Project 2: Dynamic programming

COT 4400, Summer 2022

Due July 22, 2022

1 Overview

For this project, you will develop an algorithm to efficiently fit together a set of extensible teeth. Designing and implementing this solution will require you to model the problem using dynamic programming, then understand and implement your model.

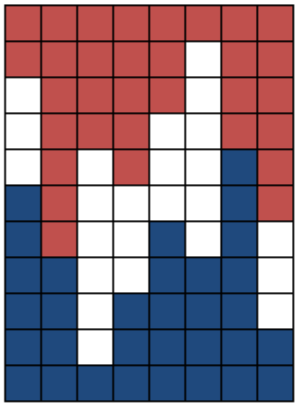
You are only allowed to consult the class slides, the textbook, the TAs, and the professor. In particular, you are not allowed to use the Internet. This is a group project. The only people you can work with on this project are your group members. This policy is strictly enforced.

In addition to the group submission, you will also evaluate your teammates' cooperation and contribution. These evaluations will form a major part of your grade on this project, so be sure that you respond to messages promptly, communicate effectively, and contribute substantially to your group's solution. Details for your team evaluations are in Section 5.2. You will submit the peer evaluations to another assignment on Canvas, labelled "Project 2 (individual)."

A word of warning: this project is team-based, but it is quite extensive and a nontrivial task. You are highly encouraged to start working on (and start asking questions about) this project early; teams who wait to start until the week before the due date may find themselves unable to complete it in time.

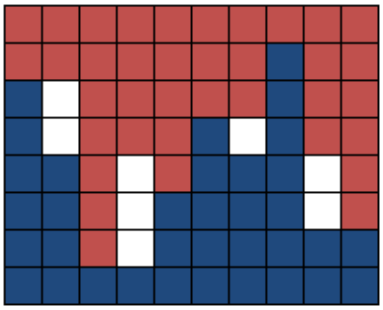
2 Problem Description

In this problem, you are given two arrays, representing the heights of two sets of teeth, and your goal is to fit the teeth together in such a way that you minimize the total height of the interlocking teeth. As an example, if one set has heights 2, 7, 4, 5, 3, 1, 4, 6, and the other has heights 6, 4, 1, 3, 5, 4, 7, 2, you can fit these two sets of teeth one on top of the other with a total height of 11, as shown below:



Height 2, 7, 4, 5, 3, 1, 4, 6 on top
Height 6, 4, 1, 3, 5, 4, 7, 2 on bottom
Aligned height: 11

Of course, if the teeth are required to be aligned exactly, finding the minimum height would be trivial: simply add the heights of the corresponding teeth and take the maximum height. In this project, while you are required to align the first and last teeth on both ends, you are allowed to "extend" the teeth horizontally, effectively creating a new tooth of the same height. (You are allowed to extend the same tooth as many times as you want.) If you are allowed to extend the teeth in the example above, you can get the optimum height of 8 by extending the teeth of height 2 and 3 in the first set and extending the teeth of height 1 and 2 in the second:



Height 2, 2, 7, 4, 5, 3, 3, 1, 4, 6 on top
Height 6, 4, 1, 1, 3, 5, 4, 7, 2, 2 on bottom
Aligned height: 8

Note that the original sets of teeth may not have the same number of teeth. You would need to extend teeth an unequal number of times so that these teeth line up.

3 Project report

In your project report, you should include brief answers to 8 questions. Note that you must use dynamic programming to solve this problem; other solutions will not receive substantial credit.

1. How you can break down a problem instance of aligning a set of n teeth $(a_1, a_2, \ldots, a_n)$ with a set of m teeth $(b_1, b_2, \ldots, b_m)$ into one or more smaller instances? Your answer should include how you calculate the minimum height for the original problem based on the solution(s) to the subproblem(s).

Hint: try to make one small decision and solve the rest of the problem recursively (similar to LCS).

2. What are the base cases of this recurrence?

3. What data structure would you use to recognize repeated problems? You should describe both the abstract data structure, as well as its implementation.

