

General instructions. Answer the problems given in the rest of this document. Edit this LaTeX template to add your answers, as described in the file named `00_README.txt`, then compile it to produce `homework_9.pdf`.

Don't know? Say so. If you don't know the answer to a problem or any part of the problem, we encourage you to simply write “*I don't know*” instead of giving an answer. (If you write “I don't know” you can also add any questions you might have.)

Gentle reminder: homework is for learning. As noted in [Lecture Note 1](#), engaging fully with the homeworks is the best way to learn the material. *The primary role of homeworks is to help you learn the material, and the primary role of feedback on homeworks is to help you improve your ideas.*

Please try to explain your ideas as clearly as you can. The clearer your explanations are, the more useful the feedback on them can be.

Collaboration and citation policy. If you are doing this homework as part of a course, follow the course's collaboration and citation policy. In any case, at the end of each answer, list the people you collaborated with, and cite every source that your answer uses ideas from (other than the lecture notes and the sources listed in the lecture notes). If there are none, write “none”.

Problem 1. (Sections 7.5 and 7.6; Lecture Note 2)

Consider the following problem, Counting Subset Sums:

input: A pair $I = (v, T)$ where $v = (v_1, v_2, \dots, v_n)$ is a sequence of positive integers and $T \geq 0$ is an integer.

output: The number of subsequences of v with total T . That is,

$$\left| \{S \subseteq \{1, 2, \dots, n\} : v(S) = T\} \right|,$$

where for notational convenience throughout we define $v(S) = \sum_{i \in S} v_i$.

This problem differs from Counting Ways to Make Change (HW 7 Problem 3). In this problem each of the given numbers can be used at most once.

Fix an input (v, T) . For each $i \in \{0, 1, \dots, n\}$ and $t \in \{0, 1, \dots, T\}$, define $N(i, t)$ to be the number of subsequences of $(v_1, \dots, v_i)$ with total t . That is,

$$N(i, t) = \left| \{S \subseteq \{1, 2, \dots, i\} : v(S) = t\} \right|.$$

The correct output for the given input is $N(n, T)$.

- (a) Give a recurrence relation for $N(i, t)$ for all $i \in \{0, 1, \dots, n\}$ and $t \in \{0, 1, \dots, T\}$, including the base cases. State the recurrence relation as a lemma (see the template).
- (b) Complete the partial long-form proof of the lemma given in the template. You can use a proof structure similar to the one for the long-form proof of correctness for Knapsack in Section 7.6.

You don't need to give the resulting algorithm or analyze the running time. But you can test your recurrence, by implementing the resulting algorithm, at <https://www.hackerrank.com/nealeyong-algs101-subset-sums-and-piles> (the “Counting Subset Sums” challenge, which asks for the count mod 2^{30}). (This is just for testing. Hacker-Rank submissions won't be graded.)

Problem 1 collaboration and citations

I collaborated on this problem with the following people:

● ★ ⋯ ★

My answers use ideas from the following sources (other than the lecture notes and textbooks):

● ★ ⋯ ★

Problem 1 answer**(a)**

Lemma 1. For any $i \in \{0, 1, \dots, n\}$ and $t \in \{0, 1, \dots, T\}$, for N as defined in the problem statement,

$$N(i, t) = \begin{cases} \star \cdots \star & \text{if } \star \cdots \star, \\ \star \cdots \star & \text{if } \star \cdots \star, \\ \star \cdots \star & \text{if } \star \cdots \star, \\ \star \cdots \star & \text{if } \star \cdots \star. \end{cases}$$

(b)

Proof (long form).

1. Consider any $i \in \{0, \dots, n\}$ and $t \in \{0, 1, \dots, T\}$.
2. Recall that $N(i, t) = |\{S \subseteq \{1, 2, \dots, i\} : \sum_{j \in S} v_j = t\}|$.
- 3.1. Case 1. Consider the first base case $\star \cdots \star$.
- 3.2. $\star \cdots \star$
- 3.3. So $N(i, t) = \star \cdots \star$.
- 4.1. Case 2. Consider the second base case $\star \cdots \star$.
- 4.2. $\star \cdots \star$
- 4.3. So $N(i, t) = \star \cdots \star$.
- 5.1. Case 3. Next consider the case when $\star \cdots \star$.
- 5.2. Consider the subsets $S \subseteq \{1, 2, \dots, i\}$ such that $v(S) = t$.
- 5.3. Partition these into two types: (A) those that don't include i , and (B) those that do.
- 5.4. $N(i, t)$ is the number of subsets of either type.
- 5.5. The type-A subsets are just the subsets of $\{1, \dots, i-1\}$ with $v(S) = t$.
- 5.6. $\star \cdots \star$
- 5.7. The type-B subsets $\star \cdots \star$
- 5.8. $\star \cdots \star$
- 5.9. By Steps 5.4, 5.6 and 5.8, $\star \cdots \star$
- 5.10. So the recurrence holds in this case.
- 6.1. Case 4. Finally consider the remaining case, when $\star \cdots \star$.
- 6.2. $\star \cdots \star$
- 6.3. $\star \cdots \star$
- 6.4. $\star \cdots \star$
- 6.5. So the recurrence holds in this case.
7. By the case analysis (Cases 1–4) above, the recurrence holds. □

Problem 2. (Sections 7.5 and 7.6)

Design a dynamic-programming algorithm for Smallest Subset Sum:

input: A pair $I = (v, T)$ where $v = (v_1, v_2, \dots, v_n)$ is a sequence of positive integers and $T \geq 0$ is an integer.

output: The minimum size of any subsequence of v with total T :

$$\min \{|S| : S \subseteq \{1, 2, \dots, n\}, v(S) = T\},$$

where, for notational convenience throughout we define $v(S) = \sum_{i \in S} v_i$.

