# Практика по алгоритмам

Всеволод Опарин, Алексей Давыдов

Осень, 2014

# 1 Практика 1. Асимптотика

## 1.1 Всопомогательные факты

1. Сумма арифметической прогрессии ($a_{i+1} = a_i + d$): $a_1 + a_2 + \cdots + a_n = \frac{a_1+a_n}{2} \cdot n$.
2. Сумма геометрической прогрессии: $1 + p + \cdots + p^{n-1} = \frac{p^n-1}{p-1}$, $\sum_{i=0}^{\infty} p^i = \frac{1}{1-p}$. при $0 < p < 1$.
3. Гармонический ряд: $\sum_{i=1}^{n} \frac{1}{n} = \log n + O(1)$, $\sum_{i=1}^{n} \frac{1}{n^{1+\epsilon}} = O(1)$.
4. Оценки на суммы:
   (a) Мажорирование:
   $$\sum_{i=0}^{n} i \le \sum_{i=0}^{n} n = O(n^2).$$
   (b) Разделение на суммы
   $$\sum_{i=0}^{n} i \ge \sum_{i=\lfloor n/2 \rfloor}^{n} \lfloor n/2 \rfloor = \Omega(n^2).$$
   (c) Интегрирование
   $$\Omega(n^2) = \int_0^n x\,dx \le \sum_{i=1}^{n} i \le \int_1^{n+1} x\,dx = O(n^2).$$

## 1.2 Практика

Асимптотика:

1. $n^k$ и $c^n$ сравнить по $O$-нотациями.
2. $p(n) = a_0 + a_1 x + \cdots + a_d x^d$. Для каких $k$ какие отношения между $p(n)$ и $n^k$.
3. $\max(f(n), g(n)) = \Theta(f(n) + g(n))$?
4. $2^{n+1} = O(2^n)$? $2^{2n} = O(2^n)$?
5. $f(n) = O(f(n)^2)$?
6. $f(n) + g(n) = \Theta(\min(f(n), g(n)))$?
7. $f(n) = O(g(n)) \Rightarrow \log f(n) = O(\log g(n))$?
8. $f(n) = O(g(n)) \Rightarrow 2^{f(n)} = O(2^{g(n)})$?

9. $f(n) = f(n/2)$?
10. $f(n) + o(f(n)) = \Theta(f(n))$?
11. $\log n! = \Theta(n \cdot \log n)$?

Суммы:

1. $\sum_{i=1}^{n} \frac{1}{n} = \Omega(\log n)$ через интегралы и разбиения на части.
2. $\sum_{k=1}^{n} \frac{1}{k^2} = O(1)$.
3. $\sum_{k=0}^{\log n} \lceil n/2^k \rceil$.
4. $\sum_{k=1}^{n} \frac{1}{2k-1} = ln(\sqrt{n}) + O(1)$.
5. $\sum_{k=0}^{\infty} (k-1)/2^k = ?$.
6. $\prod_{k=1}^{n} (2 \cdot 4^k)$.
7. $\prod_{k=2}^{n} (1 - 1/k^2)$.

### 1.3 Домашнее задание

**Дедлайн: 23.59 11.09.2014**

Группа Давыдова решает все суммы в качестве домашнего задания.

1. Посчитать произведения
   (a) $\prod_{k=1}^{n} (2 \cdot 4^k)$.
   (b) $\prod_{k=2}^{n} (1 - 1/k^2)$.
2. Заполнить табличку.

| $A$ | $B$ | $O$ | $o$ | $\Theta$ | $\omega$ | $\Omega$ |
|---|---|---|---|---|---|---|
| $\lg^k n$ | $n^\epsilon$ | | | | | |
| $n^k$ | $c^n$ | | | | | |
| $\sqrt{n}$ | $n^{\sin n}$ | | | | | |
| $2^n$ | $2^{n/2}$ | | | | | |
| $n^{\lg m}$ | $m^{\lg n}$ | | | | | |
| $\lg(n!)$ | $\lg(n^n)$ | | | | | |

