This is an individual/group homework assignment. Groups of **up to 2** students can submit joint solutions. **EXACTLY** one student in the group should submit the solution document, and at the beginning of the document, you should clearly **state the names of all groupmates**. The student who submits the solution on behalf of the group is responsible for sharing the grading feedback with the rest of the group. If a groupmate's name is missing on the submitted file, that student will NOT receive any points for the corresponding assignment.

No credit will be given for handwritten solutions (except for figures or long equations that might be handwritten). You must submit your assignment on BB in the related digital repository (located under the Assignments/Digital Repository for the assignments) by the specified deadline.

As it appears in the course syllabus, for the homework assignments, students are encouraged to discuss the problems with others, but you are expected to turn in the results of your own effort (not the results of a friend's efforts in another group or so). Even when not explicitly asked, you are supposed to justify your answers concisely and clearly.

Question 1.

Let $f(n)= n^4 \log n + 2n^6 + 10^{10}$. Bound the growth of function f(n) asymptotically. Can you come up with a tight bound? You should clearly justify your answer here (use formal proof).

Question 2. Consider an instance of the stable matching problem with 4 men and 4 women. Provide their preference lists as you would like (show them using two tables as in the slides).

a) Then provide an assignment that is *unstable*.
b) Now provide an assignment that is *stable*.

For each case above, explain in detail why the assignment is stable/unstable.

Question 3.

The interest, naturally, was in finding a way of assigning each student to at most one hospital, in such a way that all available positions in all hospitals were filled. (Since we are assuming a surplus of students, there would be some students who do not get assigned to any hospital.)

We say that an assignment of students to hospitals is *stable* if neither of the following situations arises.

- First type of instability: There are students s and s', and a hospital h, so that
  - s is assigned to h, and
  - s' is assigned to no hospital, and
  - h prefers s' to s.
- Second type of instability: There are students s and s', and hospitals h and h', so that
  - s is assigned to h, and
  - s' is assigned to h', and
  - h prefers s' to s, and
  - s' prefers h to h'.

So we basically have the Stable Matching Problem, except that (i) hospitals generally want more than one resident, and (ii) there is a surplus of medical students.

Show that there is always a stable assignment of students to hospitals, and give an algorithm to find one.

Select ONLY one out of the two questions below to solve.
Question 4. Let f(n)= 3 log n^5+ n^2 +14, then f(n) is O(?). Justify.

Question 4. True or false? Assume you have functions F(n), G(n) and T(n) such that F(n)= O(T(n)) and G(n)= O(T(n)). Then is this true that F(n) +G(n) = O(T(n))? Give a proof or a counterexample.

Extra Practice Questions. No need to submit your answers to them:
Question 5)

a) Consider the bubble sort algorithm. Write the algorithm and briefly explain how it works.
b) Consider the following list (array) of integer numbers:

12, 3, 9, 32, 4, 56, 0, 1, 2
Show how the algorithm works on the given list of numbers to sort it in ascending order. Show all your work step by step. Explain enough.

c) Write **a complete program** in C++/Java to implement your bubble sort algorithm in part (a). Paste your complete code here as the solution for part (c). Also, share the screenshots of the output you get from your program once you run your code for the list given at part (b).

d) Analyze your complete program for the worst-case time complexity STEP BY STEP as what we did for the **linear search**. **Show all your work**. Then specify the worst-case time complexity of **bubble** sort in Big-O notation.

Question 6)

a) Consider the **selection** sort algorithm. Write the algorithm and briefly explain how it works.
b) Consider the following list (array) of integer numbers:
12, 3, 9, 32, 4, 56, 0, 1, 2
Show how the algorithm works on the given list of numbers to sort it in ascending order. Show all your work step by step. Explain enough.
c) Write **a complete program** in C++/Java to implement your **selection** sort algorithm in part (a). Paste your complete code here as the solution for part (c). Also, share the screenshots of the output you get from your program once you run your code for the list given at part (b).

d) Analyze your complete program for the worst-case time complexity STEP BY STEP as what we did for the **linear search**. **Show all your work**. Then specify the worst-case time complexity of the **selection** sort in Big-O notation.