ADA(B) Homework Assignment 2 (Theory)

Deadline : Feb 22 (Monday) 11.59 pm.

The theory assignment has to be done in a team of at most two members, as already selected by you. The solutions are to be typed either as a word document or latex-ed and uploaded as pdf on GC. We shall strictly not accept solutions written in any other form. Remember that both team members need to upload the HW solution on GC. Collaboration across teams or seeking help from any sources other than the lectures, notes and texts mentioned on the homepage will be considered an act of plagiarism

General instructions:

Each question is worth 15 points. For each question we need you to provide the following:

1. (3 points) A precise description of the subproblems you want to solve in the dynamic program. *Notice this is just the subproblem, not how to solve it, or how it was obtained.*
2. (3 points) A recurrence which relates a subproblem to to "smaller" (Whatever you define "smaller" to mean) subproblems. *Notice this is just the recurrence, not the algorithm or why the recurrence is correct.*
3. (1 points) A procedure to obtain the solution for the original problem using the solutions to the different subproblems. *Often but NOT always, this is the answer of the "largest" subproblem.*
4. (3 points) A clear proof for why the recurrence is correct. Specifically, prove that an optimal solution for your subproblem can be obtained using optimal solutions for "smaller" subproblems via your recurrence. *Notice this is the proof of why the recurrence is correct. Often this involves contradiction arguments involving the optimal solution. This is NOT(!) to prove correctness of the final algorithm.*
5. (3 points) A pseudo-code for a dynamic program which solves the recurrence efficiently
6. (2 points) An argument for the running time of your dynamic program

We don't need an induction proof of correctness of the final dynamic program. Further, we are only interested in the *value* of the optimal solution, not the optimal solution itself. We will give full credit for *any* polynomial time algorithm for every question, so no need to over optimize.

If your solution does not have the structure above, you will be awarded *a zero* (yes, you read that right). There will be *no* concessions on this.

The first question below is from the text by Jeff Erickson. You can find an online copy here.

Problem 1. Solve Question 14 from the above text.

Problem 2. Solve Question 13 from the above text.

Problem 3. Upon time traveling to an ancient laboratory, you find n drops of a *mystical* liquid arranged in an order. The *volume* of each drop is given to you. You find a scroll with the following instructions.

You can do the following operation as many times as you like until just one drop remains:

Combine any two *adjacent* drops and place the combined drop at the same place in the order

When you combine two drops of volumes x and y respectively, you get a new drop whose volume is the sum $x + y$ of the two original volumes (i.e. volume is conserved). Also, you gain an amount of energy for your time traveling device equal to $x^2 + y^2$.

For example,

1. Start with volumes
   $$1 \quad 4 \quad 2 \quad 3 \quad 5$$
2. Combine the second and the third drop to get volumes
   $$1 \quad 6 \quad 3 \quad 5$$
   and gain energy $4^2 + 2^2 = 20$
3. Combine the third and the fourth drop to get volumes
   $$1 \quad 6 \quad 8$$
   and gain energy $3^2 + 5^2 = 34$
4. Combine the first and the second drop to get volumes
   $$7 \quad 8$$
   and gain energy $1^2 + 6^2 = 37$
5. Combine the remaining two drops to get volumes
   $$15$$
   and gain energy $7^2 + 8^2 = 113$

Hence, you gain a total energy of $20 + 34 + 37 + 113 = 204$.

Eager to time travel back to 2021, you'd like to maximize the energy gained for your time traveling device. Give an efficient algorithm to find the sequence of operations to achieve this.

Ungraded (previous problem 2) In the city of Sidhapur, there is a very long straight street on which n people live in houses built next to each other (not necessarily at the same gap) - house i is identified by a coordinate x_i. The citizens currently need to get their bread from the neighboring town of RotiGarh, which they find annoying. Hence the local councillor decides to open k bakeries on the long street. The residents are happy to lease out one floor of their houses for opening a bakery. Your task would be to pick k out of the n houses. Remember that each citizen needs to travel everyday to their closest bakery. Hence, you better come up with a solution such that the total distance travelled by all citizens each day is as small as possible. Design a polynomial (in n and k) time algorithm to pick the k spots.