3. Упорядочить функции в порядке возрастания.

$$\begin{array}{cccccc}
\lg(\lg^* n) & 2^{lg^* n} & (\sqrt{n})^{\lg n} & n^2 & n! & (\lg n)! \\
(3/2)^n & n^3 & \lg^2 n & \lg n! & 2^{2^n} & n^{1/\lg n} \\
\ln\ln n & \lg^* n & n \cdot 2^n & n^{\lg\lg n} & \ln n & 1 \\
2^{\ln n} & (\lg n)^{\lg n} & e^n & 4^{\lg n} & (n+1)! & \sqrt{\lg n} \\
\lg^* \lg n & 2^{\sqrt{2\lg n}} & n & 2^n & n\lg n & 2^{2^{n+1}}
\end{array}$$

Примечание: $\lg^*(n) = \begin{cases} 0 & \texttt{если } n \le 1; \\ 1 + \lg^*(\lg n) & \texttt{иначе.} \end{cases}$

# 2 Практика 2. Линейные алгоритмы

## 2.1 Практика

1. Реализовать стек с операциями `PUSH`, `POP`, `MAX` при условии, что каждая операция работает за константное время.

2. Реализовать очередь с операциями `PUSH`, `POP`, `MAX` при условии, что каждая операция работает за константное время.

3. Дана последовательность $a_1, a_2, \cdots, a_n \in \mathbb{N}$ и $S \in \mathbb{N}$. Найти $l, r$ $(1 \leq l \leq r \leq n)$ такие, что сумма $\sum_{i=l}^{r} a_i = S$. Задачу требуется решить за линейное от $n$ время.

## 2.2 Домашнее задание

**Дедлайн: 18 сентября, 23.59**

1. Дана последовательность $a_1, a_2, \cdots, a_n \in \mathbb{Z}$. Найти $l, r$ $(1 \leq l \leq r \leq n)$ такие, что сумма $\sum_{i=l}^{r} a_i$ была бы максимальной. Задачу требуется решить за линейное от $n$ время.

2. Дана скобочная последовательность, составленная из скобок `'('`, `')'`, `'['`, `']'`, `'{'`, `'}'`. Последовательность называется корректной, если каждой открывающей скобке соответствует закрывающая скобка того же типа, и соблюдается вложенность. Примеры, `([{}])` и `()()` – корректные, а `[)` и `[(])` – нет.

   Предоставьте алгоритм, который проверяет корректность последовательности за линейное время.

3. Дан массив целых чисел $a_i$. Придумайте структуру данных, которая бы умела отвечать на запросы вида: "По данным $l$ и $r$ вернуть $\sum_{i=l}^{r} a_i$." за $O(1)$.

   Разрешается сделать предподсчет за $O(n)$. Значения в массиве не меняются.

4. Дано число, представленное $n$ цифрами в десятичной записи без ведущих нулей. Из числа требуется вычеркнуть ровно $k$ цифр так, чтобы результат был бы максимальным. Задачу требуется решить за линейное от $n$ время.

   **Внимание!** $k$ подается на вход и может быть порядка $n$. Решение за $O(kn)$ приравнивается к квадратичному от $n$.

5. (*) Дана последовательность объектов $a_1, a_2, \cdots, a_n$. Над объектами определена операция сравнения. Известно, что в последовательности есть элемент присутствующий не менее чем $\left\lfloor \frac{n}{2} \right\rfloor + 1$. Требуется найти элемент $a$ за линейное время и константу дополнительной памяти при условии, что последовательность задана форвард-итератором.

6. (*) Дана последовательность $a_1, a_2, \cdots, a_n \in \mathbb{N}$. Найти $l, r$ $(1 \leq l \leq r \leq n)$ такие, что значение $(r - l + 1) \min_{i \in [l,r]} a_i$ было бы максимально. Задачу требуется решить за линейное от $n$ время.

