KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS

Information and Computer Science Department

2011/2012 First Semester (Term 111)

ICS102: Introduction to Computing (2-3-3)

FINAL EXAM

Saturday, January 14th 2012, 07:30 AM – 09:30 AM

120 MINUTES

Student Information

Name:								
ID:								

Circle your section

Bagais	SM 8:00 – 8:50 am	
Garout	SM 10:00 – 10:50 am	
Al-Turki	SM 13:10 – 14:00 pm	SM 9:00 – 9:50 am

Question No.	Maximum Score	Score
01	15	
02	25	
03	20	
04	40	
TOTAL	100	

Question 1 (15 points):

Choose the correct answer in the following questions:

1. The error in this code is:

```
int [][]x = {{1, 2}, {3}};
int [] y = {5, 6, 7};
x[1] = y;
```

   a. The rows in x have different sizes

   b. Incompatible assignment for x[1] = y

   c. x[1] is out of bound

   d. There is no error in the code

2. The error in this code is:

```
String [][] a = {{"a", "b", "c"}, {"d"}, {"e", "f"}};

a[1][1] = "true";
```

   a. The rows have different sizes

   b. The element a[1][1] is out of bounds

   c. Incompatible type for assignment of a[1][1] = "true"

   d. There is no error in the code

3. The error in this code is:

```
public static int sum(int c, int d){
   int result;
   result = c + d;

}
```

   a. result variable is not in the parameter list

   b. Missing return statement

   c. The parameters c and d are separated by a comma not semicolon

   d. c and d are not initialized inside the method

4. Given the following method header:

```
public static void meth(boolean b)
```

Which of the following is a legal call?

   a. `meth()`

   b. `meth("Java")`

   c. `meth(true)`

   d. `double d = meth(false)`

5. Given the following method header

```
public static String meth(String a, char c)
```

which of the following is a legal call?

```
a. double x = meth("word", 'o')
b. String s = meth('o', "word")
c. String s =  meth("word", 'o')
d. char c = meth("word", c)
```

6. Given the following code:

```
double [][] a = {{7.0, 8.5, 9.0}, {10.5, 11.0, 12.5}};
double [] b = a[0];
double c = b[1];
```

which of the following is true?

a. a[0] cannot be assigned to b

b. c and a[0][1] have the same value

c. b[2] and a[1][2] have the same value

d. None of the above is true

7. If the method: `public float aMethod(float a, float b)` is in a class, which of the following is legal overloading:

```
a. public double aMethod(float a, float b)
b. public float aMethod(float a2, float b2)
c. public float aMethod(double a, double b)
d. public int aMethod(float a, float b) throws Exception
```

8. Which of the following may overload a method whose signature is `void xyz(float f)`:

```
a. float xyz(float f)
b. void xyz2(float f)
c. int xyz(float f)
d. void xyz(short f)
```

9. Given the class:

```
class A{
  public static void main(String [] args){
    A o = new A();
    o.printResult(1, "abc", 4);
  }
  private void printResult(long a, String b, int c){
    System.out.println(a + b + c);
  }
}
```

Which of the following statements is true?

a. the code will not compile because a long and a string cannot be added

b. the code will compile and run and will display 1abc4

c. the code will not compile because there is no constructor in the class

d. the code will not compile because the `printResult` method is private

10. Consider the following class

```
class C{
  private int x;
  public C(int i){
    x = i;
  }
  public boolean equals(C c){
    return x == c.x;
 }
}
```

Which of the statements is true about the objects?

```
C c1 = new C(1);
C c2 = new C(2);
C c3 = new C(1);
C c4 = c3;
```

a. `c1` and `c3` refer to the same object

b. the variable `x` has the same value for objects `c4` and `c1`

c. `c1 == c3` is true

d. `c2.equals(c4)` is true

Question 2 (25 points):

Find the output of the following Java code

I)

```
char [] b = {'A', 'B', 'C', 'D', 'E'};
char [] a = {'F', 'G', 'H', 'I', 'J'};
int x = 3;
a[--x] = b[0];
a[2 * x] = b[1];
a[++x] = b[2];
for(int i = 0; i < a.length; i++)
  System.out.print(a[i] + " ");
```

OUTPUT

II)

```
public static void main(String [] args){
  String  a = "Java";
  a = m2(m1(a));
  System.out.println(a);
}
public static String m1(String a){
  return a.substring(0, 2);
}
public static String m2(String a){
  return a.toUpperCase();
}
```

OUTPUT

IV)

```
public static void main(String []args){
  int [] a = {10, 20, 30};
   if(m1(a))
     m2(a, 0);
   else
     m2(a, 1);
   for(int i = 0; i < a.length; i++)
     System.out.println(a[i] + ” “);
}
public static boolean m1(int [] a){
   return a[0] == a[1] / 2;
}
public static void m2(int [] a, int b){
    a[b] = 1;
}
```

OUTPUT

V)

```
public static void main(String [] args){
double [] a = {1, 2, 3};
m1(a);
System.out.println(a[0]+"  "+a[1]+"  "+a[2]);
}
public static void m1(double b[]){
  m2(b[1]);
   b[1]++; b[2] += 5;
}
public static void m2(double  c){
   c = c * c;
  System.out.println(c);
}
```

OUTPUT

VI)

```
class C{
  private double i;
  private String s;
  private static double j = 5;
  public C(double i, String s){
    this.i = i;
    this.s = s;
  }
  public void setS(String x){
    s = x;
  }
  public void setIJ(double x){
    i = x;
    j += 2 * i;
  }
  public double getI(){
    return i;
  }
  public String toString(){
    return s + "  " + i + " " + j;
  }
  public static void main(String [] args){
    C a = new C(2.1, "abc");
    C b = new C(2.1, "def");
    b.setS("abc");
    System.out.println(a == b);
    a.setIJ(5);  b.setIJ(a.getI());
    System.out.println("a = " + a);
    System.out.println("b = " + b);
}
}
```

OUTPUT

Question 3: (20 points)

Write a static method that takes as a parameter a 2-D array quiz (Each row has 6 quiz grades) that has the quiz grades of ICS102 students and returns a 1-D array that has the sum of the best 5 quizzes for each student. **Do not** write the main method.

