1. Consider the following flow network.
   • (10 marks) In the following network a minimum cut has capacity 10. Either prove that this statement is true, or show that it is false.
   • (5 marks) Find a minimum cut \((S, T)\) separating \(s\) from \(t\). You need to indicate which vertices are in \(S\) and which are in \(T\).

   ![Flow Network Diagram]

2. (10 marks) Let \(G = (V, E)\) be a flow network with source \(s\), sink \(t\) and a positive integer capacity \(c(u, v)\) on every edge \((u, v) \in E\). Let \((S, T)\) be a minimum cut of \(G\) separating \(s\) from \(t\). Let \(k > 0\) be a positive integer.

   **Claim.** If the capacity of every edge \((u, v) \in E\) is increased by \(k\), so the new capacity of edge \((u, v)\) is \(c'(u, v) = c(u, v) + k\), the cut \((S, T)\) will still be a minimum cut with respect to the new capacities.

   Determine whether the above statement is true or false. If it is true, give proof for it. If it is false, give an example proving the statement wrong.

3. Let \(G = (V, E)\) be a flow network with source \(s\), sink \(t\), and integer capacities. As always, let \(m\) be the number of edges and \(n\) be the number of vertices of \(G\). Assume that we are given a maximum flow \(f\) of \(G\), i.e. we know the value of the flow \(f(u, v)\) on every edge \((u, v)\) of \(G\).

   a. (10 marks) Let the capacity of a single edge \((u, v) \in E\) be increased by an integer constant value \(k > 0\). Write in pseudocode, similar to the one used in class, an \(O(m + n)\)-time algorithm to compute a maximum flow for the modified flow network.
   
   b. (10 marks) Prove that your algorithm computes a maximum flow.
   
   c. (10 marks) Show that the time complexity of your algorithm is \(O(m + n)\).

4. A set \(P = \{P_1, P_2, \ldots, P_k\}\) of \(k\) computer programs are to be executed by a group \(C = \{C_1, C_2, \ldots, C_l\}\) of computers. The computer programs are classified into \(b\) types \(T_1, T_2, \ldots, T_b\), depending on how many computer resources they require. Each computer program \(P_i\) has a unique type, \(T_{\pi(i)}\).

   Each computer \(C_i\) can execute a set of at most \(a(i)\) programs, but it can process at most \(a(i, j)\) programs of type \(T_j\), for each \(j = 1, 2, \ldots, b\). The problem is to assign all the programs
in \( P \) to the computers so the the above conditions are satisfied, or to show that such an assignment is not possible.

For example, assume that there are 7 programs \( \{P_1, P_2, P_3, P_4, P_5, P_6, P_7\} \) of 3 types \( T_1, T_2, \) and \( T_3 \). Programs \( P_1 \) and \( P_2 \) are of type \( T_1 \), programs \( P_3, P_4, \) and \( P_5 \) are of type \( T_2 \) and the other ones are of type \( T_3 \). There are 3 computers, \( C_1, C_2, \) and \( C_3 \):

- \( C_1 \) can process at most 2 programs, of which 1 could be of type \( T_1 \) and at most 2 could be of type \( T_2 \), so \( a(1) = 2, a(1, 1) = 1, a(1, 2) = 2, \) and \( a(1, 3) = 0. \)
- For \( C_2 \) we have \( a(2) = 3, a(2, 1) = 1, a(2, 2) = 1, a(2, 3) = 1. \)
- For \( C_3 \) we have \( a(3) = 2, a(3, 1) = 2, a(3, 2) = 0, a(3, 3) = 1. \)

A solution is to assign \( P_3 \) and \( P_4 \) to \( C_1 \), \( P_1, P_5 \), and \( P_6 \) to \( C_2 \) and \( P_2 \) and \( P_7 \) to \( C_3 \).

\( i. \) (20 marks) Write in pseudocode an algorithm for solving this problem. If there is a way of assigning programs to computers, the algorithm must return the value true, otherwise it must return the value \( false. \)

\( ii. \) (15 marks) Prove that your algorithm correctly solves the above problem.

\( iii. \) (10 marks) Compute the time complexity of your algorithm.