# 3 Практика 3. Разделяй-и-властвуй

## 3.1 Вспомогательные факты

1. Пусть $a \geq 1$ и $b > 1$ — константы, $f(n)$ — функция, $T(n)$ определено при неотрицательных $n$ формулой
$$T(n) = aT(n/b) + f(n),$$
где под $n/b$ понимается либо $\lceil n/b \rceil$, либо $\lfloor n/b \rfloor$. Тогда
   (a) если $f(n) = O(n^{\log_b a - \epsilon})$ для некоторого $\epsilon > 0$, то $T(n) = \Theta(n^{\log_b a})$;
   (b) если $f(n) = \Theta(n^{\log_b a})$, то $T(n) = \Theta(n^{\log_b a} \lg n)$;
   (c) если $f(n) = \Omega(n^{\log_b a + \epsilon})$ для некоторого $\epsilon > 0$ и если $af(n/b) \leq cf(n)$ для некоторой константы $c < 1$ и достаточно больших $n$, то $T(n) = \Theta(f(n))$.
2. Комментарий к задачам про оценку асимтотики. Можно считать, что для некоторой константы $n_0$ и всех $n \leq n_0$ верно, что $T(n) = 1$.
3. **Утверждение.** Пусть $y(n)$ – монотонная неограниченная функция, и $f(n) = O(g(n))$. Тогда $f(\,y(n)\,) = O(g(\,y(n)\,))$.

   **Доказательство.** Нам известно, что для некоторой константы $c$ и $n_0$, $f(n) \leq c \cdot g(n)$ при $n \geq n_0$. Заметим, что, начиная с некоторой величины $n'$, $y(n) > n_0$. Значит, $f(\,y(n),\,) \leq c{\cdot}g(\,y(n)\,)$, начиная с $n'$. Значит, $f(\,y(n)\,) = O(g(\,y(n)\,))$.
4. Разбор задачи $T(n) = 2 \cdot T(\sqrt{n}) + \log n$.

   Проведем замену переменных. Пусть $y = \log n$. Тогда $T(2^y) = 2 \cdot T(2^{\frac{y}{2}}) + y$. Пусть $T'(y) = T(2^y)$. Тогда, для $T'$ справедлива следующая рекурентная формула
$$T'(y) = 2 \cdot T'(y/2) + y.$$
   По теореме о рекурентных соотношениях получаем, что $T(2^y) = T'(y) = O(y \cdot \log y)$. Подставим $y = \log n$. По предыдущему пункту получим, что $T(2^{\log n}) = T(n) = O(\log n \cdot \log\log n)$.

## 3.2 Практика

1. Определить асимптотику.
   (a) $T(n) = 2 \cdot T(\lfloor \frac{n}{2} \rfloor + 17) + n$.

## 3.3 Домашнее задание

**Дедлайн: 25 сентября, 23.59**

1. Определить асимптотику.
   (a) $T(n) = T(\lceil \frac{n}{2} \rceil) + 1$.
   (b) $T(n) = T(a) + T(n - a) + n$ для произвольной константы $a$.
   (c) $T(n) = T(\alpha \cdot n) + T((1 - \alpha) \cdot n) + n$ для произвольной константы $\alpha \in (0, 1)$.
   (d) $T(n) = 4 \cdot T(\lfloor \frac{n}{2} \rfloor) + n^k$ для $k \in \{1, 2, 3\}$.
2. Определить ассимптотику $T(n) = 2 \cdot T(\lfloor \log n \rfloor) + 2^{\log^* n}$.
3. Есть $k$ отсортированных массивов. В сумме массивы содержат $n$ элементов. Слить массивы за $O(n \log k)$.

4. Назовем массив `A[1..n]` унимодальным, если он сначала возрастает, а потом убывает. Строго говоря, существует такое $m \in [1, n]$, что

   - `A[i]` $<$ `A[i + 1]` для $1 \le i < m$;
   - `A[i]` $>$ `A[i + 1]` для $m \le i < n$.

