Домашнее Задание по алгоритмам №13

Павливский Сергей Алексеевич , 873

04.05.2019

Задание 1.

На вход задачи подаётся число n и последовательность целых чисел $a_1, ..., a_n$. Необходимо найти номера i и j ($1 \leqslant i < j \leqslant n$), такие что сумма $\sum_{k=i}^{j} a_k$ максимальна.

1. Постройте линейный алгоритм, решающий задачу.

2. Постройте онлайн-алгоритм (достаточно выполнить только этот пункт).

Решение

2) Заведем переменные i , j , sum . Изначально i = j = 1 , sum = A[1] (первый считанный элемент) . Каждый раз когда считываем k-й элемент делаем проверку :

- если sum + A[k] > A[k] , то j увеличивается на 1 , в противном случае делаем i = j = k , присваиваем sum значение A[k] .

Ответом будут значения переменных i и j после обработки всех входных данных .

Ассимтотика : каждый элемент рассматривается по одному разу , на каждом шаге делается const число операций , значит алгоритм линейный от длины входа , что и требовалось .

Корректность :

Докажем по индукции . База для одного считанного элемента очевидна . Индукционный переход : пусть верно для первых n членов . Докажем , что верно и для n + 1 . Предположим противное . Пусть алгоритм выдает корректный результат для n членов , но некорректный для n + 1 . Но так как рассматриваемые члены последовательности образуют непрерывный по индексам отрезок , а также так как если ответ и мог изменится , то новая последовательность содержит A[n + 1] , то это либо сама A[n + 1] , либо наибольшая по сумме непрерывная подпоследовательность первых n членов . Но так как на первых n членах алгоритм корректен , то наибольшая по сумме непрерывная подпоследовательность первых n членов имеет j = n . Но тогда приходим к противоречию , так как , исходя из выше сказанного , алгоритм корректен и для n + 1 члена .

Задание 2.

Фирма производит программное обеспечение для банкоматов разных стран мира. Иногда, купюры какого-то вида заканчиваются и банкомату нужно определить, возможно ли выдать клиенту требуемую сумму. Вход задачи: число n, номиналы купюр $v_1, ...v_n$ и сумма клиента s.

1. Постройте алгоритм, решающий задачу за O(ns).

Решение

Воспользуемся жадным алгоритмом : будем вычитать из s номинал самой большой купюры до тех пор , пока оставшееся значение не станет меньше номинала . Дальше из остатка будем вычитать номинал второй по величине купюры , и так далее до тех пор , пока не останется 0 , или остаток будет меньше чем наименьший номинал купюр . В первом случае мы можем отмотать операции в обратном порядке и выдать клиенту купюры на сумму s , иначе ответ , что выдать данную сумма нельзя .

Ассимптотика : в худшем случае мы делаем O(n) шагов , а на каждом O(s) операций , то есть всего O(ns) , что и требовалось .

Задание 3.

Рассмотренный нами алгоритм вычисления расстояния редактирования строк длины m и n заполняет таблицу размера O(mn). При больших m и n такому алгоритму просто не хватит памяти. Как можно обойтись меньшим объёмом памяти?

1. Допустим, что нас интересует только расстояние редактирования, но не соответсвующее оптимальное выравнивание. Покажите, что тогда в каждый момент не нужно хранить всю таблицу и можно обойтись объёмом памяти O(n).

2. Теперь допустим, что мы хотим найти и оптимальное выравнивание. Легко видеть, что это эквивалентно поиску кратчайшего пути из вершины (0, 0) в вершину (n, m) в соответсвующем графе: вершины графа — клетки таблицы, а стоимость ребра — 1 или 0, в зависимости от конкретного перехода. Любой такой путь должен проходить через вершину (k, m/2) для некоторого k. Модифицируйте алгоритм поиска расстояния редактирования, чтобы он заодно выдавал и k. Считайте, что m — степень двойки.

Решение

Задание 4.

Постройте алгоритм, который, получив на вход числа n и k, выводит все последовательности длины k целых чисел от 1 до n.

Решение

Заведем массив A длины k + 1 . Изначально все элементы равны 1 . Изначально a = 0. На каждом шаге :

```
{
A[0]++;
for (j = 0; j < a && A[j] == k + 1; j++)
A[j + 1]++;
A[j] = 1;
if (A[a] == k + 1)
A[a] = 1;
a++;
puts(A);
}
```

Делаем это пока a < k . В итоге имеем все последовательность длины k в диапазоне от 1 до n .

Задание 5.

В одном языке программирования операция разрезания строки на две части реализовано через копирование. Разрезать строку w = uv на две части u и v будет стоить длину строки O(|w|), где |w| — длина строки w, вне зависимости от длин строк u и v. Постройте алгоритм, который получив на вход строку w = $u_1u_2...u_k$ и номера позиций $i_1 = |u_1|, i_2 = |u_1| + |u_2|, ..., ik = |u_1| + ... + |u_k|$ строит последовательность разрезов строки w, которая приводит к разрезанию w на подстроки $u_1, ..., u_k$ и использует для этого минимальное число операций. Заметьте, что в зависимости от порядка разрезов меняется общее число операций: если строку длины 20 нужно разрезать на строки с номерами позиций 3, 10, то отрезав сначала строку в позиции 3 будет затрачено 20 + 17 = 37 операций, а отрезав сначала строку в позиции 10, будет потрачено 20 + 10 = 30 операций.

Решение

Задание 6.

Алиса и Боб играют в следующую игру. Есть n карт, на i-й карте записано число $1 \leqslant a_i \leqslant n$ (числа могут повторяться!). Игроки ходят по очереди, первой ходит Алиса. На каждом ходу игрок выбирает карту и выбрасывает её. Кроме того, он выбрасывает все карты, на которых число меньше, чем на выбранной. Формально, если игрок выбирает карту с номером i, он выбрасывает эту карту, а также каждую карту с номером j, такую что $a_j < a_i$. Игрок проигрывает, если он не может сделать ход, то есть если не осталось карт.

1. Постройте алгоритм, который определяет победителя, если игроки играют оптимально.

2. Постройте жадный алгоритм, решающий эту задачу.

Решение

1.

Перебираем карточки по убыванию номинала . Ищем первое значение , карточек с которым нечетное количество . Если такое находится , то победит Алиса : просто убирает одну карточку этого номинала , автоматически убираются все меньшие . Тогда какую бы карточку дальше не убирал Боб , Алиса сможет убрать еще одну карточку такого же типа , но нечетного номера по счету среди равных , а тот кто первый уберет карточку максимального значения и нечетного номера среди равных , тот и победит (просто по четности) . Если же карточек всех номиналов четное количество , то победит Боб , потому что сможет после каждого хода Алисы убирать карточку нечетного номера , но того же номинала , а значит на каждом номинале он будет первым убирать карточку нечетного номера , то есть из вышесказанного победит .

2.

Пусть оптимальным ходом на каждом шаге будет брать карточку с наибольшим номером . Тогда победитель будет определяться четностью количества карточек с наибольшим на момент начала игры номером :

- если их нечетно , то побеждает Алиса
- если их четно , то побеждает Боб

Задание 7.

Дан выпуклый n-угольник (заданный координатами своих вершин в порядке обхода). Его разрезают на треугольники диагоналями, для чего необходимо n-2 диагонали (это можно доказать индукцией по n). Стоимостью разрезания назовём сумму длин всех использованных диагоналей. Постройте полиномиальный алгоритм, который находит минимальную стоимость разрезания. (Перебор не подходит, так как число вариантов не ограничено многочленом.)

Решение