Алгоритмы и модели вычислений
Задачи

Содержание

Сортировки 2
Алгебра и теория чисел 3
Графы и алгоритмы 10
Алгебра и теория чисел 20

Сортировки

Задача A.1. Рассмотрим два утверждения:

(A) В любом корректном алгоритме поиска минимального элемента в массиве из n чисел, использующем только попарные сравнения, каждый реализуемый путь от корня к листу имеет не менее $(n-1)$ ребра.

(B) Для корректности алгоритма $\mathcal{A}$ нахождения минимума необходимо, чтобы граф $G_{\mathcal{A}}^{P(A)}$ был связен для любого массива A.

(a) Докажите импликацию $(B) \Rightarrow (A)$.

(b) Докажите утверждение (B).

Задача A.2. Переопределим выбор случайного барьерного элемента следующим образом: сопоставим каждому элементу ранг — целое число $x \in [1; n]$, а выбор барьерного элемента будем проводить, выбирая каждый раз из текущего массива элемент с минимальным рангом. Покажите, что этот способ эквивалентен классическому способу выбора барьерного элемента на каждом шаге.

Задача A.3. **(a)** Покажите, что любое разрешающее дерево поиска медианы позволяет также восстановить индексы всех элементов, больших медианы, и всех элементов, меньших медианы.

(b) Покажите, что любое разрешающее дерево для медианы содержит путь от корня к листу длины $\frac{3n}{2} - O(\log n)$.

Задача A.4 $(*)$**.** Дан массив чисел $a[1..n]$. Постройте алгоритм, использующий $O(\log n)$ дополнительных битов, совершающий не более двух односторонних проходов по массиву и находящий элемент, встречающийся более $\frac{n}{2}$ раз [если такой есть].

Задача A.5. В массиве $a[1..N]$ записано N целых чисел. Все втречаются по 2 раза, кроме одного, которое встречается 3 раза. Требуется найти число, встречающееся 3 раза. Время работы должно быть $O(N \log \max(i, a[i]))$, а память — $O(\log \max(i, a[i]))$.

Задача A.6. Нужно отсортировать массив n чисел, размер которого превышает размер оперативной памяти компьютера в k раз. Предложите как можно более быструю процедуру и оцените ее трудоемкость.

Задача A.7. Покажите, что задача сортировки массива из N чисел может быть сведена за линейное время к задаче построения выпуклой оболочки некоторого множества на плоскости.

Задача A.8. Даны n горизонтальных отрезков, лежащих на оси Ox и заданных массивом координат своих концов $[a_i, b_i]$. Отрезки можно сдвигать вправо и влево вдоль оси Ox. Необходимо с помощью сдвигов сделать так, чтобы пересечение всех отрезков стало непустым. Предложите $O(n \log n)$ алгоритм, который находит требуемый набор сдвигов минимальной суммарной длины. Считайте, что все арифметические операции с координатами выполняются за $O(1)$.

Алгебра и теория чисел

Задача B.1. Рассмотрим аналог алгоритма Карацубы, в котором число делится на три «равные» части: перемножая a и b, мы выбираем базу x так, чтобы $a = a_2x^2 + a_1x + a_0$, $b = b_2x^2 + b_1x + b_0$, остальные операции проделываются также, как в Карацубе. Постройте детально этот алгоритм и оцените его сложность.

Задача B.2. Последовательность $(X_n)_{n\in\mathbb{N}}$ задана первыми членами $X_0, \dots X_{k-1}$ и линейным реккурентным соотношением $X_n = a_1X_{n-1} + \dots + a_kX_{n-k}$. Постройте алгоритм, вычисляющий X_n за $O(k^2 \log n)$ арифметических операций.

Задача B.3. Пусть p — простое число, $\alpha \in \mathbb{Z}/p\mathbb{Z}$, а $n \geqslant p$. Покажите, что α^n можно вычислить некоторым алгоритмом, имеющим сложность $O(\log n \log p + \log^3 p)$.

Задача B.4.
- **(a)** Положим, что нам надо посчитать α^m для фиксированного $\alpha \in \mathbb{Z}/n\mathbb{Z}$ и различных m не больше длины ℓ. Покажите, что для любого положительного целого k мы можем сделать не более k возведений в квадрат и 2^ℓ дополнительных умножений в $\mathbb{Z}/n\mathbb{Z}$ так, что после этого можно посчитать α^m, используя не более $\ell/k + O(1)$ возведений в квадрат и $\ell/k + O(1)$ умножений.
- **(b)** Даны $\alpha \in \mathbb{Z}/n\mathbb{Z}$, $m_1, \dots, m_r$, где $m_i > 1$. Положим $m = \prod_i m_i$. Покажите, что

$$\left(\alpha^{\frac{m}{m_1}}, \dots \alpha^{\frac{m}{m_r}}\right)$$

можно вычислить алгоритмом, имеющим сложность $O(\log r \log m)$. Покажите, что если $r = O(\log\log m)$, то можно понизить сложность до $O(\log m)$.

