You must submit your assignment through the Crowdmark system. You will receive by email an invitation through which you can submit your work in the form of separate PDF documents with your answers to each question of the assignment. To work with a partner, you and your partner must form a group on Crowdmark. Crowdmark does not enforce a limit on the size of groups. The course policy that limits the size of each group to at most two remains in effect: submissions by groups of more than two persons will not be graded.

- It is your responsibility to ensure that the PDF files you submit are legible. To this end, I encourage you to learn and use the LaTeX typesetting system, which is designed to produce high-quality documents that contain mathematical notation. You are not required to produce the PDF files you submit using LaTeX; you may produce it any way you wish, as long as the resulting document is legible.

- By virtue of submitting this assignment you (and your partner, if you have one) acknowledge that you are aware of the policy on homework collaboration for this course. If you collaborate with another student on an assignment, you and your partner must submit only one copy of your solution, with both of your names. The solution will be graded in the usual way and both partners will receive the same mark. Collaboration involving more than two students is not allowed.

- For any question, you may use data structures and algorithms previously described in class, or in prerequisites of this course, without describing them. You may also use any result that we covered in class, or is in the assigned sections of the official course textbooks, by referring to it.

- Unless we explicitly state otherwise, you should justify your answers. Your paper will be graded based on the correctness and efficiency of your answers, and the clarity, precision, and conciseness of your presentation.

---

**Question 1.** (15 marks) A temporary employment agency has a set of workers \( W = \{w_1, w_2, \ldots, w_n\} \). Each worker \( w_i \) is qualified for a subset \( Q_i \) of a set of skills \( S = \{s_1, s_2, \ldots, s_k\} \) (e.g., answering phones, filing, word-processing, Excel, etc). In addition, each worker \( w_i \) is available for a certain number \( h_i \) of hours per day, where \( h_i \) is an integer in the range 0 to 8.

Every day, the agency receives a set of \( m \) tasks \( T = \{t_1, t_2, \ldots, t_m\} \) to be carried out. Each task \( t_j \) requires exactly one skill \( \ell_j \in S \), and a non-negative integer number of hours \( r_j \).

An assignment (of workers to tasks) is a set \( A \) of triples of the form \((w, t, d)\) where \( w \) is a worker, \( t \) is a task, and \( d > 0 \) is a number of hours (not necessarily an integer), subject to the constraints listed below. Such a triple indicates that worker \( w \) is assigned to task \( t \) for \( d \) hours. Note that the assignment \( A \) may contain several triples with the same first component (when a worker is assigned to multiple tasks), or with the same second component (when several workers are assigned to the same task). The constraints that an assignment must satisfy are:

(a) If a worker \( w_i \) is assigned to task \( t_j \), then \( w_i \) is qualified for the skill \( \ell_j \) that \( t_j \) requires.

(b) The total number of hours that worker \( w_i \) is assigned to tasks does not exceed his or her availability \( h_i \).

(c) The total number of hours that workers are assigned to task \( t_j \) does not exceed the task’s time requirement \( r_j \).
Note that an assignment is not required to fill all of a worker’s available hours or to provide all the hours required to complete a task.

a. Describe an algorithm that, given the set of workers $W$, the set of tasks $T$, the set of skills $S$, the qualifications $Q_i$ and availability $h_i$ of every worker $w_i \in W$, and the skill $\ell_j$ and time requirement $r_j$ of every task $t_j \in T$, finds an assignment $A$ that maximizes the total number of hours assigned to the workers. Your algorithm should run in $O(n^2m)$ time. Justify the correctness of your algorithm, and analyse its running time.

**Hint:** Reduce the given problem to maximum flow.

b. Suppose that for every hour that a worker is assigned to task $t_j$, the agency makes a profit $p_j$. The agency wants to find an assignment that maximizes its total profit.

Express this optimization problem as a linear program. Explain the meaning of your variables and of each of your constraints. You are not asked to justify anything here; only to provide a linear program and the required explanation.

**Question 2.** (10 marks) We are given the sequence of “weights” of $n$ items, $w_1, w_2, \ldots, w_n$, where $0 < w_i \leq 1$ is the weight of item $i$, $1 \leq i \leq n$. We want to place these $n$ items into the smallest possible number of identical containers, each of capacity 1. That is, the sum of the weights of the items placed on any container must not exceed 1. For example, suppose the weights are 0.5, 0.25, 0.75, 0.5, 0.75. Then the minimum number of containers in which to place these items is three.

Finding the minimum number of containers is a well-known NP-hard problem.

Consider the following greedy heuristic for this problem: Suppose the containers are numbered 1, 2, \ldots; these are initially empty. Consider the items in the order given. When considering item $i$, place it in the first container that (together with the items previously placed in it) can accommodate the item without exceeding its capacity; that is, place $i$ in container $j$, where $j$ is the smallest positive integer such that the weight of the items presently in container $j$ plus $w_i$ is at most 1. Applied to the example above, this heuristic places items 1 and 2 in the first container, and each of the remaining three items in a separate container — i.e., it uses four containers instead of the optimal three.

Prove that this heuristic is a 2-approximation algorithm. That is, for any input $w_1, w_2, \ldots, w_n$, where $0 < w_i \leq 1$, if the minimum possible number of containers for items with these weights is $c^*$ and the number of containers used by the above greedy algorithm is $c$, then $c \leq 2c^*$.

**Hint:** (1) Note that $c^* \geq \lceil \sum_{i=1}^{n} w_i \rceil$. (You should explain why this is true.) (2) In the assignment of items to containers made by the greedy algorithm, how many containers can be at most half-full? (Prove your answer.)

**THAT’S IT WITH HOMEWORK, FOLKS!**