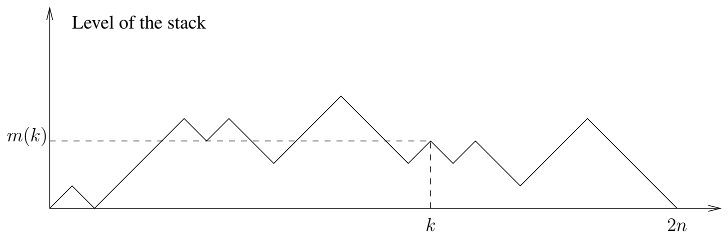
UE Mathematics for Computer Science

Final exam December 18, 2008 (3 hours)
This exam is closed book. Neither documents nor calculators are allowed.
Use separated sheets for problem 1 and problem 2.

Counting stacks *(10 points)*

Consider a stack with the two primitives *push* and *pop*. An execution of a program consists in n operations *push* and n *pop* which could be interleaved. The execution is represented by a *mountain*, the function $m(k)$ that gives the level of the stack after k operations.



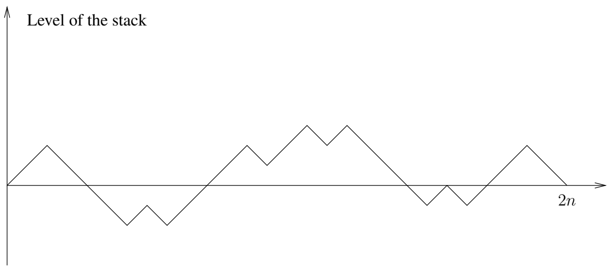
Denote by M_n the number of *mountains* with n *push* (up-stroke) and n *pop* (down-stroke) operations and set $M_0 = 1$.
Question 1.1 : Small n cases

For $n = 1, 2, 3$ give the possible *mountains* and deduce M_1, M_2, M_3.

Question 1.2 : Graph formulation

Propose a formulation of the problem as a path problem in a graph.

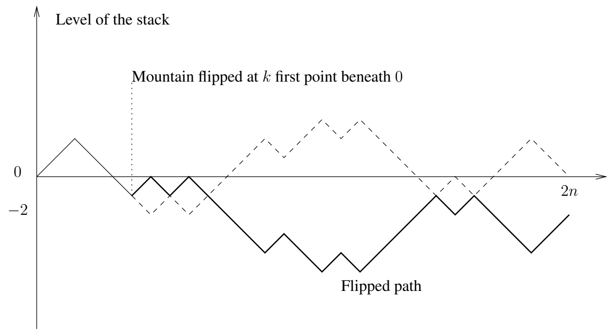
An extended *mountain* of length $2n$ with n up-strockes and n down-strockes allows to be under the sea level (bad mountains):



Question 1.3 : Extended *mountains*

Compute the number of extended mountains with length $2n$.

The flip operation consists in exchanging all the slopes after the first passage below 0:



Question 1.4 : Flipped mountains

Show that the set of bad *mountains* is in bijection with the set of *mountains* with $n-1$ up-strokes and $n+1$ down-strokes.

Question 1.5 : Computation

Prove that

$$M_n = \frac{1}{n+1}\left(\begin{array}{c} 2n \\ n \end{array}\right) = \frac{2n!}{(n+1)!n!}.$$

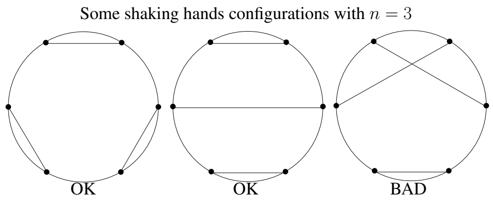
Question 1.6 : Recurrence relation

Show directly on mountain diagrams that the M_n numbers satisfy the recurrence equation:

$$M_n = M_0 M_{n-1} + M_1 M_{n-2} + \cdots + M_{n-1} M_0.$$

Question 1.7 : Random mountains

From a uniform random generator *random()* propose an algorithm that generates mountains uniformly.



Question 1.8 : Shake hands

Suppose that $2n$ persons are seated around a table, how many ways could they shake hands without crossing ?

Question 1.9 : Balanced parenthesis

Explain why M_n is the number of balanced parenthesis expressions with n opening and n closing parenthesis.

Traveling salesman problem *(10 points)*

Let us consider a salesman who wants to organize the visit of his clients as best as possible. It consists in visiting them in various cities with his vehicle. Of course, he must go in every cities and his objective is to minimize the total distance done with his vehicle. The only information he has is the list of the cities and a map with all inter-cities distances, and some old souvenir from his courses in graph theory...

