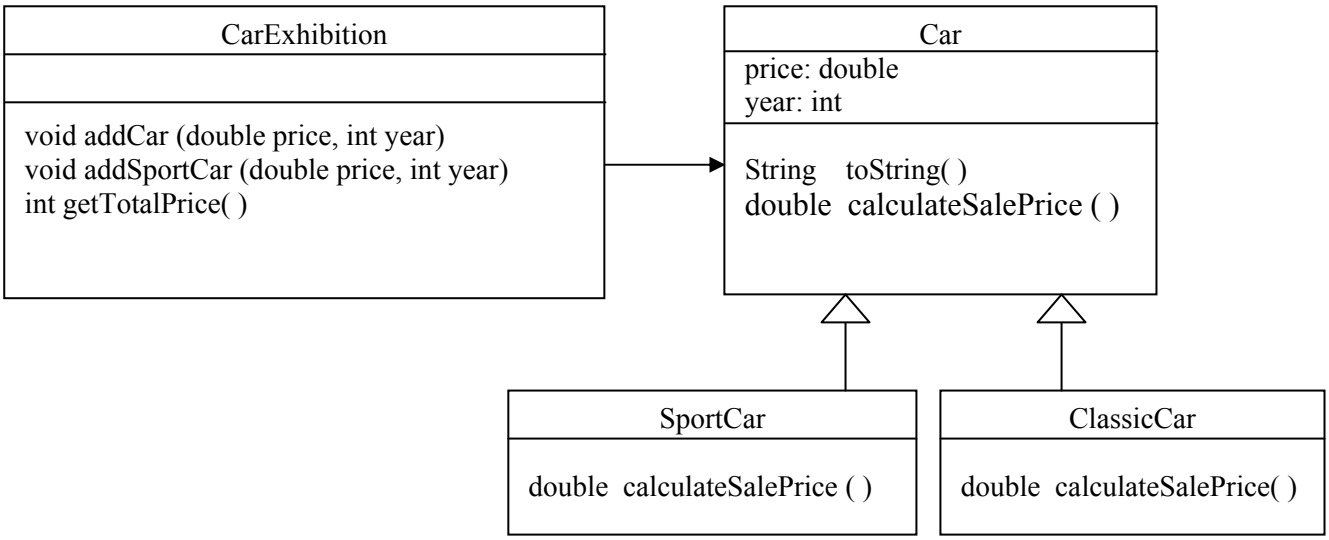
public class Robot
{ private Point position;
// Constructor for objects of class Robot
public Robot(int k, int m)
{ position = new Point (k,m); }
public int moveLeft(int d)
{ return (position.getX() - d); }
public int moveRight(int d)
{ return (position.getX() + d); }
public int moveForward(int d)
{ return (position.getY() + d); }
}
	1 mark
	
	1 mark
	1 mark
	1 mark
	1 mark

public class Point
{ private int x, y;
// Constructor for objects of class Point
public Point(int x1, int y1)
{ x = x1;
y = y1;
}
public int getX()
{ return x ; }
public int getY()
{ return y ; }
}
	1 mark
	
	1 mark
	
	1 mark
	1 mark

……

Question 5 (13 marks)

Consider the following class hierarchy where Class **Car** is the supper class and the classes **ClassicCar** and **SportCar** are two subclasses derived from **Car**.
Class **CarExhibition** contains a filed of type ArrayList that stores objects of type **Car**.



(a) Class **Car** is an abstract class. Explain the role of an abstract class. **[1 marks]**

(b) Write a Java version of class **Car** assuming it has this constructor:
public Car(double price, int year)
and that the method **calculateSalePrice ( )** is abstract. **[3 marks]**

(c) Write a Java version of class **ClassicCar** assuming it has this constructor:
public ClassicCar (double price, int year) **[2 marks]**
and that the method **calculateSalePrice ( )** returns 10,000 as the sale price of the car.

d) Write a Java version of class **SportCar** assuming it has this constructor:
public SportCar(double price, int year) **[3 marks]**

and that the method **calculateSalePrice ( )** calculates the sale price of the car as follow:
if year > 2000 then the sale price is 0.75 * its original price; if year > 1995 then the sale price is 0.5 * its original price; otherwise the sale price is 0.25 * its original price

e) Write a Java version of class **CarExhibition** assuming it has this constructor:
public CarExhibition( ) **[4 marks]**
where **CarExhibition** has cars of different types stored in an arraylist and **getTotalPrice** method that returns the total prices of all cars in the exhibition.

……..……..……..……

The aim of this question is to evaluate that students can solve familiar problems with ease and can make progress towards the solution of unfamiliar problems in object interaction, inheritance and abstract classes.

The outline solution is as follows:

(a)An abstract class provides sort of generalization. It can be reused for different concrete classes where, each concrete class that inherits from an abstract class can implement any abstract method in it.	**1 mark**
(b) **Car Class Definition** abstract public class Car { protected double price; protected int year; public Car(double price, int year) { this.price = price; this.year = year; } public String toString()	 **1 mark** **1 mark**

{ return ("Price = "+price+" Year = "+year); }
public abstract double calculateSalePrice ( ); //abstract method
}
	0.5 mark
	0.5 mark

(c) **ClassicCar Definition** public class ClassicCar extends Car { public ClassicCar(double price, int year) { super (price, year); } public double calculateSalePrice ( ) { return 10000; } }	 **0.5 mark** **0.5 mark** **1 mark**

(d) **SportCar Class Definition**
 	
public class SportCar extends Car
{ public SportCar(double price, int year)
{ super (price, year); }
public double calculateSalePrice ( )
{ double salePrice;
if (year > 2000)
salePrice = 0.75 * price;
else if (year > 1995)
salePrice = 0.5 * price;
else
salePrice = 0.25 * price;
return salePrice;
}
}
	
	0.5 mark
	0.5 mark
	
	2 mark

(e) **CarExhibition Class Definition**
 	
import java.util.ArrayList;
import java.util.Iterator;
 	
public class CarExhibition
{ private ArrayList cars;
public CarExhibition()
{ cars = new ArrayList(); }
 	
public void addCar (double price, int year)
{ Car cr = new ClassicCar(price, year); //Superclass/subclass relationship
cars.add(cr);
}
public void addSportCar (double price, int year)
{ cars.add(new SportCar(price, year)); }
 	
public double getTotalPrice()
{ double result = 0;
Iterator it = cars.iterator();
while (it.hasNext())
{ Car cr = (Car) it.next();
result = result + cr.calculateSalePrice();
}
return result;
}
}
	
	0.5 mark
	
	0.5 mark
	
	0.5 mark
	
	0.5 mark
	
	1.5 marks