Задача B.5. Пусть $A : \mathbb{Q}^n \to \mathbb{Q}^m$ — линейный оператор, заданный матрицей. Предъявите полиномиальные алгоритмы для нахождения $\mathrm{Ker}A$ и $\mathrm{Im}A$.

Задача B.6. Пусть $A \in \mathrm{Mat}_n(R)$ — матрица над некоторым кольцом главных идеалов (например, $\mathbb{Z}$). Постройте полиномиальный алгоритм, вычисляющий нормальную форма Смита для A.

Задача B.7. Предъявите полиномиальные алгоритмы, разрешающие следующие задачи
- **(a)** нахождение вычета, удовлетворяющего КТО, то есть нахождения единственного x из промежутка $[0; p_1p_2 \dots p_k)$, удовлетворяющего системе сравнений по различным простым модулям p_i, $i = 1, \dots, k$

$$x = a_1 \bmod p_1, \dots, x = a_k \bmod p_k$$

- **(b)** проверки совместности системы сравнений

$$x = a_1 \bmod p_1, \dots, x = a_k \bmod p_k$$

где модули m_i — произвольные натуральные числа;
- **(c)** решения системы линейных диофантовых уравнений $\sum_{j=1}^m a_{ij}x_j = b_i$, $a_{ij}, x_j, b_i \in \mathbb{Z}$, $i = 1, \dots, n$, $j = 1, \dots, m$

Задача B.8. Постройте алгоритм, имеющий сложность $O(mn(m+n)\log^2\ell)$ и вычисляющий по данной матрице $A \in \mathrm{Mat}_{m\times n}(\mathbb{Z}/\ell\mathbb{Z})$ матрицы $X \in \mathrm{Mat}_{m\times m}(\mathbb{Z}/\ell\mathbb{Z})$ и $B \in \mathrm{Mat}_{n\times n}(\mathbb{Z}/\ell\mathbb{Z})$ такие, что X обратима, а B — ступенчатая верхнетреугольная.

Задача B.9. Рассмотрим «вариант» быстрого БПФ-умножения: по входным многочленам f и g степеней $\deg g \leqslant \deg f = n$ вычисляются значения в корнях $n+1$ степени из единицы, затем соответствующие последовательности поточечно перемножаются и к полученной последовательности применяется обратное ДПФ. Покажите, что результирующий многочлен равен $fg \mod x^n - 1$.

Задача B.10. Даны два многочлена $f(x)$ и $g(x)$ степени, причем $\deg f < \deg g = n$.

- **(a)** Постройте алгоритм, который за $o(n^2)$ вычисляет $f^{-1} \mod x^n$
- **(b)** Пользуясь предыдущим пунктом, постройте алгоритм Евклида для многочленов, сложность которого есть $o(n^2\log n)$.

Задача B.11. Пусть A — подмножество целых чисел в отрезке $[1, m]$. Покажите, как за $O(m^2)$ найти

- **(a)** $A + A$, минковскую сумму множества с самим собой;
- **(b)** число арифметических прогрессий длины 3

Задача B.12. Пусть $\Sigma = \{1,\dots N\}$ — некоторое конечное множество; будем считать, что 0 кодирует «джокер», то есть метасимвол, на месте которого может быть любая буква. Рассмотрим задачу *поиска подслова*: для последовательностей $\overline{a} = a_0,\dots a_n$ и $\overline{b} = b_0,\dots b_m$, где $a_i, b_j \in \Sigma \cup \{0\}$, определить, является ли $\overline{b}$ подсловом $\overline{a}$. При этом 0 может «совпадать» с любой буквой. Например, 102 является подсловом 22112, а 222 — не является.

- **(a)** Постройте алгоритм для поиска подстрок в тексте, имеющий сложность $O(n\log n)$.
- **(b)** Покажите, как понизить трудоемкость построенного алгоритма до $O(n\log m)$.

Задача B.13. Определим последовательность *матриц Адамара* H следующим образом:

$$H_0 = \begin{bmatrix}1\end{bmatrix}, \quad H_n = \begin{bmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1}\end{bmatrix}$$

Покажите, что если v — столбец размера 2^k, то $H_k v$ может быть вычислено за $O(n\log n)$ арифметических операций.

