# Homework 2

CS302 - Fall 2025

*(You should try to solve the problem by yourself first and then compare your solution with the one from AI tools)*

**Question 1:** Fibonacci sequence is defined as

$F_0 = 1$, $F_1 = 1$, $F_n = F_{n-1} + F_{n-2}$ for $n > 1$

(a) Prove that

$$F_n = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^{n+1} - \left(\frac{1-\sqrt{5}}{2}\right)^{n+1}\right]$$

(b) Develop an algorithm to calculate $F_n$ faster than O(n)

**Question 2:** Each semester FUV offers many courses, each course has a capacity and is linked to an instructor and a major. Main FUV campus has some classrooms, each class room has a capacity. There is a list of time slots (e.g. 8:00am - 9:30 am Mon/Wed, 9:45am - 11:15am Wed/Friday, 6:00pm - 7:30pm Thu/Tuesday) when a course can be scheduled. We need to find a way to schedule all courses, each one is at a room and a time slot such that (i) at a time point each room is for one course only, (ii) course capacity is less than room capacity, (iii) no time conflict for an instructor who teaches two or more courses, (iv) no time conflict for courses of the same major, and (v) the number of courses scheduled at 8:00am or 6:00pm is minimized.

(a) What is the solution space of this problem?

(b) What is the time complexity if we use the exhaustive search on this space?

(c) Describe a greedy algorithm to solve this problem.

**Question 3: Knapsack problem**

(a) Given the item list below and the weight limit is 18, find the best item set using three greedy strategies (i) the most valuable item first, (ii) the lightest item first, (iii) the best average value (i.e. best value/weight) item first.

| **Item** | **value** | **weight** |
|---|---|---|
| 1 | 7 | 2 |
| 2 | 11 | 3 |
| 3 | 18 | 5 |
| 4 | 22 | 6 |

| | | |
|---|---|---|
| 5 | 25 | 7 |
| 6 | 28 | 9 |

(b) Fill the value table (see below) of the Knapsack problem

| Item set | weight limit | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| {} | | | | | | | | | | | | | | | | | | | |
| {1} | | | | | | | | | | | | | | | | | | | |
| {1,2} | | | | | | | | | | | | | | | | | | | |
| {1,2,3} | | | | | | | | | | | | | | | | | | | |
| {1,2,3,4} | | | | | | | | | | | | | | | | | | | |
| {1,2,3,4,5} | | | | | | | | | | | | | | | | | | | |
| {1,2,3,4,5,6} | | | | | | | | | | | | | | | | | | | |

(c) In the lecture, the algorithm solving Knapsack problem is only to find the optimal value we can get, but it does not show us the optimal item set. How can we modify the algorithm to find the optimal item set?

**Question 4 (Maximum sub-array sum):** Given an array of numbers, find a continuos sub-array of this array that the sum is maximum

(a) Present a simple $O(n^3)$ algorithm to solve this problem

(b) Develop a dynamic programming algorithm to solve this problem in O(n). More specifically, you need to find

(i) The maximum sum

(ii) The start index and the end index of the sub-array

**Question 5 (Longest Increasing Subsequence, optional):**

Subsequence of an array is a continuous or non-continuos sub-array of this array. The Longest Increasing Subsequence (LIS) problem is to find the length of the longest subsequence of a given array such that all elements of the subsequence are sorted in increasing order.
Example 1: The length of LIS of the array [50, 3, 10, 7, 40, 80] is 4 and LIS is [3, 7, 40, 80]
Example 2: The length of LIS of the array [10, 22, 9, 33, 21, 50, 41, 60, 80] is 6 and LIS is [10, 22, 33, 50, 60, 80]

You should try to solve the problem and then check the solution at https://www.geeksforgeeks.org/longest-increasing-subsequence-dp-3/

**Question 6 (Pots of Gold game, optional):**
There are two players, A and B, in Pots of Gold game, and pots of gold arranged in a line, each containing some gold coins. The players can see how many coins are there in each gold pot, and each player gets alternating turns in which the player can pick a pot from either end of the line. The winner is the player who has a higher number of coins at the end. The objective is to "maximize" the number of coins collected by A, assuming B also plays "optimally", and A starts the game.

| Pots of gold | Player | Opponent |
|---|---|---|
| 4, 6, 2, 3 | 3 | |
| 4, 6, 2 | | 4 |
| 6, 2 | 6 | |
| 2 | | 2 |
| Total | 9 coins | 6 coins |

You should try to solve the problem and the check the solution at https://www.techiedelight.com/pots-gold-game-dynamic-programming/