   Элемент с номером $m$ назовем *пиком*.

   (a) Постройте алгоритм для поиска пика за $O(\log n)$.

   (b) Дан выпуклый многоугольник на плоскости из $n$ вершин. Вершины заданы в порядке обхода по часовой стрелке. Никакие три подряд идущие вершины не лежат на одной прямой. Требуется найти минимальный прямоугольник со сторонами, параллелльными осям координат, содержащий данный многоугольник (bounding box) за $O(\log n)$.

5. * Дан массив из $n$ различных элементов. Требуется найти число инверсий за $O(n \log n)$ (число инверсий в массиве $a$, это число таких пар $(i, j)$, что $i < j$, но $a[i] > a[j]$).

6. * Стуктура данных файл последовательного доступа поддерживает следующие операции:

   - *Read()*: чтение числа из файла на текущей позиции и перевод позиции вперед на 1 элемент.
   - *Write(x)*: запись числа в файл в текущую позицию и перевод позиции вперед на 1 элемент.
   - *Rewind()*: перевод позиции на начало файла.

   Требуется отсортировать файл за $O(n \log n)$ используя $O(1)$ памяти и два дополнительных файла.

# 4 Практика 4. Сортировки

## 4.1 Практика

1. Даны два массива $a$ и $b$ длины $n$, сгенерировать все попарные суммы $a_i + b_j$ в сортированном порядке.

   (a) За $O(n^2 \log n)$.

   (b) За $O(n^3)$ с использованием $O(n)$ дополнительной памяти.

   (c) За $O(n^2 \log n)$ с использованием $O(n)$ дополнительной памяти.

2. В свободное время Анка-пулеметчица любит сортировать патроны по серийным номера. Вот и сейчас она только разложила патроны на столе в строго отсортированном порядке. Но тут Иван Васильевич распахнул дверь с такой силой, что все патроны на столе подпрыгнули и немного перемешались. Оставив ценные указания, Иван Васильевич отправился восвояси. Как оказалось, патроны перемешались не сильно. Каждый патрон отклонился от своей позиции не более чем на $k$. Всего патронов $n$. Помогите Анке отсортировать патроны.

   (a) Отсортируйте патроны за $O(nk)$.

   (b) Отсортируйте патроны за $O(n + I)$, где $I$ — число инверсий.

   (c) Докажите нижнюю оценку на время сортировки $Omega(n \log k)$.

   (d) Отсортируйте патроны за $O(n \log k)$.

3. Дано $n$ точек на плоскости. Никакие три не лежат на одной прямой. Соединить точки

   (a) $(n-1)$–звенной ломаной без самопересечений (не замкнутой);

   (b) $n$–звенной ломаной без самопересечений (замкнутой)

   за $O(n \log n)$.

4. Дан набор из $n$ отрезков $[a_i, b_i]$. Числа $a_i$, $b_i$ — вещественные.

   (a) Найти такое вещественное $x$, что $|\{i\colon x \in [a_i, b_i]\}|$ максимально.

   (b) Длину объединения отрезков.

   (c) Для каждого $k$ определими множество точек $S_k$, покрытых ровно $k$ отрезками. Множество $S_k$ можно представить как набор непересекающихся отрезков. Найти суммы длин этих отрезков для каждого $k$.

   за $O(n \log n)$

5. Даны два массива $a$ и $b$ одинаковой длины.
   Нужно найти такую перестановку $p$, что $\sum_{i=1}^{n} a_{p_i} b_i \to \max$. Решение обосновать.

