Introduction to Computer Science
HW #4
Due: 2018/05/23

Homework Rules:

Hand-written homework can be handed in **before lecture starts**. Otherwise, you may contact the TA in advance and then bring the hardcopy to the TA in MD-631 (please send e-mail in advance).

As for the programming part, you need to upload it to CEIBA before the deadline. The file you upload must be a **.zip** file that contains the following files:

README.txt

HW04_b04901XXX (a folder that contains all .cpp & .h as required),

1. Do not submit executable files (.exe) or objective files (.o, .obj). Files with names in wrong format will not be graded. You must **remove any system calls**, such as system ("pause"), in your code if any.
2. In README.txt, you need to describe which compiler you used in this homework and how to compile it (if it is in a “project” form).
3. In your .cpp files, we suggest you write comments as detailed as you can. If your code does not work properly, code with comments earns you more partial credits.

Chapter 6 Review Problems (5% each):

21, 40, 41, 51, 58.

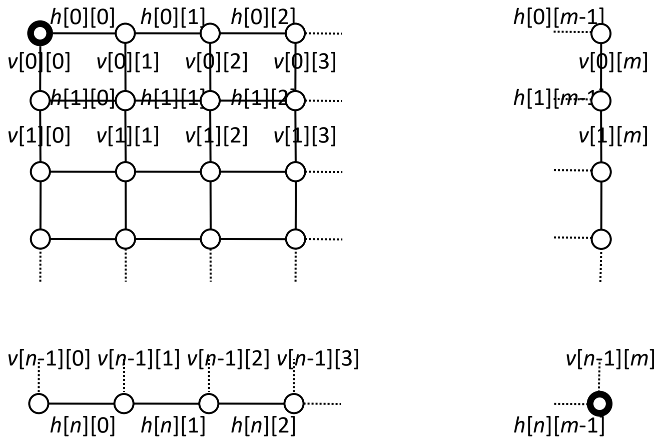
Introduction to Computer Science
HW #4
Due: 2018/05/23

Programming Problem I (50%): Shortest Path

You may use either python or C/C++ for this homework;

however, you need to deal with the “read” by yourself.

Consider a graph of *m* by *n* grids:



Use dynamic programming to find the shortest path from the upper-left node to the lower-right node with **moves in all four directions**. Costs of vertical edges are stored in an 2D array *v*[0..*n*-1][0..*m*]; costs of horizontal edges are stored in another 2D array *h*[0..*n*][0..*m*-1]. These costs are **positive real-values**.

You can use **readParameters()** to read all parameters (*m*, *n*, *v*[][], and *h*[][]) from **input**. Remember to call release() when done. Check out **hw4.cpp / hw4.py** for more information.

Your output (“result.txt”) should be in the following format: The 1st line is the total cost of the shortest right-down path. The 2nd line is a string of characters of 'u', 'd', 'l' or 'r', standing for up, down, left or right respectively.

Introduction to Computer Science
HW #4
Due: 2018/05/23

Programming Problem II (25%):

Write a **prolog** program to solve the maximum subarray problem: given an array a with length n, find

$$M = \max(L)\,, L = \Sigma_{x=i}^{j} a[x] \;\; 1 \le i, j \le n$$

You need to write the function (save your code in **"msp.pl"**):

- ***maxsum*(*L,M*):** *solve the* maximum subarray problem and find maximum ***M*** for the given list ***L.***

Note that if given an array of all negative values, the maximum value is 0.

Introduction to Computer Science
HW #4
Due: 2018/05/23

Bonus (5%)

In this part you will be given a list of integers. You're the task is to split the list into two lists such that the difference of the sum of the two lists (L1, L2) are minimized.

The input file will be only one line. The first integer would be the length of the list and the remaining are the numbers in the list. Example is shown below:
2 0 1

Your output should be in the following format: The 1st line should be the difference between the two lists. The 2nd and 3rd lines should list the elements in L1 and L2 respectively.

If you accomplished the bonus, save your code in **"bonus.cpp" (or bonus.py)** to hand in.

If you meet the bonus requirements, write "I finished the bonus part." with further details (at least how to do the job) in the readme file to let TA know.