Задача B.14. Пусть Σ — конечный алфавит; будем называть *кодом* длины n подмножество Σ^n [всех последовательностей длины n из элементов Σ]. Элементы кода будем называть *кодовыми словами.*

Мы рассматриваем Σ^n как пространство с метрикой Хэмминга. Будем считать, что $\Sigma = \mathbb{F}_q$, а кодовому слову $a_0,\dots,a_{n-1}$ соотвествует некоторый $(a_0,\dots,a_{n-1}) \in \mathbb{F}_q^n$. Назовем код $C = [n,k,d]_q$ *линейным*, если $C \subset \mathbb{F}_q^n$ — линейное подпространство, а расстояние между любыми двумя кодовыми словами не меньше d.

(a) Постройте алгоритм, проверяющий по данному слову $x \in \mathbb{F}_q^n$ и линейному коду C [считаем, что известен базис C] истинность $x \in C$.

(b) Будем говорить, что код C *циклический*, если

$$(a_0, \ldots, a_{n-1}) \in C \Rightarrow (a_{n-1}, a_0, \ldots, a_{n-2})$$

Покажите, что C соответствует главный идеал $(g(x))$ кольца $\mathbb{F}_q[x]/(x^n - 1)$ и наоборот. Многочлен $g(x)$ называется *порождающим* кода C.

(c) Пусть ξ — примитивный корень $\mathbb{F}_q[x]/(x^n - 1)$, а g — минимальный многочлен, зануляющийся во всех элементах множества $\{\xi, \ldots \xi^d\}$ для некоторого $d < \phi(n)$. Циклический код, порожденный g, будем называть *БЧХ-кодом*. Покажите, что расстояние между любыми кодовыми словами БЧХ не меньше d.

Задача B.15. Рассмотрим следующий алгоритм.

function Cyclotomic($n \in \mathbb{N}$, $p_1, \ldots p_r$ — простые делители n)
  $f_0 \leftarrow x - 1$
  for $i \in [1; r]$ **do**
    $f_i \leftarrow \frac{f_{i-1}}{(}x^{p_i})f_{i-1}(x)$
  end for
  $f_r(x^{\frac{n}{p_1 \cdots p_r}})$
end function

Покажите, что этот алгоритм корректно вычисляет циклотомический многочлен, и оцените его сложность.

Задача B.16 $(**)$**.** Рассмотрим генератор псевдослучайных чисел вида

$$X_n = \sum_{j=1}^{n} \alpha_j \phi_j(X_0, \ldots X_{n-1}) \mod m$$

Будем говорить, что у нас есть *эффективный способ предсказания последовательности*, если при известных и полиномиально вычислимых ϕ_j, известных $X_0, \ldots X_{n-1}$ и неизвестных коэффициентах α_j и m мы можем вычислить за полиномиальное по $\log m$ и k время X_n, совершив не более $poly(\log m, k)$ [не зависящее от n] ошибок [неверных предсказаний элемента последовательности]. [Считаем, что после совершения ошибки алгоритм получает правильное значение и соответствующим образом меняет значения α_i или m.]

Определим

$$V_i = (\phi_1(X_0, \ldots x_{i-1}), \ldots \phi_k(X_0, \ldots x_{i-1}))$$

(a) Пусть $L = (l_1, l_2, \ldots, l_r)$ — возрастающая последовательность натуральных чисел, а $l_s > l_r$. Положим, что существуют константы $c_1, c_2, \ldots, c_r \in \mathbb{Q}$ такие, что $\sum_{i=1}^{r} c_i V_{l_i} =$

V_{l_s}. Положим также, что $c_i = e_i/f_i$, где $e_i, f_i \in \mathbb{Z}$, а $d = {}^{\circ}(f_1, \dots f_r)$. Тогда

$$m | d\left(X - \sum_{i=1}^{r} c_i X_{l_i}\right).$$

Покажите, как можно определить m, совершив не более полиномиального числа ошибочных предсказаний X_n.

(b) (*) Будем говорить, что множество функций $\{\phi_j(X_0, \dots X_{n-1})\}$ обладает свойством *единственной экстраполяции длины* r, если для любого модуля m по данным $\{y_0, \dots y_{k-1}\}$ и истинности соотношения $y_n = \sum\limits_{j=1}^{n} \alpha_j \phi_j(y_0, \dots y_{n-1}) \mod m$ для $n \in [0; r-1]$ однозначно восстанавливается вся последовательность $(y_n)_{n\in\mathbb{N}}$.

