Домашнее Задание

Атабекян Эдгар Ваагнович Б05-151

September 2022

1 Задание

Дан указатель на начало списка (сам он не дан вам). Опишите алгоритм, позволяющий определить, есть ли в нем цикл. Гарантируется, что список конечен, указатель последнего узла указывает на NULL. Не гарантируется, что цикл в списке содержит первый узел. Время линейное, а память константная.

Решение: *(В решении слово **цикл** означает обычный цикл действий алгоритма (т.е. for, while)).* Возьмём 2 указателя P1, P2 и присвоим им значние данного нам указателя. С помощью этих указателей мы циклически пробежимся по списку, но 1-й указатель за каждый цикл пройдет по одному шагу а 2-ой по 2 шага(т.е по однаму шагу два раза в цикле). В какой то момент , либо во время первого, либо во время второго шага 2-ого указателя он либо встретиться с 1-ым указателем и это будет означать что в списке есть ЦИКЛ, либо будет равен NULL, что значит что в списке ЦИКЛА нету.

2 Задание

За какое время будет работать бинарный поиск в двусвязном списке, если разрешено выделить O(1) доппамяти.

Решение: Для начала мы завиксируем 3 указателя : START-Начало списка, MID(в начале алгоритма тоже начало списка) и END-Конец списка. Во время работы алгоритма MID делает сперва N/2 шага в право (направление последующих шагов зависит от данных списка), потом N/4 шага, N/8 шага, N/16 шага и так далее до $N/2^k$.

Получается во время всего алгоритма указатель делает всего $N/2 + N/2^2 + N/2^3 + ... + N/2^k$ шагов. k произвольное натуральное число значит максимум бинпоиск будет работать за O($\frac{\frac{N}{2}}{(1-\frac{1}{2})}$) = O(N).

Ответ: O(N).

3 Задание

Назовем гистограммой набор из N столбцов единичной ширины и высоты h_i , выровненных снизу по общей базовой линии. Найдите максимальную площадь прямоугольника, который целиком содержится в гистограмме за O(N) времени.

Решение: В начале возьмём переменную $MaxSquare$=0. Возьмём стек и указатель на первый столбец гистограммы и спомощью указателя будем «шагать» по её столбцам. И на каждом шагу проверяем эти условия:

а. Если высота указанного столбца(i-й столбец) меньше или равна высоты указанной последнего добавленного(Верхнего) элемента стека: Рекурсивно проверяем верхний элемент стека пока её высота не станет меньше или равным высоты указанного столбца.Если станет то смотрим следующий шаг ***б.*** .

Иначе снимаем последний элемент из стека и опять проверяем условие ***а.*** .

б. Если стек пустой либо высота указанного столбца(i-й столбец) больше высоты указанной последнего добавленного(Верхнего) элемента стека: Добавляем элемент с высотой h_i и индексом i на вершину стека.

в. Если i<N: Переходим на следующий столбец.

г. Если i=N: То мы получили лестницу. Теперь пока стек не станет пустым делаем следующее. На каждом шагу вынимаем верхний елемент и находим полщадь максимального прямоугольника включающий столбик с высотой и индексом из этого элемента. Для этого отнимим из N индекс предыдущего элемента из стека, и полученное умножим на высоту данного элемента который мы вынули из стека. Присвоим $MaxSquare$-у максимальное значение $MaxSquare$-а и полученного числа.

$$\text{elem} = \text{pop(stack)}$$
$$\text{square} = \text{(N-stack.top.index)*elem.height}$$

$$\text{MaxSquare} = \max\{\text{MaxSquare, square}\}$$

После того как стек стал пустым мы получили максимальную площадь нужного прямоугольника.

4 Задание

Есть N гоблинов, у i-го h_i очков здоровья. При ударе по i-му гоблину посохом ему сносят p очков, а всем остальным по q. Найдите наименьшее число ударов посохом, чтобы все гоблины умерли (то есть уровень их здоровья стал нулевым или ниже)

Решение: 1.Сначала зафиксируем переменную click = 0.

2.Сортируем массив очков здаровья гоблинов.

3.Смотрим на последний элемент h_N:

3.1.Если $h_N \leq 0$***:*** Возвращяем значение click и закончиваем.

3.2.Если $h_N > 0$***:*** а)Если $p \geq q$ тогда кликаем на h_N, пременной click прибавляем 1 и возвращяемся на 2 шаг. б) Если $p < q$ тогда кликаем на h_1, пременной click прибавляем 1 и возвращяемся на 2 шаг.

