Домашнее задание 4

Христолюбов Максим, 771

Задание 6 (Итеративный MergeSort)

Пусть для простоты в исходном массиве A 2^k элементов.На исходный массив можно взглянуть как на массив упорядоченных массивов длины 1. Изначально $i = 1$. Следующий цикл выполняется до тех пока не будет выполнятся $l = 2^k$.

Производим сортировку слиянием попарно массивов с началом в $A[2^l p]$ и $A[2^i p + 2^{i-1}]$ длины $l = 2^{i-1}$ (на первом шаге каждый из них состоит из одного элемента $l = 1$), $p = 0,1,2,...,(2^{k-i} - 1)$. Получим отсортированные массивы $A[2^i p]$ длины $l = 2^i$, $p = 0,1,...,(2^{k-i-1} - 1)$.

На каждой следующей итерации $i = i + 1$, т.е. уже полученные отсортированные массивы в новом обозначении будут выглядеть так: массивы с началом в $A[2^{i-1}p]$ длины $l = 2^{i-1}$, $p = 0,1,...,(2^{k-i} - 1)$, что абсолютно соответствует входным данным в первой итерации. Будем продолжать итерации пока не настанет момент $i = k$. В итоге получим отсортированный массив $A[0]$ длины $l = 2^k$. Если в A кол-во элементов - не степень 2, то в последнем подмассиве будет меньше чем 2^{i-1} перед слиянием, что не мешает алгоритму.

Задание 1

$F(3,5) = ((1 \cdot 3)^2)^2 \cdot 3 = 3^5 = 243$

Это функция быстрого возведения x в степень m. Действительно, как было показано в одном из предыдущих заданий для того чтобы быстро возвести в степень нужно разложить m так, чтобы было максимальное кол-во умножений на 2 (возведение x^k в квадрат) и минимальное кол-во добавлений 1 (умножение x^k на x). Это разложение можно получить из двоичной записи m. Начинаем с 1, если в разряде стоит 1, то нужно возвести число в квадрат и умножить на x, что и делается, т.к 1 переводится в SX, что приводит к возведению в квадрат и умножению, а если в разряде 0, то просто возвести в квадрат, т.е. S.

Так как на каждом шаге выполняется возведение в квадрат и в худшем случае еще умножение, а всего шагов - двоичная запись числа m, то сложность алгоритма $\Theta(\log m)$.

Задание 2

Найдем $\frac{2n}{3}$ и $\frac{2n}{3} - 1$ порядковую статистику r_i - их номер k и p. Точки принадлежащие ровно $\frac{2n}{3}$ отрезкам находятся между r_p и r_k, а также между l_k и l_p, так как если упорядочить r_i, то внутри отрезка с концом в r_k ($\frac{2n}{3}$ порядковой статистики) лежит $\frac{2n}{3} - 1$ вложенных отрезков. Эти 2 порядковые статистики можно найти за линейное время, сложность алгоритма $\Theta(n)$.

Задание 3

$T(n) = T(\frac{n}{7}) + T(\frac{5n}{7}) + Cn$, так как на следующем шаге рассматривается массив длины не больше $\frac{5n}{7}$, поскольку искомая порядковая статистика находится правее(и левее) чем $4 \cdot \frac{n}{2 \cdot 7}$ элементов.

$T(n) = O(n)$, докажем по индукции: $T(n) = T(\frac{n}{7}) + T(\frac{5n}{7}) + Cn = O(\frac{n}{7}) + O(\frac{5n}{7}) + Cn = O(n)$.

Задание 4

Перебираем все элементы массивы по порядку, если элемент равен 0, то записываем его в конец будущего отсортированного массива (изначально он пуст), после этого проходим по данному в условии массиву еще раз и если элемент равен 1, то записываем его в конец результирующего массива.

Задание 5

$ax + b \equiv 0(\mod M)$

$ax + kM = -b$

Решить данное уравнение можно за $O(n^3)$ по алгоритму евклида (если производить деление за $O(n^2)$). После этого мы будем знать x. n^3 - полином.

Задание 6

В худшем случае массив будет разбиваться каждый раз опорным элементом на часть равную остальному массиву и пустую часть (опорный элемент при partition будет вставать в конец или начало) и потребуется n рекурсивных вызовов.

Для того чтобы в худшем случае глубина стека было $\Theta(\log n)$ за опорный элемент можно брать медиану, тогда массив всегда будет делится на 2 почти равные части.