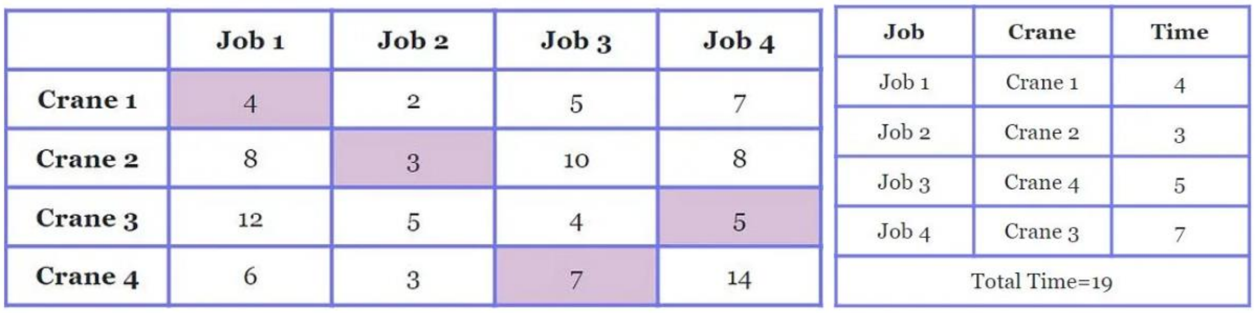
Assignment 2 of algorithm design

Please upload the assignment file in pdf format in elearn.

1. **Write a pseudo-code** that combines(جمع بزند) two (n*n) matrices such as A and B using the conquer-and-divide method. in such a way that in each step it divides each of the matrices into 4 parts (n/2)*(n/2) and recursively performs addition for the sub-matrices in a similar way.
   Calculate the time complexity of the proposed algorithm.

2. In a construction site there are 4 cranes. Each crane must be allocated to one job. The time required for each job for each crane is shown in the table below. Find the best assignment of cranes to the jobs so that the time required to finish the jobs is minimum. (explain the answer using **Greedy Approach**)



	Job 1	Job 2	Job 3	Job 4
Crane 1	4	2	5	7
Crane 2	8	3	10	8
Crane 3	12	5	4	5
Crane 4	6	3	7	14

Job	Crane	Time
Job 1	Crane 1	4
Job 2	Crane 2	3
Job 3	Crane 4	5
Job 4	Crane 3	7
Total Time=19		

The highlighted boxes show the most optimal assignment.

3. **Cookie assignment** Consider the following problem:
   You are baby-sitting n children and have m > n cookies to divide between them. You must give each child exactly one cookie (of course, you cannot give the same cookie to two different children). Each child has a greed factor gi, 1 ≤ i ≤ n which is the minimum size of a cookie that the child will be content with; and each cookie has a size sj, 1 ≤ j ≤ m. Your goal is to maximize the number of content children, i.e.., children i assigned a cookie j with gi ≤ sj.

```
CookieGive ($Children[1..n], Cookies[1..m]$).
 1. Sort $Children$ by greed , $Cookies$ by size, largest to smallest.
 2. $I \leftarrow 1, J \leftarrow m$.
 3. FOR $K = 1$ to $N$ do:
 4.     IF $s_I \geq g_K$ THEN $Assign[K] = I, I++$
 5.                 ELSE $Assign[K] = J, J--$.
 6. Return $Assign$.
```

a) Give a correct greedy algorithm for this problem.
b) Calculate the time complexity of the algorithm.

4. Longest Common Subsequence of Two Sequences

Problem Description:

Task: Given two sequences $A = (a1, a2, \dots, an)$ and $B = (b1, b2, \dots, bm)$, find the length of their longest common subsequence, i.e., the largest non-negative integer p such that there exist indices $1 \leq i1 < i2 < \cdots < ip \leq n$ and $1 \leq j1 < j2 < \cdots < jp \leq m$, such that $ai1 = bj1, \dots, aip = bjp$.

Input Format :First line: n. Second line: $a1, a2, \dots, an$. Third line: m. Fourth line: $b1, b2, \dots, bm$.

Constraints: $1 \leq n, m \leq 100$; $-109 < ai, bi < 109$.

Output Format: Output p.

Sample 1.

Input:

```
3
2 7 5
2
2 5
```

Output:

```
2
```

A common subsequence of length 2 is $(2, 5)$.

Sample 2.

Input:

```
1
7
4
1 2 3 4
```

Output:

```
0
```

The two sequences do not share elements.

Sample 3.

Input:

```
4
2 7 8 3
4
5 2 8 7
```

Output:

```
2
```

One common subsequence is $(2, 7)$. Another one is $(2, 8)$.

Write a **code** using any programming language you want(Python, C++,..). You should also put the **output image** of your code in the assignment file.

5. You are given an integer array coins representing coins of different denominations and an integer amount representing a total amount of money. Write a function to compute the minimum number of coins needed to make up that amount. If it's impossible to make up the amount, return -1. You may assume that you have an infinite number of each kind of coin.
Implement a dynamic programming solution to solve this problem efficiently.

Example:

Input: coins = [1, 2, 5], amount = 11

Output: 3

Explanation: You can use one 5-cent coin, one 2-cent coin, and one 1-cent coin to make up 11 cents, so the minimum number of coins needed is 3.

Input: coins = [2], amount = 3

Output: -1

Explanation: It's impossible to make up 3 cents using only 2-cent coins, so return -1.

6. Given an unsorted array of integers, find the length of the longest increasing subsequence. Implement a dynamic programming solution to find the length of the longest increasing subsequence.

Example:

Input: nums = [10, 9, 2, 5, 3, 7, 101, 18]

Output: 4

Explanation: The longest increasing subsequence is [2, 3, 7, 101], therefore the length is 4.