NYU SHANGHAI 上海纽约大学

CSCI-SHU 210 Data Structures 100 Points

Homework Assignment 2 - Analysis of Algorithms

Problem 1 – Maximum Subarray Sum - 25 Points

You are given an array of integers, nums. Your task is to find the contiguous subarray (containing at least one number) with the largest sum and return its sum.

Requirements

- Your solution must have a time complexity of O(n), where n is the length of the input nums.
- Your solution must have a space complexity of O(1), regardless of the size of the input.

Important

- You cannot use numpy or any other third-party library.
- Your subarray has to be in sequence. [1,4,2,1,4] is not a solution in the below example.

Example 1

- nums = [-2, 1, -3, 4, -1, 2, 1, -5, 4]
- result = max_subarray_sum(nums)
- print(result) # should print 6, based on the subarray [4, -1, 2, 1]

Problem 2 – Enchanted Forest - 25 Points

In the Enchanted Forest, a magical grove contains ancient trees, each labeled with a unique positive integer. The trees must be arranged in ascending order of their value to unlock the hidden portal leading to the treasure. You must implement a program that employs the Insertion Sort algorithm to arrange the trees in the correct order.

Requirements

- Complete the given function $sort\_enchanted\_trees(l)$.
- Your algorithm has to be O(n^2) time complex.
- You are only allowed to use O(1) additional space.
- You can not use the built in sort function.
- You have to use insertion sort.

Example 1

- tree_values = [30, 20, 40, 10, 50, 15, 35, 25, 45]
- sorted_trees = sort_enchanted_trees(tree_values)
- # output should equal [10, 15, 20, 25, 30, 35, 40, 45, 50]

NYU SHANGHAI 上海纽约大学

Problem 3 – Root Finder - 25 Points

You are tasked with finding a solution to an arbitrary f of the equation $f(x) = 0$. If a given function has two points, low and $high$ so that $f(low)$ and $f(high)$ have opposite signs, then there must be a root between low and $high$. Write a function to solve this problem. Find the root using binary search.

Requirements

- Complete the given Python function $solver(f, low, high)$ and solve for zero.

Important

- Please make sure to terminate your function. What must be done to ensure termination?
- Note that functions may have more than one root.
- The function has to return a single root. You can return any root in the defined range.
- Use the following bounds as limits: $low = -100$ and $high = 100$

Example 1 - f(x) = 2x + 3

- Input: $f1 = lambda\ x: 2 * x + 3$
- Function: $x = solver(f1, -10, 10)$
- Result: One of [-1.5]

Example 2 - f(x) = x^3 - 100x^2 - x + 100

- Input: $f1 = lambda\ x: x ** 3 - 100 * x ** 2 - x + 100$
- Function: $x = solver(f1, -10, 10)$
- Result: One of [-1.0, 1.0]

Example 3 - f(x) = 6x^3 - x^2 + 2x + 22

- Input: $f1 = lambda\ x: 6 * x ** 3 - x ** 2 + 2 * x + 22$
- Function: $x = solver(f1, -10, 10)$
- Result: One of [-1.42]

NYU SHANGHAI 上海 纽约大学

Problem 4 – Missing Element - 25 Points

A list l contains a range of n − 1 unique integers in the bounds of [1, n − 1]. One number from this sequence is not in l. Write a Python function to find the missing number. Your function has to return the missing number or $none$ if the sequence is complete.

Requirements

- Complete the given function $find\_missing(l)$
- Your algorithm has to be $O(n)$ time complex
- You are only allowed to use $O(1)$ additional space
- You can not use Python set or dict.

Important

- The largest element is always included in the test

Example 1

- Input: $l = [i\ for\ i\ in\ range(5)\ if\ i\ != 2]$
- Calculation: $missing = find\_missing(l)$
- Result: 2

Example 2

- Input: $l = [i\ for\ i\ in\ range(49)\ if\ i\ != 40]$
- Calculation: $missing = find\_missing(l)$
- Result: 40

Example 3

- Input: $l = [4, 1, 3, 0]$
- Calculation: $missing = find\_missing(l)$
- Result: 2