King Fahd University of Petroleum and Minerals
College of Computer Science and Engineering
Information and Computer Science Department
First Semester 2006/2007 (061)
ICS 102 – Introduction to Computing

Major Exam 02 on Wednesday, 6th December 2006
Time: 90 minutes

Name: _____

ID#: _____

Please ***circle*** your section number below:

Section	01	02	03	04	05	06
Instructor Name	Sukairi	Sukairi	Krishna	Sebakhy	Krishna	AlShanyou
Day and Time	SM 8-8:50	SM 9-9:50	UT 9-9:50	UT 10-10:50	UT 1:10-2:00	UT 11-11:50

Question #	Maximum Marks	Obtained Marks
1	13	
2	14	
3	15	
4	12	
5	26	
Total	80	

Q. 1: Short questions.

(a) [**5 marks**] Write output of the following code in the space provided:

```
public class Q1
{
  public static void main(String[] args)
  {
    int x, y;
    String s1, s2;
    Date d1, d2;
    x = 10; y = x;
    s1 = "Hello"; s2 = s1;
    d1 = new Date(2006, 11, 6); d2 = d1;
    y = 20;
    s2 = "How are you? " ;
    d2.setYear(2000);
    System.out.println("x = " + x + " s1 =  " + s1 +  " d1 =  " + d1);
  }
}
```

(b) [**2 marks**] **MyClass** has a static method m1() and myObj is an object of **MyClass**. Is it legal to invoke m1 as follows: myObj.m1() ?

(c) [**2 marks**] If an instance variable is declared private, how can we access its value outside the class definition?

(d) [**2 marks**] Is the following code legal?

```
MyClass anobject = null;
anobject.set(100);
```

(e) [**2 marks**] What is an anonymous variable?

Q. 2: [2+2+3+4+3= **14 marks**] Design and implement a class to represent a user-defined type for *OlympicNation*. This class is used to represent name, gold, silver, bronze. Define

(a) appropriate instance variables,
(b) standard constructors (no-arguments, and all-arguments),
(c) standard accessor and mutator methods,
(d) addGold, addSilver, addBronze and getTotal methods, and
(e) equals method.

As always, you are required to follow standard conventions of java regarding visibility modifiers. You will be penalized otherwise.

Q. 3: [**3+2+5+5= 15 marks**] Design and implement a class to represent a user-defined type for *time*. This class has three instance variables: hours (0 to 23), minutes (0 to 59), and seconds (0 to 59). Define

(a) appropriate constructors (no/all/some-arguments),
(b) standard accessor and mutator methods,
(c) *addMinutes* (this method should result in 12:10 when we add 90 minutes to 10:40),
(d) *toString* method to display time in 12 hour format (time before 12:00 is displayed as it is with AM at the end, time after 11:59 is displayed appropriately with PM at the end).

As always, you are required to follow standard conventions of java, and overload methods if needed. You will be penalized otherwise.

Q. 4: [**1+2+3+3+3 = 12 marks**] Design and implement a class to represent a user-defined type for *Point*. This class has two instance variables: *x, y* (x determines the horizontal position of the point, and y determines the vertical position of the point). Define

(a) a constructor to initialize the two coordinates.
(b) accessor methods getX and getY

Also write a Java class called LineSegment, which represents a line segment as two Point objects. Define

(c) non-static method *length* (this method should return the length of the line segment),
(d) non-static method *slope* (this method should return the slope of the line segment),
(e) static method *slope*.

As always, you are required to follow standard conventions of java, and overload methods if needed. You will be penalized otherwise.

The formulae are:

Let slope $s = \frac{y2 - y1}{x2 - x1}$

Then *area* = $\sqrt{(y2 - y1)^2 + (x2 - x1)^2}$

Q. 5: [4+4+4+3+3+4+4 =**26 marks**] Design and implement a class to represent a user-defined type for *Employee*. This class has four instance variables: *name, salary, startingDate, expectedRetirementDate, leftDate*, and 3 static variables: *employeesJoined, employeesLeft, employeesRetired*. Define

(a) appropriate instance/static variables and initialize if needed,
(b) appropriate constructors (all-arguments, all-arguments except *leftDate* and copy constructor),
(c) appropriate accessor methods (including the ones for static variables),
(d) *resign* method (this method takes a date as input and assigns that to the instance variable *leftDate* and increments the counter *employeesLeft* by 1),
(e) *retire* method (this method takes a date as input and assigns that to the instance variable *leftDate* and increments the counter *employeesRetired* by 1),
(f) *incrementSalary* method (this method takes increment-percentage as input),
(g) *extendRetirementDate* method (this method takes an integer as input and changes the year filed of *expectedRetirementDate*).

You are required to avoid **privacy leak**. You will be penalized otherwise. Assume that you have **Date** class with copy constructor.