### 4.2 Домашнее задание

**Дедлайн: 2 октября, 23.59**

1. Дано бинарное дерево: $Tree ::= Node(Tree, Tree)|Empty$ (эта запись означает, что дерево — это либо вершина с парой потомков-деревьев, либо особое значение $Empty$). Определим функцию $\mathbf{rank}(x)$ следующим образом:

   - $\mathbf{rank}(Empty) = 0$
   - $\mathbf{rank}(Node(left, right)) = min(\mathbf{rank}(left), \mathbf{rank}(right)) + 1.$

   Назовем бинарное дерево *скошенным влево (левацким)*, если для его вершин выполнено следующее свойство:
   $$\forall_{x=Node(left,right)}\mathbf{rank}(left) \geq \mathbf{rank}(right).$$

   *Скошенная влево (левацкая) куча* — это скошенное влево дерево, в вершинах которого хранятся данные, для которых выполнено свойство кучи.

   - Докажите, что для любого скошенного влево дерева $|T| \geq 2^{\mathbf{rank}(T)-1}$ ($|T|$ обозначает количество вершин в дереве $T$).
   - Придумайте, как слить две скошенные влево кучи $H_1$ и $H_2$ за время $O(\log|H_1| + \log|H_2|)$.
   - Придумайте, как используя операцию слияния, построенную на предыдущем шаге, реализовать операции:
     - *Insert(x)* — добавление элемента $x$ в кучу,
     - *Pull()* — удаление минимального элемента из кучи.

2. Дано $2n-1$ коробок с черными и белыми шарами. В $i$-ой коробке находится $w_i$ белых и $b_i$ черных шаров. Всего в коробках находится $W$ белых и $B$ черных шаров. Требуется выбрать $n$ коробок таким образом, чтобы суммарное число белых шаров в них было не менее $\frac{W}{2}$, а черных не менее $\frac{B}{2}$. Решить за $O(n \log n)$.

3. На прямой расположено $n$ точек $p_1$, $p_2$, $\cdots$, $p_n$ в порядке возрастания. Каждая точка имеет вес $w_i \geq 0$. Требуется найти такую точку $q$, что $\sum_i w_i \cdot |p_i - q|$ имела бы минимальное значение. Время работы: $O(n)$.

   **Примечание.** Точки на вход подаются в порядке возрастания.

4. (*) Дан массив из $n+1$ целого числа от 1 до $n$. Массив доступен только на чтение, есть $O(1)$ дополнительной памяти. Найти за $O(n)$ любое число, которое встречается хотя бы два раза.

# 5 Практика 5. Быстрая сортировка

## 5.1 Практика

1. Робот Иван Семеныч пробует пирожки. Содержимое пирожков делится на три типа. Всего пирожков $n$. Каждый пирожок можно попробовать не более одного раза. Пирожки можно менять местами. Память у робота маленькая, $O(\log n)$.

   Помогите Ивану Семенычу отсортировать пирожки по типу: сначала первый, потом второй, потом третий. Сортировка должна работать за линейное время.

2. Разработайте алгоритм, который позволит находить $k$-порядковую статистику через медиану.

3. Рассмотрим следующий алгоритм быстрой сортировки массива `A[1..n]`.

   - Взять элемент с номером `pivot` $= \lceil \frac{n}{2} \rceil$.
   - Перенести в отдельный массив все элементы меньшие `A[pivot]`, затем равные, затем большие, сохраняя порядок.
   - Запустить алгоритм рекурсивно на элементах меньших `A[pivot]`, затем на больших.

   Будем оценивать сложность алгоритма как суммарную длину всех массивов, на которых будет рекурсивно запущен алгоритм.

   Постройте алгоритм, который получает на вход $n$ и строит наиболее сложный пример для сортировки за линейное от $n$ время.

4. (*) Куча хранится в массиве длины $n$. Родитель $p$ хранит детей в ячейках $2 \cdot p + 1$ и $2 \cdot p + 2$. Алгоритм приступает к сортировке. Сортировка устроена следующим образом.

   - Поменять первый и последний элемент кучи местами.
   - Уменьшить размер кучи на единицу.
   - Запустить `heapifyDown` на первом элементе.

