ESC101 : Fundamental of Computing

Lab Test for 12th November 2008

Instructions:

1. The duration of the test is 3 hrs (from **2:00 pm to 5:00 pm**).
2. **Directory Structure:** Create a directory and name it with your roll number. e.g. If your roll number is $Y8001$, the directory should be named $Y8001$ (`Y` should be upper case). Inside this directory, create three directories named **One**, **Two** and **Three** corresponding to the programs for the three problems of this test. In directory **One**, create file *AmicableNumbers.java* for the java program for first problem. In directory **Two**, create files *Matrix.java* and *Matrix_example.java* for the second problem. In directory **Three**, create file *Partition.java* for the java program of the third problem.
3. Use of good coding practices (indentation, use of methods, and proper naming of variables and methods) carries weightage.

Problems:

1. **Amicable Numbers :** (marks = 10)
   A positive integer d is a *proper divisor* of integer n if d *divides* n and $d \neq n$. e.g. The proper divisors of 12 are $\{1, 2, 3, 4, 6\}$.

   Amicable numbers are two different numbers so related that the sum of the proper divisors of one of the numbers is equal to the other. e.g. $(220, 284)$ are Amicable Numbers because:

   The set of proper divisors of 220 is $\{1, 2, 4, 5, 10, 11, 20, 22, 44, 55, 110\}$, and $1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110 = 284$.

   The set of proper divisors of 284 is $\{1, 2, 4, 71, 142\}$, and $1 + 2 + 4 + 71 + 142 = 220$.

   Other examples of Amicable numbers are $(1184, 1210)$ and $(2620, 2924)$.

   Write a JAVA program that takes two positive integer n_1 and n_2 from command-line and prints whether they are amicable numbers.

2. 2×2 **Matrix :** (marks = 15)
   You have to design a JAVA class Matrix that represents a 2×2 matrix. The elements of such a matrix — $a_{11}, a_{12}, a_{21}, a_{22}$ should be of type *integer*.

   Write two constructors for this class. One constructor does not have any parameter, and it initializes the matrix to the null matrix $\{\{0, 0\}, \{0, 0\}\}$. The second constructor takes four arguments of type *integer* and initializes the elements of the matrix to the same value as these arguments.

   You should design and implement the following four *non-static* methods:

   (a) **print**: Prints the elements of *this* matrix as: $\{\{a_{11}, a_{12}\}, \{a_{21}, a_{22}\}\}$.

   (b) **add**: Takes an object of the class Matrix as argument, say M and adds matrix M to matrix *this*. Note that the contents of matrix *this* may change accordingly by the execution of this method.

   (c) **multiply**: Takes an object of the class Matrix as argument, say M and multiplies the matrix *this* by matrix M. Note that the order of multiplication is *this* $*M$. This is important since matrix multiplication is not commutative. Also note that the contents of matrix *this* may change accordingly by the execution of this method.

   (d) **findDeterminant**: Returns the determinant of *this* matrix.

Now, write a *main* method (in the file *Matrix_example.java*) that tests the class that you have written. The program takes the elements of a matrix from command line as input. Let us denote this matrix as A. Then the program create one more object B of class Matrix and initialize so that $B = \{\{1, 1\}, \{3, 4\}\}$.

The program should perform the following actions sequentially.

(a) print the matrix A.

(b) print the matrix A after invoking method **add**(B) on it.

(c) print the matrix A after invoking method **multiply**(B) on it. (Note that this step is executed after the previous step which invokes **add** on A).

(d) print the determinant of the final matrix A.

3. **Partitions of a number with restrictions** (marks=15)
We discussed the problem of enumerating partitions of a positive integer in tutorial. We also discussed and solved its variants in the practice problems. The current problem which you have to solve is the partitioning problem with some more restrictions. Please understand it clearly, and then solve it accordingly.

   Given two positive integer n and k, write a java program to enumerate all those sets consisting of positive numbers satisfying the following constraints.

   - Sum of all the numbers of a set is n
   - Difference between each pair of numbers in the set is at least k

   The program must read the values of n and k from command line (first argument should be n and the second should be k).

   For an example, if $n = 7, k = 2$, the output should be the sets

```
{7}
{1,6}
{2,5}
```

   For $n = 13, k = 3$, the output should be the sets

```
{13}
{1,12}
{1,4,8}
{2,11}
{3,10}
{4,9}
{5,8}
```

   Please note that the order does not matter while prining the elements of a set.

Hint : Please note that for $k = 1$, the above problem is the same as one of the practice problem whose solution was posted one week back.