CS207A: Data Structures and Algorithms (Module #3)
Assignment #1

```
Max marks:  55
Due on/before:17.00, 2-July-2017                                  23-June-2017
```

Note: All questions will be graded using an automated judge so please ensure that your programs follow the input-output instructions exactly.

1. (a) For the first coin game discussed in class implement a function that prints out the coins that are chosen. Your program should read in input from a file called INPUT which has the following structure.

   1^{st} line: n - the number of coins in the sequence.
   2^{nd} line: $c_1\ c_2\ \ldots\ c_n$
   It should write its output in a file called OUTPUT where the chosen coins c_i are printed separated by a single space and terminated at the end by a newline.

   (b) For the Simple Knapsack Problem write a program that finds the set of items that goes into the knapsack for which the value is a maximum. The program should read its input from a file called INPUT which has the following structure:

   1^{st} line: n - the number of items.
   2^{nd} line: $c_1\ v_1\ c_2\ v_2\ \ldots\ c_n\ v_n$
   3^{rd} line: C - capacity of the knapsack.

   It should write the output into a file called OUTPUT with the following format:

   $c_{i_1}\ c_{i_2}\ \ldots\ c_{i_m}$
   where $c_{i_1} \leq c_{i_2} \leq \ldots \leq c_{i_m}$

   [10,10=20]

2. For the optimal binary search tree (BST) we assumed that the search key was always present in the tree. In general this is not true. The search key may not be present in the tree. We can model this situation as follows:

   For each i, $i = 1..(n-1)$, assume we have probability q_i which gives the probability that a search key, say k lies between k_i and k_{i+1}, that is $k_i < k < k_{i+1}$ - remember the keys $k_1, \ldots, k_n$ are in increasing order.In addition, let q_0 be the probability that a search key $k < k_1$ and q_n be the probability that $k > k_n$. So, in addition to probabilities $p_1, \ldots, p_n$ for a *hit* in the tree we have probabilties $q_0, \ldots, q_n$ for a miss in the tree. Of course, we have $\sum_{i=1}^{n} p_i + \sum_{j=0^n q_j} = 1$.

   Implement the optimal BST algorithm to find the optimal BST and the minimum expected value of the number of comparisons. The file INPUT in which all input is given should have the following structure:

   1^{st} line: n - number of keys.

2^{nd} line: $p_1\ p_2\ \ldots\ p_n$
3^{rd} line: $q_0\ q_1\ \ldots\ q_n$

The output should be in file OUTPUT with the following format:

1^{st} line: x - the expected value for the number of comparisons in the optimal binary tree.
2^{nd} line: BST - written as a nested linear structure using round brackets. The tree should be written recursively as follows: ($root$(left-BST)(right-BST)).

[20]

3. Given an $m \times n$ binary array (or matrix) using dynamic programming to find the largest contiguous square containing only zeroes in the matrix/array. This is useful in locating large free area on a computer screen or land for construction. The input will be in file INPUT and structured as follows:

   1^{st} line: m n x - where m, n is row size and column size, x is the number of 1s in the matrix/array.
   2^{nd} line: $i_1\ j_1\ i_2\ j_2\ \ldots i_x\ j_x$ - locations of the x ones in the array or matrix.

   The output should be in file OUTPUT and should give (i, j) coordinates of the top left and bottom right 0 entries without brackets. For example, if the square is $(2, 3)$ till $(7, 8)$ the output should be:
   2 3 7 8. [15]