   `heapifyDown` меняет родителя с наибольшим ребенком (при условии ,что ребенок больше родителя) и запускается рекурсивно. Сложность кучи определим через число вызовов функции `heapifyDown`. Требуется придумать алгоритм, который по $n$ строит самую сложную кучу. Ответ должен быть в виде перестановки из $n$ элементов. Решить за $O(n \log n)$.

5. * Придумайте структуру данных, которая поддерживает следующие операции (в оценках времени работы $n$ — текущее количество элементов) :

   - *Insert(x)* — добавление элемента $x$ за $O(\log n)$,
   - *Pull()* — удаление минимального элемента за $O(\log n$,
   - *Copy()* — копирование структуры за $O(1)$ (после копирования с каждой из копий можно независимо проделывать любую из данных трех операций).

   Подсказка: за основу можно взять левацкую кучу.

# 6 Практика 6. Медианы

## 6.1 Практика

1. Дан массив `A[1..n]` из $n$ различных чисел. Массив не обязательно отсортирован. Требуется найти $k$ ближайших к медиане элементов за линейное время. Решить для двух метрик.

   (a) По позиции в отсортированном массиве.

   $$d(x, \texttt{median}) = |\texttt{pos}(x) - \texttt{pos}(\texttt{median})|,$$

   где $\texttt{pos}(x)$ — позиция элемента $x$ в отсортированном массиве.

   (b) По значению.

   $$d(x, \texttt{median}) = |x - \texttt{median}|.$$

2. Дано два массива $a_1, a_2, \cdots, a_n$ и $p_1, p_2, \ldots p_m$. Нужно за $O(n \log m)$ для каждого $i$ найти $p_i$-ую порядковую статистику в массиве $a$.

3. Медианой называется $\lfloor \frac{n}{2} \rfloor$-я порядковая статистика. Придумайте структуру данных на основе heap, которая умеет делать Insert(x), DeleteMedian(), все операции за $O(\log n)$.

4. Дано множество, представленное в виде бинарной кучи. Найти $k$-ую статистику за:

   (a) $O(k \log n)$

   (b) $O(k^2)$

   (c) $O(k \log k)$

5. Дано множество, представленное в виде бинарной кучи с минимумом в корне. Найти $k$-ую статистику за $O(k \log k)$.

6. (*) Найти отрезок массива, на котором $(\min_{i \in [l,r]} a_i) \cdot \sum_{i \in [l,r]} a_i$ максимально. Время $O(n)$. Числа целые.

## 6.2 Домашнее задание

**Дедлайн: 22 октября, 23.59**

1. Придумайте структуру данных на основе кучи со следующими операциями: $\widehat{\text{I}}\text{nsert}(x)$, $\widehat{\text{D}}\text{elete}\{\text{Median,Min,Max}\}()$, $\widehat{\text{G}}\text{et}\{\text{Median,Min,Max}\}()$. Все операции должны выполняться за $O(\log n)$.

2. Пусть есть массив, состоящий из различных элементов, и алгоритм $A$, который находит $k$-ую порядковую статистику, используя только попарные сравнения. Покажите, что, используя результаты сравнений, проведенных алгоритмом, можно разделить элементы на меньшие $k$-ой статистики и большие её.

3. Дан набор из $n$ целых чисел. Найдите пару чисел, сумма которых наиболее близка к нулю. Время работы $O(n \log n)$.

4. Дан набор из $n$ пар гаек и болтов, все размеры которых различны. Гайки и болты перемешаны. Требуется для каждой гайки найти соответствующий болт. Сравнивать можно только болты с гайками (сравнить две гайки между собой, или два болта между собой — невозможно). Время работы:

   (a) $O(n \log n)$ в среднем,

   (b) (*) $O(n \log n)$.

5. (*) Назовем массив `A[0..n-1]` почти отсортированным (на 90%), если из него можно вычеркнуть 10% элементов, чтобы оставшиеся оказались в отсортированном порядке.

   Наша задача построить эффективный алгоритм для проверки на почти отсортированность. Для простоты, далее будем рассматривать только массивы, все элементы которых различны.

