**EE4371 DSA Final Exam – Nov 9, 2018 – 30 marks**

| |
|---|
| Parts of the exam to be submitted as programs are marked **TH**. All other parts are to be attempted and submitted in class. Answers without explanations, and algorithms without run-time analysis, will not be accepted. |

Roll No.: ______________________ Name: ______________________

1. (2 marks) An undirected graph can be *2-coloured* if we can divide the vertices of the graph into two subsets $S_1$ and $S_2$, such that all edges have one end in $S_1$ and the other end in $S_2$ . Give an algorithm to determine if a graph can be 2-coloured.

2. (2 marks) Given a numerical sequence such as `[10,30,40,20,50]`, we can form *subsequences* of smaller size. For example, $S_1 =$`[10,40,50]`, $S_2 =$`[30,40,20]`, $S_3 =$`[10,30,20]` are all valid subsequences, but `[20,30,40]` is not, because the numbers are not in the same order as the original sequence. Also, $S_1$ is an *increasing* subsequence, but $S_2$ and $S_3$ are not, because the values are not in increasing order.

   Find the (or any one if more than one exists) longest increasing subsequence of the sequence

   ```
   [0,8,4,12,2,10,6,14,1,9,5,13,3,11,7,15]
   ```

3. You are in charge of scheduling an *all-play-all* tournament. That is, if we have $N$ teams numbered 0 to $N-1$ (assume $N$ is even), we need to schedule $N/2$ matches per day, over a period of $N-1$ days, such that every team plays every other team exactly once.

   For example, if we have 4 teams, then a possible schedule is:

   ```
   Day 1:   0 vs 1, 2 vs 3
   Day 2:   0 vs 2, 1 vs 3
   Day 3:   0 vs 3, 2 vs 1
   ```

   (a) (1 mark) Draw up a 7-day schedule for a tournament featuring 8 teams.

   (b) (1 mark) If you decide to do a backtracking based algorithm to solve this, estimate the complexity of the resulting algorithm.

   (c) (4 marks) Construct a matrix $M$ of size $N-1 \times N-1$, where the first row has the numbers 1 to $N-1$ in sequence, and subsequent rows are rotated (you have to decide how to rotate) versions of the first row. Show how you can use this matrix to solve the scheduling problem. What is the complexity of the resulting algorithm?

   (d) (3 marks) **TH** Write a program that, given a number $N$ (assume even) will print out a schedule for the entire tournament to complete in $N-1$ days.

4. On the map of TN already given, you will be given a set of $K$ node IDs ($K \ll N$ where $N$ is the total number of nodes in the map) corresponding to cities. You need to specify a *tour*: a sequence of the given nodes as well as any intermediate nodes not in this list, such that you start from the first node, visit each node in the sequence specified, and finally return to the starting point.

   The objective is to find the tour with minimum total length (so you can ignore speed, capacity etc.).

   (a) (2 marks) Briefly describe an algorithm to solve this problem (with analysis)

   (b) (5 marks) **TH** Implement the code for this - you will be given sample inputs as specified later on the Moodle.

5. It is placement season, and the realization has finally dawned on you that you can make much more money in the financial sector than by making chips (or worse, teaching people how to make chips). So you decide to put your DSA course to work solving an investment problem, a small example of which is given here: Assume you have Rs.2,000 to invest, and have 2 possible places to invest, and a time period of 2 years after which you want to maximize your return. The return rate for each investment is given in the table below:

| Year | Inv 1 | Inv 2 |
|---|---|---|
| 1 | $r_{11}$ =1.10 | $r_{21}$ =1.09 |
| 2 | $r_{12}$ =1.10 | $r_{22}$ =1.12 |

This means that if you invested Rs.1,000 in Inv 1 in year 1 (call this $I_{11}$), you would have Rs.1,100 ($= I_{11} \times r_{11}$) at the end of year 1.

For year 2, you have two options:

- Leave your investments as they are and let them grow in the same places, but as per the updated rates for the new year.
- Change how much is invested in each investment, but then you have to pay a lumpsum fee of Rs.100 in order to make the changes (no matter how small or how many changes, the amount is constant at Rs.100).

For example, two possible investment strategies are shown below:

| **Strategy 1** | | | | | | **Strategy 2** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Year $i$ | $I_{1i}$ | $I_{2i}$ | Invested | Fee | Total | Year $i$ | $I_{1i}$ | $I_{2i}$ | Invested | Fee | Total |
| start 1 | 1000 | 1000 | 2000 | 0 | 2000 | start 1 | 1000 | 1000 | 2000 | 0 | 2000 |
| end 1 | 1100 | 1090 | - | - | 2190 | end 1 | 1100 | 1090 | - | - | 2190 |
| start 2 | 1100 | 1090 | 2190 | 0 | 2190 | start 2 | 0 | 2090 | 2090 | 100 | 2190 |
| end 2 | 1210 | 1220.8 | - | - | **2430.8** | end 2 | 0 | 2340.8 | - | - | **2340.8** |

Assume you are given the initial sum available for investment ($S$), the fee that will be charged each year that you switch ($f$), and all the $r_{ij}$ values (for all investments and all years). In other words, you have perfect knowledge of the future of the investment business.

(a) (2 marks) For the problem above (Rs.2,000 initial, Rs.100 fee per change), find the optimal allocation of funds to investments over the course of 2 years to maximize the return. Repeat for the case where your initial capital to be invested is Rs.20,000 in total, but the fee remains the same.

(b) (3 marks) Show how to formulate the problem in the general case so that you have the "optimal sub-structure" property, that can be used to implement a recursive algorithm.

(c) (5 marks) **TH** Given a table of $r_{ij}$ values (return on investment $i$ during year $j$), a fee $f$ for switching investments from one year to the next, and a total initial sum of money $S$ that you are allowed to spend in year 1, implement an algorithm that will decide the $I_{ij}$ values to maximize the return at the end of $N$ years.