

COS 330: Great Ideas in Theoretical Computer Science

Fall 2026

Precept 6

Mid Practice

PROBLEM-SOLVING SESSION

Today is a **problem-solving session** dedicated to midterm practice. You will likely run out of time before finishing the problem, but make sure you can solve part (a) on your own, and then try to make as much progress as you can on part (b).

Problem 1: Platform Jumping

You are playing the following video game:

- There are platforms numbered $0, 1, \dots, n$, for some $n \geq 1$. Each platform i contains a reward of $c_i \geq 0$ coins. You start on platform 0, and $c_0 = 0$.
- You have $T \geq n$ total turns to collect as many coins as you can.
- Each turn, you can either *stay put* or *hop right*.
 - If you *stay put* on platform i , you get c_i coins this turn.
 - If you *hop right* from platform i , you move to platform $i + 1$, and get c_{i+1} coins.¹

For example, one strategy is to stay put on platform 0 for one turn, hop right to platform 1, stay put there for a few turns, and then hop right again. A strategy can mix stays and hops in any order, as long as it uses exactly T turns.

(a) Let $M = \max_{1 \leq i \leq n} c_i$ be the largest reward, and consider the *greedy* strategy: hop right to a platform with reward M (the leftmost one, if there are several), and then stay put there for all the remaining turns. Let OPT be the maximum possible number of coins you can earn with any strategy, and call a strategy *optimal* if it earns OPT coins.

- (i) Show that the greedy strategy is not always optimal.
- (ii) Prove that the greedy strategy always earns at least $\text{OPT} - n \cdot M$ coins.

¹You may not hop right from platform n .

(b) Prove that there is an optimal strategy of the following form: hop right k times, for some $k \in \{0, 1, \dots, n\}$, and then stay put on platform k for all of the remaining $T - k$ turns.

(c) Design an $O(n)$ -time algorithm to compute the maximum possible number of coins you can earn with any strategy.² Prove that your algorithm is correct, and analyze its running time.

²To be extra clear: any arithmetic operation costs only one unit of time – you do not need to worry about the size of numbers when counting arithmetic operations.