   Рассмотрим следующий алгоритм.

```
def binary_search(a, key, left, right):
    if left == right - 1:
        return left
    else:
        mid = left + (right - left) / 2
        if key < a[mid]:
            return binary_search(a, key, left, mid)
        else:
            return binary_search(a, key, mid, right)


def check-sort(a):
    for r in range(0, k):
        i = random.range(0, n)
        j = binary_search(a, a[i], 0, len( a ))
        if i != j:
            return False
    return True
```

   Докажите, что при выборе соответствующего $k$ данный алгоритм

   - выдает всегда `True` для отсортированного массива;
   - выдает `False` с вероятностью хотя бы $\frac{2}{3}$, если массив не отсортирован на 90%.

   Оцените значение $k$.

# 7 Практика 7. Графы

## 7.1 Практика

1. Дано корневое дерево $T = \langle V, E\rangle$. Каждая вершина может иметь произвольное число детей. Придумаайте структуру данных, которая позволяла бы отвечать на запросы вида: "Является ли вершина $u$ потомком вершины $v$?"за $O(1)$. Разрешается сделать препроцессинг за линейное время.

2. Дан граф $G = \langle V, E\rangle$. Проверить, является ли он двудольным. Время: $O(V + E)$.

3. Дан граф $G = \langle V, E\rangle$. Есть ребро $e$. Проверить, лежит ли ребро на каком-нибудь цикле. Время: $O(V + E)$.

4. Для заданного неориентированного графа $G = \langle V, E\rangle$ посчитать минимальное число ребер, которые нужно добавить в граф, чтобы он стал связным, за $O(V + E)$.

5. Корневое дерево $T$ на $V$ вершинах задается массивом из $V$ элементов. Все вершины пронумерованы. Для каждой вершины в массиве указан его родитель. Для корня $r$ значение в массиве равно -1. Требуется опредить как будет выглядеть новое представление дерева, если корень $r$ сменить на корень $q$. Разрешается использовать $O(1)$ дополнительной памяти. Менять массив можно. Время $O(V)$.

6. (*) Дан неориентированный граф $G$. Степень каждой вершины графа не превосходит трех. Требуется разбить граф на две доли так, чтобы у каждой вершине в ее же доле было не более одного соседа. Время $O(V + E)$.

# 8 Практика 8. Поиск в глубину

## 8.1 Практика

1. Дан ориентированный ациклический граф $G = \langle V, E \rangle$. Найти в нем гамильтонов путь за $O(V+E)$ или сказать, что такого нет.

2. Есть три бочки на 10, 7 и 4 литра. 7- и 4-литровые бочки заполнены водой. Разрешается переливать воду между бочками до момента либо пока одна бочка не заполнится полностью, либо пока из другой не выльется вся вода. Проверить, можно ли отмерить 2 литра. Если возможно, найти минимальное число переливаний. Как решить аналогичную задачу при условии, что емкости бочек $a$, $b$ и $c$ ($a \le b \le c$); бочки с емкостями $a$ и $b$ заполнены водой? Время: $O(a \cdot b)$.

3. В стране $n$ аэропортов. Самолет может сделать перелет из аэропорта $i$ в аэропорт $j$, израсходовав $w_{ij}$ горючего. При этом $w_{ij}$ может отличаться от $w_{ji}$, и $w_{ii} = 0$. Требуется найти минимальный размер бака, позволяющий добраться самолету из любого города в любой, возможно с дозаправками. Решить за $O(n^2 \log W)$, где $W$ — максимум по всем $w_{ij}$.

4. Дан неориентированный граф $G = \langle V, E \rangle$. Назовем ребро хорошим, если оно не лежит на одном цикле. Найдите все хорошие ребра за $O(V + E)$.

5. Постройте граф с отрицательными весами, на котором Деикстра выдавал бы неверный ответ. Граф не должен содержать отрицательных циклов. Где в доказательстве используется положительность весов ребер?