5 Задание

Для каждого префикса массива $a_1, ..., a_N$ найти подотрезок с максимальной суммой. То есть для каждого p найти подорезок с максимальной суммой в массиве $a_1, ..., a_p$. Время алгоритма должно составлять O(N).

Решение:

1. Строим префиксные суммы $b_1, ..., b_N$, где

$$b_i = a_1 + ... + a_i$$

2. Проходимся по массиву b и на каждом i-ом шагу находим минимальную префиксную сумму для i-ого префикса и сохраняем в массив min, т.е.

$$b_1 ... b_i ... b_N$$
$$min_1 ... min_i ... min_N$$
$$\text{Где } min_i = min\{b_i, min_{i-1}\}$$

3. Проходимся через массив префиксных сумм и на каждом шагу находим максимальную сумму подотрезков этого префикса таким образом,

$$max_i = b_i - min_i$$

И выводим max_i.

6 Задание

Дан массив длины N из целых чисел. Необходимо обработать Q запросов вида (l_i, r_i, b_i, d_i). Данный запрос означает следующее:

$$a_{l_i} \to a_{l_i} + b_i$$
$$\mathrm{a}_{l_i} + 1 \to a_{l_i} + 1 + (b_i + d_i)$$
$$...$$
$$\mathrm{a}_{l_i} + k \to a_{l_i} + k + (b_i + d_i k)$$
$$...$$
$$\mathrm{a}_{r_i} \to a_{r_i} + (b_i + d_i(r_i - l_i))$$

То есть прибавляем на отрезке арифметическую прогрессию. Нужно вывести массив после обра- ботки всех запросов, временная сложность: O(N + Q).

Решение: Берем массивы B[N] и D[N],так чтобы для любого i такого что $1 \leq i \leq N$ B[i]=D[i]=0 и переменную Dsum=0.

Во время обработки запросов делаем следующее:

Для каждого l_i, r_i, b_i, d_i
$$\mathrm{B[\ l}_i] := B[l_i] + b_i$$
$$\mathrm{D[\ l}_i] := D[l_i] + d_i$$
$$\mathrm{B[\ r}_i + 1] := B[r_i + 1] - b_i$$
$$\mathrm{D[\ r}_i + 1] := D[r_i + 1] - d_i * (r_i - l_i + 1)$$

Потом проходим через данный масив **a** и на каждом шагу делаем следующее:

Для каждого a_i
B[i] := B[i] + B[i-1]
X:=D[i]
D[i] := Dsum + D[i-1]
Dsum := Dsum + X
$\mathrm{a}_i := a_i + B[i] + D[i]$
И выводим a_i

7 Задание

Отсортированный массив a 1 $\leq$ a 2 $\leq$ . . . $\leq$ a N сдвинули циклически на k и дали вам по- лученный массив a N $-$k+1 , . . . , a N , a 1 , . . . , a N $-$k на вход. Считайте, что массив уже считан в память, так что читать его вам не нужно.

(a) (2 $*$ балла) Докажите, что нельзя построить алгоритм, находящий k за O(log N)

(b) (1 $*$ балл) Пусть каждый элемент повторяется не более m раз в массиве. Приведите алгоритм поиска k за O(m + log N)

Решение а: При таких массивах и сдвигах что получим в итоге массив например такого вида a, a, a, a, ... a, ... a,b,a невозможно будет бинпоиском за O(logN) найти k, а едиственный способ за такое время найти k это бинпоиск. Бинпоиском не получится потому что на каждом шагу бинпоиска мы еще должны будем проверить элементы слева и справа и найти не равный **a** элемент. А это нам добавляет по крайней мере еще $\approx N$ шагов на каждый шаг бинпоиска. В итоге выходит что будут тесты работающие за O(NlogN).

Решение б: 1.Сперва проверим если $a_1 < a_N$ то сразу выводим 0.

2.Иначе Если $_1 \geq a_N$

$min = 1$

max = N

mid = (min + max) / 2

2.1.Пока $min! = max - 1$ проверяем

2.1.1.Если $a_{mid} \leq a_{min}$

$max := mid$

2.1.2.Иначе $min := mid$

2.1.3.Если $mid = (min + max)/2$

2.2.Проверяем если $a_{mid} = a_{mid+1}$ то добавляем mid-у 1 столько раз чтобы $a_{mid} > a_{mid+1}$

2.3.Выводим mid