CSC 373 - Algorithm Design, Analysis, and Complexity **Summer 2014**

Assignment 1: Due Friday June 6, 10PM

Please follow the instructions provided on the course website to submit your assignment. You may submit the assignments in pairs. Also, if you use **any** sources (textbooks, online notes, friends) please cite them for your own safety.
You can use those data-structures and algorithms discussed in CSC263 (e.g. merge-sort, heaps, etc.) and in the lectures by stating their name. You do not need to provide any explanation or pseudo-code for their implementation. You can also use their running time without proving them: for example, if you are using the merge-sort in your algorithm you can simply state that merge-sorts running time is $\mathcal{O}(n \log n)$.

Every time you are asked to design an efficient algorithm, you should provide both a short high level explanation of how your algorithms works in plain English, and the pseudo-code of your algorithm similar to what we've seen in class. State the running time of your algorithm with a brief argument supporting your claim. You must prove that your algorithm finds an optimal solution!

1 ~~Lo~~Lallapalooza!

It's summer! It's music fests season! This year at Lallapalooza, they're giving a free trip (everything included) to the 2015 Lallapalooza in ...SPACE! $^{\text{what?}}$ The winner will be drawn at random. To get tickets for the draw, you must attend performances.

Lallapalooza[1] is a music festival for emerging artists. In order to encourage people to attend their performance, artists give free tickets for the draw. Your goal is of course to collect as many tickets as possible.

Each performance lasts h hours, and you get v tickets if you attend. However, if you leave during a performance, you get a fraction of those tickets. For instance, suppose a performance last 5 hours and hands in 10 tickets in total. If you attend 3 hours only of the performance, you collect $3 * 10/5 = 6$ tickets.

There are k stages set up. You're given a schedule of all the performances ahead of time. Each performance P_i has (s_i, h_i, v_i), the start time, the duration and the profit where $s_i, h_i, v_i \in \mathbb{N}$. You can assume that the fraction you get (if any) is always an integer. You can also assume that moving from one stage to another takes an insignificant amount of time[2]. Devise an efficient algorithm that maximizes the number of tickets you will put into the draw. Give a high-level description of how your algorithm works, state the running time of your algorithm and prove that it is optimal.

Input : A list of n performances $\mathcal{P} = \{P_1, P_2, ..., P_n\}$ where each P_i is represented with the tuple (s_i, h_i, v_i).
Output : The maximum possible number of tickets for the draw.

2 Ew Mud!

One of the reasons Lallapalooza is not very popular is because of its location in MudVille. We have a mud problem in MudVille. When running for mayor, you promised the people that if they elect you, you will pave

[1] One day, it'll be big...
[2] Yay, teleportation

the roads so they don't have to walk through mud anymore! AND tourists won't be so reluctant to come to Lallapalooza anymore. The girls are happy they don't have to walk with their heels in the mud! The business owners are happy people will make it to their stores more often! And old people are happy they won't drown in mud anymore! You won the elections! But when you got to office, you realized you have a low budget. As an honest *cough* politician, you decided to keep your promise (sort of) and pave the roads that connect only the main destinations (subway stations, banks, malls, Lallapalooza plaza, etc...). As long as the MudVillers can get from one main destination to another they're happy!

The city council gave you a map of Mudvillle with a list of n locations that need to be reached. In addition, you also have access to a list of all the potential roads that connect these locations, plus the cost of paving each road. Clearly, there could be more than one way to get from one location to another. Paving a road will cost c dollars, where c is always a positive integer.

Input : A list of k roads to be paved $\{r_1, r_2, ..., r_k\}$, a list of k costs $\{c_1, c_2, ..., c_k\}$ where c_i is the price you pay to pave road r_i. A map $\mathcal{M}$ of n locations $\{\mathcal{L}_1, \mathcal{L}_2, ..., \mathcal{L}_n\}$ where each r_i connects two locations $\mathcal{L}_a, \mathcal{L}_b$ in $\mathcal{M}$.
Output : The total minimum cost to pave the roads to connect the n locations.

Devise an algorithm to solve this problem efficiently. Give a high-level description of how your algorithm works, state the time complexity of your algorithm, and prove that your algorithm is optimal.