6. Рассмотрим взвешенный граф $G = \langle V, E \rangle$. Пусть некоторые ребра имеют отрицательный вес, но граф не имеет отрицательных циклов. Назначим каждой вершине $v$ некоторое число $\phi_v$ так, что $w'_{uv} = w_{uv} + \phi_v - \phi_v \ge 0$ для любого ребра $(u, v) \in E$. Будем искать кратчайшие пути алгоритмом Деикстра на графе с весами $w'$. Может ли алгоритм Деикстры дать неверный ответ при таком подходе? Ответ докажите.

7. (*) Пусть есть связный граф $G = \langle V, E \rangle$. Минимальный разрез графа – это минимальное по размеру множество ребер такое, что после его удаления граф теряет связность. Рассмотрим операцию стягивания ребра $(u, v)$. Это операция, после которой ребро $(u, v)$ пропадает, вершины $u$ и $v$ сливаются, а множества ребер, исходящих из них, объединяются, возможно образуя кратные ребра. Рассмотрим следующий алгоритм. Будем равномерно случайно брать ребро из графа и стягивать его. Будем так делать, пока в графе не останутся две вершины.

   (a) Докажите, что число ребер между оставшимися двумя вершинами будет не меньше размера минимального разреза.

   (b) Оцените снизу вероятность того, что это число будет равно размеру минимального разреза.

   (c) Постройте эффективный алгоритм поиска минимального разреза, который ошибается с константной вероятностью.

## 8.2 Домашнее задание

**Дедлайн: 30 октября, 23.59**

1. Есть клетчатое поле $n \times m$. Северо-западная клетка имеет координаты $(1, 1)$. Некоторые клетки закрашены черным цветом. Есть обычный игральный кубик: каждая сторона помечена числом от 1 до 6, сумма чисел на противоположных гранях равна 7. Размеры грани кубика и клетки совпадают. Кубик стоит в клетке $(1, 1)$. На верхней грани написано число 1, на южной – 2, на восточной 3. Кубик разрешается перекатывать в соседнюю по ребру клетку, если она не покрашена в черный. Можно ли привести кубик в клетку $(n, m)$, чтобы его пространственное положение было таким же, как изначально в $(1, 1)$. Задача: построить для данной задачи граф из $O(nm)$ ребер и $O(nm)$ вершин.

2. В долине потоп. Долина представляет собой клетчатое поле $n \times m$. В каждой клетке написана высота. Максимальная высота клетки – $h$, минимальная – 0. Изначально уровень воды 0, но каждую минуту он увеличивается на 1. Чтобы пройти между соседними по ребру клетками высоты $h_1$ и $h_2$, необходимо потратить $1 + |h_1 - h_2|$ минут. Турист находится в клетке $(1, 1)$, ему необходимо добраться до клетки $(n, m)$, не коснувшись воды. Получится ли у него сделать это? Как определить минимальное время, чтобы достигнуть цели, если это возможно? Постройте граф размера $O(nmh)$ для данной задачи.

3. Дан взвешанный ориентированный граф $G = \langle V, E \rangle$ с выделенной вершиной $s \in V$. Требуется построить дерево “кратчайших” путей из вершины $s$ во все остальные по следующим метрикам.

   (a) Вес пути определяется весом самого тяжелого ребра.

   (b) (*) Вес пути определяется парой весов самого тяжелого ребра и следующего за ним по весу. Пары сравнивается по элементно: сначала первые элемент; если они равны, то вторые.

   Время – $O((V + E) \log V)$.

4. (*, группа Опарина) Дан неориентированный граф $G = \langle V, E \rangle$. Назовем ребро хорошим, если удалив его, число компонент связности в графе увеличивается. Найдите все хорошие ребра за $O(V + E)$.

5. (*, группа Давыдова) Дан неориентированный граф $G = \langle V, E \rangle$. Назовем вершину хорошей, если удалив её, число компонент связности в графе увеличивается. Найдите все хорошие вершины за $O(V + E)$.