Recall that the minimum of the empty set is ∞ . (So, if no subsequence of v has total T , the output should be ∞ .)

- Per Section 7.4, define the subproblems that your algorithm solves for a given input $(v = (v_1, \dots, v_n), T)$.
- Which subproblem gives the final answer?
- State an appropriate recurrence relation, including boundary cases.
- In terms of n and T , what is the big- Θ running time of the resulting iterative (or recursive and memoized) algorithm?

You can test your algorithm at <https://www.hackerrank.com/neaileyong-algs101-subset-sum-and-lcs>. (This is just for testing. Hacker-Rank submissions won't be graded.)

Problem 2 collaboration and citations

I collaborated on this problem with the following people:

● ★ ⋯ ★

My answers use ideas from the following sources (other than the lecture notes and textbooks):

● ★ ⋯ ★

Problem 2 answer

- (a) Here are the subproblems that the algorithm solves.

For each $\star \cdots \star$, define $\star \cdots \star$ to be $\star \cdots \star$. That is,

$$\star \cdots \star = \star \cdots \star.$$

- (b) The correct output for the given input is

$$\star \cdots \star.$$

- (c) Here is the recurrence relation:

$$\star \cdots \star = \begin{cases} \star \cdots \star & \text{if } \star \cdots \star, \\ \star \cdots \star & \text{if } \star \cdots \star, \\ \star \cdots \star & \text{if } \star \cdots \star, \\ \star \cdots \star & \text{if } \star \cdots \star. \end{cases}$$

- (d) The running time is

$$\star \cdots \star.$$

Problem 3. (Section 7.8)

Design an $O(\ell mn)$ -time dynamic-programming algorithm for the Three-Way LCS problem:

input: a triple $I = (A[1..\ell], B[1..m], C[1..n])$ of three sequences.

output: the maximum length of any common subsequence of $A[1..\ell]$, of $B[1..m]$, and of $C[1..n]$.
(That is, the maximum length of any sequence that is simultaneously a subsequence of A , of B , and of C .)

- Per Section 7.4, define the subproblems the algorithm solves given input $(A[1..\ell], B[1..m], C[1..n])$.
- Which subproblem gives the final answer?
- State the recurrence relation, including boundary cases.
- What is the running time of the resulting (iterative, or recursive and memoized) algorithm?

You can test your algorithm at <https://www.hackerrank.com/neaileyong-algs101-subset-sum-and-lcs>. (This is just for testing. Hacker-Rank submissions won't be graded.)

Problem 3 collaboration and citations

I collaborated on this problem with the following people:

● ★ ⋯ ★

My answers use ideas from the following sources (other than the lecture notes and textbooks):

● ★ ⋯ ★

Problem 3 answer

- (a) Here are the subproblems that the algorithm solves.

For each $\star \cdots \star$ define $\star \cdots \star$ to be $\star \cdots \star$.

- (b) The correct output for the given input is

$\star \cdots \star$

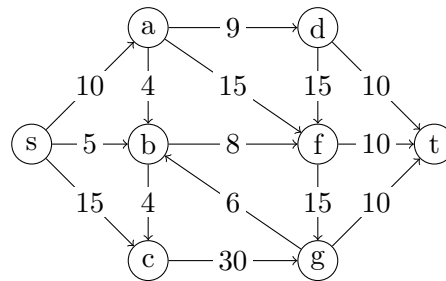
- (c) Here is the recurrence relation:

$$\star \cdots \star = \begin{cases} \star \cdots \star & \text{if } \star \cdots \star, \\ \star \cdots \star & \text{if } \star \cdots \star, \\ \star \cdots \star & \text{if } \star \cdots \star. \end{cases}$$

- (d) The running time is

$\star \cdots \star.$

Problem 4. Consider the flow network $G = (V, E)$ with edge capacities as shown below:



- Find a maximum-value s - t flow f in the network. Draw the network, labeling each edge (u, w) with the flow $f(u, w)$ on the edge.
- What is the value of the flow?
- Find an s - t cut $(S, T = V \setminus S)$ of minimum capacity in the network. For the cut you found, what is S ?
- And what is the capacity of the cut?

Problem 4 collaboration and citations

I collaborated on this problem with the following people:

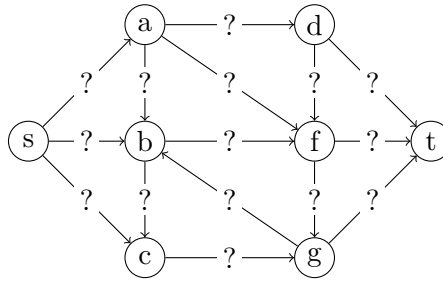
● ★ ⋯ ★

My answers use ideas from the following sources (other than the lecture notes and textbooks):

● ★ ⋯ ★

Problem 4 answer

(a) Here is the flow:



(b) The value of the flow is

★...★.

(c) The s - t cut is $(S, T = V \setminus S)$ where

$$S = \{s, \star \dots \star\}.$$

(d) The capacity of the cut is

★...★.