3 Let The Paving Begin!

You've selected the roads to pave, and chose a company to do the job: CementGiants. CementGiants doesn't pave roads the regular way, they purchase gigantic cement blocks that they cut if necessary and lay them down on the road. You can assume that each road is a straight line of length l_i. The cement blocks have a fixed length $\mathcal{CL}$, and $l_i < \mathcal{CL}$ for all i.

There are $j \leq k$ roads to pave. Once you give your list of road lengths $\mathcal{RL} = \{l_1, l_2, ..., l_j\}$ to CementGiants, they lay the blocks in the order of this list. That is, road r_a with length l_a is paved iff all the roads r_b for $b \leq a$ have already been paved.

Since $l_i < \mathcal{CL}$, every time they cut a block there is always a remaining unused piece of length $\mathcal{CL} - l_i$. CementGiants' goal is **not** to minimize the total number of remaining pieces, nor the number of blocks used in total. Their goal is to minimize the length difference between these remaining pieces. In other words, they want their remaining blocks to be of equal length as much as possible so they can reuse them for some other project. For instance, suppose the list of road lengths is $\mathcal{RL} = \{1, 3, 2, 2\}$ and $\mathcal{CL} = 5$:

- You could either use one block for each road, your remaining pieces would be of length 4, 2, 3, 3 respectively.
- You could use one block for the first road, one for the 2nd and 3rd, and one more block for the last road with remaining pieces of length 4, 0, 3 respectively.
- You could use the first block to cover both the first and last road (total of 3), and use a second block to cover the second and third roads (total of 5). Although this minimizes the number of blocks used (2), it is not a valid operation since we are changing the order in which the roads are paved.
- The best solution would be to use one block for the first two roads and one for the last 2. The remaining blocks are both of length 1 each.

Input : The value $\mathcal{CL}$ and a **fixed** list of j road lengths $\mathcal{RL} = \{l_1, l_2, ..., l_j\}$ where $l_i < \mathcal{CL}$ for all i.
Output : A grouping $(l_1, ..., l_{i_1}), (l_{i_1+1}, ..., l_{i_2}), ..., (l_{i_{s-1}+1}, ..., l_{i_s})$ that minimizes the difference between the

remaining blocks, where $l_{i_s} = l_j$

Note that this difference can be expressed as $\sum\limits_{r=0}^{s-1}(\mathcal{CL} - l_{i_r+1} - l_{i_r+2} - ... - l_{i_{r+1}})^2$.
Clearly $l_{i_r+1} + l_{i_r+2} + ... + l_{i_{r+1}} \leq \mathcal{CL}$ for all $0 \leq r \leq s - 1$.

Construct an efficient algorithm which solves the above problem. Give a high-level description of how your algorithm works, state the time complexity of your algorithm, and prove that your algorithm is optimal.

4 Ole Ola!

The World Cup starts in less than a month!!! You're the head of the soccer[3] division at SportsToday and want to send your journalists to cover all the games, but going to Brazil is expensive, and with the World Cup around the corner, all the prices will skyrocket.

The schedule is out! In reality, games don't overlap, but in our 373 world they do. Suppose some matches overlap. You're given the entire schedule ahead of time which includes the start and end time of each match, and want to send as few journalists as possible to cover all the games.

Define formally the input and output of the problem. Devise an efficient algorithm that minimizes the number of journalists you will send to Brazil. Give a high-level description of how your algorithm works, state the running time of your algorithm and prove that it is optimal.

5 The Faster The Better

Your journalists in Brazil send you daily predictions of which teams (countries) will end up in the semi final. SportsYesterday is a competitor news channel. They have their journalists in place too and get their daily predictions. The rest of the planet can only find out about what is happening in Brazil through the site predictions.com. All news channels submit their predictions there! The news channel to broadcast the predictions first gets more followers and thus makes more money. You got today's predictions: Brazil, Argentina, Spain and Germany.

At SportsYesterday, they use fixed length encoding to transfer all their messages, you can do better! You are given the frequencies of the following symbols in the English alphabet and you want to submit the following message: "BRA.ARG.SPA.GER".

A	0.21
B	0.04
E	0.09
G	0.15
P	0.04
R	0.19
S	0.01
.	0.03

Normalize these frequencies and use your probability distribution to produce a Huffman Tree. What would be the encoding of your prediction message "BRA.ARG.SPA.GER"?

[3] It's called football!