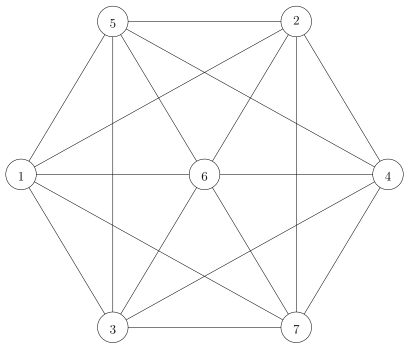
More formally, the input of the problem are n cities, denoted by $v_1, \ldots, v_n$, the distance between city i and j is denoted by $d_{i,j}$. TSP (traveling salesman problem) is to determine an optimal tour (of minimum length), that visits each city exactly once.

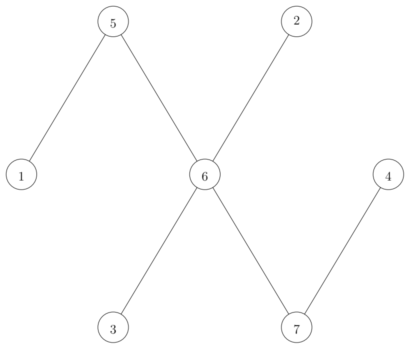
Question 2.1 :

Formalize this problem as a graph theoretical problem.

Let us now construct an efficient solution for this problem. It is well-known that TSP is a hard problem, that means we can not expect a polynomial time algorithm which solves exactly the problem. Let us construct a good solution (not too far from the optimal) in polynomial time. It procceds in three phases.

Phase 1. Determine a minimal weight spanning tree T^*. Let us denote by ω_G the weight of a graph G (sum of the weight on his edges).





Question 2.2 :

Show that ω_{T^*} is a lower bound of the value of the optimal tour.

Phase 2. Consider now the set O of the vertices of T^* whose degrees are odd.

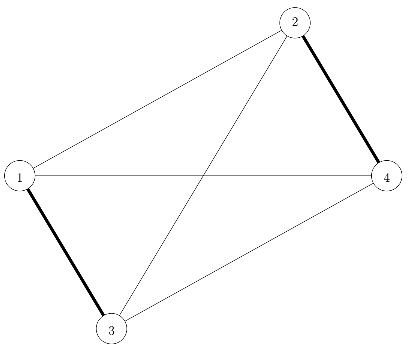
Question 2.3 :

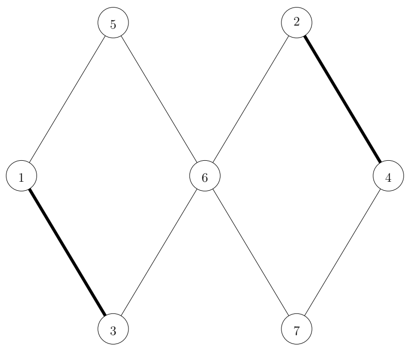
Show that the cardinality of O is even.

Let us construct now the perfect matching C^* of minimum weight between these vertices in O.

Question 2.4 :

Show that $2\omega_{C^*}$ is a lower bound of the value of the optimal tour.





Phase 3. Let us now consider the graph $T^* \cup C^*$.

Question 2.5 :

Show that all his vertices have an even degree.

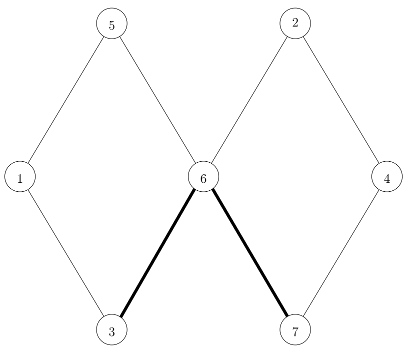
We are going to transform this graph in the following way: We will replace some edges by shortcuts. while it exists a vertex of degree greater than 4, we remove two of these consecutive edges and replace them by the opposite edge of this triangle without disconnecting the graph (as it is shown in the figure below).

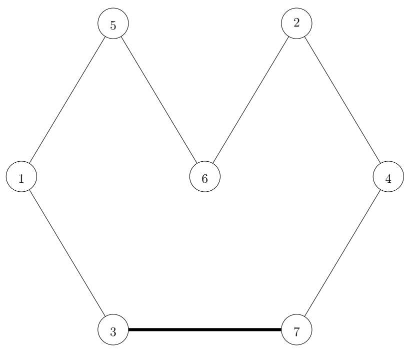
Question 2.6 :

Show that this process leads to a feasible tour.

Question 2.7 :

Show that such transformations do not increase the total weight.





Question 2.8 :

Show that the value of such a tour is lower than $\frac{3}{2}$ of the optimal tour.