10/08

10:40 Повторение

Разбор хендаута по пунктам с выделением ключевых моментов (assert, порядок роста).

NB:: порядок роста — часто непонятная штука, стоит разбирать с примерами.

10:50 пробуем править программу первый-последний самостоятельно. Решаем задачу «первый-первый».

NB:: исправляли ассерт, что было не нужно (порядка трех минут)

Разбор у доски —

1. выясняем, что ассерт менять не нужно.
2. меняем инициализацию p — p = 1
3. неравенство в while меняется на больше,
4. условие на p меняется на N+k
5. p = p-1 => p = p+1
6. after => before (условие вставки)

NB:

Прокрутка алгоритма на доске. Модельные данные: А= [20, 22, 24, 26, 29], B = [10, 12, 18, 23, 30]

Подсчет количества сравнений для вставки каждого элемента (для вставки 23 и 30 требуется 8 и 9 сравнений)

Голословное утверждение, что если у нас есть числа порядка n+n, то оценка ~n\*m

Просьба проверить не делается ли лишних проверок.

Студент: можно начинать не с начала, а с того места, где остановились.

Задание: подумать как нужно изменить алгоритм.

Правим вместе. Идея — инициализацию p изменить на что-то другое. Добавляем строчку pnew := p после энд вайл. Выясняем, что инициализация должна быть p := p + 1. Убираем pnew и ставим инициализацию p:=p+1

ВНИМАНИЕ: на занятии забыли, что теперь p нужно инициализировать перед циклом for. Разобрать на повторении.

Замечание: важно сказать, где и когда меняется p. Кроме того, стоит предложить

NB:: возможно, стоит сказать, что максимальная длина массива N+M — при каждом выолнении сравнения длина массива увеличивается на 1 (max len(C) = N+M)

Оцениваем время работы.

3. Задача на слияние 4 массивов одинаковой длины.

Студент: объединяем по два массива в пары, сливаем в пары, сливаем в один (рисунок на доске вставить)

МР у доски: бинарное дерево с 8 или 16 листьями

В каждом слове суммарное количество операций равно сумме длин массивов.

NB: если исходные массивы были длины 1, мы получили полноценный алгоритм сортировки

Время его работы — исходное количество элементов \* число слоев. T=N\*S, S ~logN => T=N\*logN

Замечание 1. На занятии не было подчеркнута роль балансировки - почему надо имено сливать по парам, а не прибавлять к существующему массиву еще один кусок (тогда число слоев N, T ~N^2).

Замечание 2. Итоговый алгоритм не был выписан, будет разобран на следующем занятии.

4. Структура данных. Списки.

Сказано, что в массивы плохо вставлять, поэтому на массивах мерджсорт не будет работать с нужным временем.

Единица хранения в списке - «доминошка». В одной ячейке доминошки хранятся данные, во второй — ссылка на другую доминошку/индикатор конца списка. Существует специальная доминошка — начало списка, которая не указывает ни на какой элемент списка. Пример: список А и B из предыдущей задачи.

При работе со списком есть указатель, который указывает на 1 *текущий* элемент списка. В начале

4.2 Структуры данных.

С.Д. - общее понятие чего-то, где хранятся данные (у нас список). Для С.Д. Известно, какие операции она допускает (напр., для массива — взять элемент A[k] и записать в A[k] выданное значение. В обоих случаях надо указать индекс элемента k)

Для списка возможны следующие операции —

* взять значение текущего элемента
* записать в текущий элемент заданное значение
* перейти к следующему элементу
* проверить, не является ли текущий элемент последним
* объявить заголовок текущим элементом (встать в начало списка)
* вставить после текущего новый элемент с заданным значением
* присвоить новое значение указателю

Пример выполнения слияния на списках

NB: Было сделано коряво, сделать хорошие картинки

Было сказано про переход к следующему, заголовок, конец списка. При повторении нужно аккуратно разобрать, в том числе про указатели. Обсудить вставку одного списка в другой и сцепление списков. Можно поговорить про тип указателей.

4.3 Операция поиска, сложность ее выполнения как на массивах, так и на списках (линейное время).

Для ускорения поиска нужен «ветвящийся список» (дерево). To be continued.