4. Give pseudocode for a memoized dynamic programming algorithm to find the minimum height when aligning extensible teeth $(a_1, a_2, \ldots, a_n)$ and $(b_1, b_2, \ldots, b_m)$.

5. What is the *worst-case* time complexity of your memoized algorithm?

6. Give pseudocode for an iterative algorithm to find the minimum height when aligning extensible teeth $(a_1, a_2, \ldots, a_n)$ and $(b_1, b_2, \ldots, b_m)$. This algorithm does not need to have a reduced space complexity relative to the memoized solution.

7. Can the space complexity of the iterative algorithm be improved relative to the memoized algorithm? Justify your answer.

8. Give pseudocode for an algorithm that identifies which teeth to extend in order to achieve the minimum height. Your algorithm may be iterative or recursive.

   Hint: read section 10.2 in your textbook (especially 10.2.2 and 10.2.3) for a hint on how to do this.

4 Coding your solutions

In addition to the report, you should implement a dynamic programming algorithm that can align two given sets of extensible teeth. Your code may be iterative or recursive, but it must be a dynamic programming algorithm. Also, you may code your solution in C++ or Java, but it must compile and run in a Linux environment. If you are using C++ and compiling your code cannot be accomplished by the command

```
g++ -o teeth *.cpp
```

you should include a Makefile that is capable of compiling the code via the `make` command.

If you choose to implement your code in Java, you should submit an executable jar file with your source. In either case, your source code may be split into any number of files.

Your code will not need to handle invalid input (e.g., teeth with negative height).

4.1 Input format

Your program should read its input from the file `input.txt`, in the following format. The first line of the file has two positive integers n and m specifying the length of the two sets of teeth. The second line will have n positive integers, representing the heights of the teeth in the first set, and the third line will have m positive integers, representing the heights of the teeth in the second set.

4.2 Output

Your program should write its output to the file `output.txt`. Your code should first print out the optimum height on a line. Then, for each of the remaining lines, you should print out an optimal

alignment. Each line should contain exactly two positive integers, representing the heights of the teeth being aligned. The first integer on each line should come from the second line of the input file, while the second integer on each line should come from the third line of the input file. The first and last lines of your alignment should be the first and last teeth in each set; however, the number of lines depends on how many "extensions" you make.

For partial credit, your output file may consist of the optimal height on a line by itself, with no alignment information.

5 Submission

Your submission for this project will be in two parts, the group submission and your individual peer evaluations.

5.1 Group submission

The submission for your group should be a zip archive containing 1) your report (described in Section 3) as a PDF document, and 2) your code (described in Section 4). If your code requires more than a simple command to compile and run then you must also provide a Makefile and/or shell script. You should submit this zip archive to the "Project 2 (group)" assignment on Canvas.

Be aware that your project report and code will be checked for plagiarism.

5.2 Teamwork evaluation

The second part of your project grade will be determined by a peer evaluation. Your peer evaluation should be a text file that includes 1) the names of all of your teammates (including yourself), 2) the team member responsibilities, 3) whether or not your teammates were cooperative, 4) a numeric rating indicating the proportional amount of effort each of you put into the project, and 5) other issues we should be aware of when evaluating your and your teammates' relative contribution. The numeric ratings must be integers that sum to 30.

It's important that you be honest in your evaluation of your peers. In addition to letting your team members whether they do (or do not) need to work on their teamwork and communication skills, we will also evaluate your group submission in light of your team evaluations. For example, a team in which one member refused to contribute would be assessed differently than a team with three functioning members.

You should submit your peer evaluation to the "Project 2 (individual)" assignment on Canvas.

6 Grading

Report	**40 points**
Question 1, 4, and 6	8 points
Questions 2 and 3	2 each
Questions 5, 7, and 8	4 each
Code	**30 points**
Compiles	5
Uses correct input and output format	5
Computes correct answer	15
Good coding style	5
Teamwork	**30 points**

Note that if your algorithm is inefficient, you may lose points for both your pseudocode and your submission. Also, in extreme cases, the teamwork portion of your grade may become negative or greater than 30. In particular, if you do not contribute to your group's solution at all, you can expect to receive an overall grade of 0 on the project.