Домашнее Задание по алгоритмам №5

Павливский Сергей Алексеевич , 873

10.03.2019

Задание №1

Пусть исходный массив - A[i] , где для любого k : A[k] = 0 или 1 .

Заведем массив B из двух элементов : B[0] и B[1].

Пройдем все элементы массива A , при этом встречая элемент с некоторым значением k , увеличиваем значение элемента B[k] на 1 .

Пройдя все элементы A , присвоим первым B[0] элементам массива A значение 0 , а всем остальным значение 1 . Полученный массив A будет отсортированной версией исходного массива A .

Асимптотика :

Алгоритм совершает два прохода по n элементам , на каждом шаге он совершает O(1) операций . Значит итого алгоритмическая сложность = O(c * 2n) = O(n) , то есть алгоритм сортировки линейный .

Корректность :

Так как массив A состоит только из 0 и 1 , мы изменяем B[A[i]] , а среди элементов B есть B[0] и B[1] , то все элементы A будут учтены в B; значение B[0] будет равно количеству 0 в массиве A , аналогично для B[1] и 1; для любого из первых B[0] элементов A после заполнения (равных 0) выполняется : для любых x < y, принадлежащих от 0 до B[0] - 1 , верно , что A[x] <= A[y] ; аналогично для любых x1 < y1, принадлежащих от B[0] до n - 1 , верно , что A[x1] <= A[y1]; также для любого x2 , принадлежащего от 0 до B[0] - 1 , и y2 , принадлежащего от B[0] до n - 1 , верно , что A[x2] <= A[y2] ; значит в результате массив отсортирован по неубыванию , т.е. алгоритм корректен .

Задание №2

Отсортируем отрезки по возрастанию левой границы , получим упорядоченную последовательность вложенных отрезков . Заведем переменные a, b, c, d. В переменную d и a занесем правую границу и левую границу соответственно первого элемента из последовательности отрезков , отсортированных по возрастанию левой границы . Далее переберем $\frac{2n}{3}$ левых границ отрезков , левую границу на которой остановились занесем в b , соответствующую ей правую границу отрезка в c. Тогда все точки , принадлежащие промежуткам [a ; b) или (c ; d] - искомые .

Асимптотика :

Сортировка отрезков по левой границы при помощи алгоритма быстрой сортировки выполняется за O(n * log n) . Далее перебор границ от первой до $\frac{2n}{3}$ выполняется за $\frac{2n}{3}$ операций , Присваивание переменным выполняется за O(1) . Итого : O(n * log n) + O($\frac{2n}{3}$) + O(1) = O(n * log n) операций .

Корректность :

После сортировки все отрезки окажутся упорядоченными так , что если i > k , то любая точка принадлежащая отрезку[i] также принадлежит отрезку[k] . Значит для любого x >= $\frac{2n}{3} + 1$ количество отрезков , которые его содержат , >= $\frac{2n}{3} + 1$, т.е. != $\frac{2n}{3}$; аналогично , для любого y < $\frac{2n}{3}$ количество отрезков , которые его содержат , < $\frac{2n}{3}$, т.е. != $\frac{2n}{3}$. То есть только при x $\in [\frac{2n}{3}; \frac{2n}{3} + 1)$ его содержат $\frac{2n}{3}$ отрезков . Данный алгоритм и находит данные промежутки .

Задание №3

Разобьем исходные n элементов на $[\frac{n}{7}]$ подгрупп, в каждой из которых выберем медиану. Хотя бы половина из медиан >= медианы x, то есть в $[\frac{n}{7}]$ подгруппах хотя бы $4 * ([\frac{1}{2}[\frac{n}{7}]] - 2) >= \frac{2n}{7}$ - 8 элементов >= x. Также хотя бы $\frac{2n}{7}$ - 8 элементов меньше x. Тогда Select рекурсивно вызывается не более чем для $\frac{5n}{7}$+8 элементов. Пусть при n < 10 T(n) = O(1), при n >= 10 T(n) = T($[\frac{n}{7}]$) + T($\frac{5n}{7}$+ 8) + O(n). Докажем, что T(n) = O(n). База индукции выполняется при достаточно больших c. Если выполняется предположение индукции:

T(n) <= c$[\frac{n}{7}]$ + c($\frac{5n}{7}$ + 8) + an , a = const

T(n) <= $\frac{cn}{7}$ + c + $\frac{5cn}{7}$ + 8c + an ? cn

$\frac{cn}{7}$ - 9c - an ? 0

Пусть c = 21a, подставим: 21an - 9 * 21a - an = 20an - 9 * 21a ? 0

20n - 189 ? 0

Так как n >= 10, то 20n - 189 > 0 , т.е. $\frac{cn}{7}$ - 9c - an < 0, т.е. T(n) =O(n).

Задание №4

Так как ax + b кратно M , то данное выражение представимо в виде ax + b = My , где y - целое . Тогда перепишем равенство в виде ax - My = -b . Получаем Диофантово уравнение . По расширенному алгоритму Евклида находим НОД (a, -M) за полиномиальное время (проводятся только операции вычитания , которые не могут дать не полиномиальное время) . Далее если b кратно НОД(a, -M) (это также проверяется вычитанием за пполиномиальное время) , то домножаем оставшиеся после использования расширенного алгоритма Евклида числа на $\frac{b}{(a,-M)}$ (также за полномиальное время по алгоритму Карацубы) и получаем решение на x , в противном случае решений нет . Так как все шаги выполнялись за полномиальное время , то и весь алгоритм работает за полиномиальное время , что и требовалось .

Задание №5

a)

В худшем случае (когда мы разбиваем массив так , что один подмассив содержит n-1 элемент , а второй подмассив содержит 1 элемент) получаем соотношение на время работы :

T(n) = T(n - 1) + T (n)

Из этого соотношения видно , что в худшем случае вызывается n рекурсивных вызовов , т.е. глубина стека рекурсивных вызовов = n .

2)

Если брать в качестве опорного элемента элемент в середине рассматриваемого массива , то получаем соотношение :

T(n) = 2 * T($\frac{n}{2}$) + T (n)

Из этого соотношения видно , что глубина стека рекурсивных вызовов будет = log n .

Задание №6