Question 4: (40 points)

A. Suppose we want to implement a class, called `“Account”`, which represents a bank account. Complete the implementation of this class `“Account”` as instructed by the enclosed comments below.

```
class Account {
 /* Three instance variables:
  * 1) 'id': a number that uniquely identifies the account,
  * 2) 'owner': the name of the customer to whom this account belongs
  * 3) 'balance': the amount in Riyals that is available in the account
  */

--------------------------------------------------------------------------

--------------------------------------------------------------------------

--------------------------------------------------------------------------

  /* A three-argument constructor to initialize all instance variables
   * with values given as parameters
   */

--------------------------------------------------------------------------

--------------------------------------------------------------------------

--------------------------------------------------------------------------

--------------------------------------------------------------------------

/* A two-argument constructor to initialize the 'id' and 'owner' fields
 * with values given as parameters, and the 'balance' field with the
 * value zero
 */
--------------------------------------------------------------------------

--------------------------------------------------------------------------

--------------------------------------------------------------------------

--------------------------------------------------------------------------

--------------------------------------------------------------------------

--------------------------------------------------------------------------

--------------------------------------------------------------------------

--------------------------------------------------------------------------
```

```
 /* A copy constructor
  */
------------------------------------------------------------------------

------------------------------------------------------------------------

------------------------------------------------------------------------

------------------------------------------------------------------------

/* An accessor method that returns the balance
 */
------------------------------------------------------------------------

------------------------------------------------------------------------

/* A mutator method that sets the 'owner' field to a string value given
 * as a parameter, but only if the given value is not null and not the
 * empty string
 */
------------------------------------------------------------------------

------------------------------------------------------------------------

------------------------------------------------------------------------

------------------------------------------------------------------------

 /* A method deposit to deposit (add) to the balanace a positive
  * amount 'a' of money given as a parameter (only if 'a' is positive)
  */
------------------------------------------------------------------------

------------------------------------------------------------------------

------------------------------------------------------------------------

------------------------------------------------------------------------

------------------------------------------------------------------------
```

```
/* A method withdraw to withdraw (subtract) from the balanace
 * positive amount 'a' of money given as a parameter (only if 'a' is
 * positive and the account has enough money
 */

------------------------------------------------------------------------

------------------------------------------------------------------------

------------------------------------------------------------------------

------------------------------------------------------------------------


/* An equals method, which returns true if the id and owner fields
 * of this account object and the account object given as a parameter
 * have identical values
 */

------------------------------------------------------------------------

------------------------------------------------------------------------

/* A toString method that returns a string containing the account's
 * id, owner, and balance field values
 */

------------------------------------------------------------------------

------------------------------------------------------------------------
```

B. Now we want to implement a class, called “Bank”, representing a bank that manages customer bank accounts. Complete the implementation of this class “Bank” as instructed by the enclosed comments below.

```
class Bank {

  /* A constant 'MAX_ACCOUNTS' that has the value 100
   */

------------------------------------------------------------------------

  /* Two instance variables:
   * 1) 'accounts': an array of Account objects
   * 2) 'numOfAccs': the actual number of account objects in the array
   */
------------------------------------------------------------------------

------------------------------------------------------------------------
```

```
/* A no-argument constructor to initialize the instance variables to
 * appropriate initial values (the size of the array 'accounts' is
 * the constant MAX_ACCOUNTS)
 */
```

```
/* A method addAccount that adds an account object given as a
 * parameter to the list of accounts if possible
 */
```

```
/* A method getAccount that returns the account object in 'accounts'
* at index k (if possible), which is given as a parameter
*/
```

```
  /* An accessor method 'getNumOfAccs' that returns the current number
   * of accounts
   */
```

```
    /* A method 'transfer' that transfers an amount of money 'a' from
     * the account at index 'from' to the account at index 'to' (using
     * the methods 'withdraw' and 'deposit' defined in the Account
     * class). 'a', 'from', and 'to' are all given as input parameters
     * to the method, and they all must have valid values.
     */
------------------------------------------------------------

------------------------------------------------------------

------------------------------------------------------------

------------------------------------------------------------

------------------------------------------------------------

------------------------------------------------------------

------------------------------------------------------------

------------------------------------------------------------
```

C. Finally, to test these classes, complete the implementation of the following simple class “BankDemo” as instructed by the enclosed comments below.

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

      /* Create an instance 'b' of the class Bank
       */
------------------------------------------------------------

      /* Add two accounts to the bank 'b':
       * 1) An account with id: 1010, owner: "Ali", balance: 20,000
       * 2) An account with id: 1022, owner: "Wael", balance: 0
       */

------------------------------------------------------------

------------------------------------------------------------

      /* Transfet 5000 from the first account to the second account
       */
------------------------------------------------------------

      /* Display the details of all the accounts in the bank 'b'.
       * Assume you don’t know how many accounts you have in the bank.
       **/
------------------------------------------------------------

------------------------------------------------------------

    }
}
```