Покажите, что существует эффективный способ предсказания последовательности, множество функций генератора которое удовлетворяет свойству эффективный способ предсказания последовательности [для некоторой длины]. Оцените максимально возможное число неверных предсказаний.

Задача B.17. Допустим, что нам не дано разложение $p-1$ на простые множители, но известен простой делитель q числа $p-1$. Постройте эффективный алгоритм для нахождения элемента $(\mathbb{Z}/p\mathbb{Z})^\times$ порядка q.

Задача B.18. Нам даны простое число p и разложение на простые $p - 1 = \prod_{i=1}^{r} q_i^{e_i}$.

(a) Допустим, что нам также известен $\alpha \in (\mathbb{Z}/p\mathbb{Z})^\times$. Постройте алгоритм, находящий порядок элемента α в $(BZ/p\mathbb{Z})^\times$ за $O(r \log^3 p)$.

(b) Постройте алгоритм, решающий ту же задачу за $O(\log r \log^3 p)$.

(c) Постройте рандомизированный алгоритм, находящий порождающий группы $(\mathbb{Z}/p\mathbb{Z})^\times$ за среднее время $O(\log r \log^3 p)$.

Задача B.19. Пусть p — простое число, $q|p-1$ — также простое, $\gamma \in (\mathbb{Z}/p\mathbb{Z})^\times$ — порождающий некоторой подгруппы G порядка q, $\alpha \in G$. Определим *представление* δ *относительно* γ *и* α для $\delta \in G$ как пару целых чисел (r, s), где $0 \leqslant r < q$ и $0 \leqslant s < q$ и $\gamma^r \alpha^s = \delta$. окажите, что существует эффективный алгоритм, который

(a) по представлению 1 относительно γ и α вида (r, s) для некоторого $s \neq 0$ вычисляет $\log_\gamma \alpha$.

(b) по данному δ и двум его различным представлениям относительно γ и α вычисляет $\log_\gamma \alpha$.

(c) при наличии «оракула», находящего для данного ему δ представление δ относительно γ и α, вычисляет $\log_\gamma \alpha$.

Задача B.20. Пусть существует эффективный алгоритм, находящий по данному n и $\alpha \in (\mathbb{Z}/n\mathbb{Z})^\times$ порядок α. Постройте по нему эффективный алгоритм факторизации n.

Задача B.21. Докажите, что на многочлене степени n алгоритм Distinct-Degree-Factorization совершает $O(n^3 \log p)$ арифметических операций в $\mathbb{F}_p$.

Задача B.22. Постройте полиномиальный алгоритм, который по данному на вход целому числу N и нормированному многочлену $f \in \mathbb{Z}[x]$ степени d определит,

(a) имеет ли f корень в $\mathbb{Z}/N\mathbb{Z}$;
(b) если f имеет корень в $\mathbb{Z}/N\mathbb{Z}$, найти этот корень.

Задача B.23 $(*)$**.** Пусть решетка в $\mathbb{R}^n$ задана базисом $\{b_1, \dots b_n\}$. Кратчайшим вектором решетки будем называть вектор решетки, имеющий минимальную евклидову норму.

Рассмотрим следующий алгоритм.

1: Построить $\{b_1^\star, \dots b_n^\star\}$ по $\{b_1, \dots b_n\}$ ортогонализаицей Грама-Шмидта
2: **for** $i \in [2; n]$ **do**
3:     **for** $j \in [i-1; 1]$ **do**
4:         $b_i = b_i - \alpha_{ij} b_j$, где $\alpha_{ij} = \lceil \frac{\langle b_i, b_j^\star \rangle}{||b_j^\star||^2} \rfloor$
5:     **end for**
6: **end for**
7: Если для некоторого i

$$\frac{3}{4}||b_i^\star||^2 > ||b_{i+1}^\star + \frac{\langle b_{i+1}, b_i^\star \rangle}{||b_i^\star||^2} b_i^\star||^2,$$

то поменять b_i и b_{i+1} местами и вернуться к шагу 1.
8: **return** $b_1, b_2, \cdots, b_n$.

Покажите, что этот алгоритм корректно вычисляет кратчайший вектор за полиномиальное время.

Задача B.24. Пусть $f(x)$ и $g(x)$ — два неприводимых многочлена над $\mathbb{F}_q$ степени d. Постройте эффективный алгоритм, явно строящий изоморфизм $\mathbb{F}_q[x]/f(x)$ и $F_q[X]/g(X)$.

Задача B.25. Пусть $\mathbb{F}$ — конечное поле характеристики p, а $f = f_1 \dots f_r$, где f_i — различные неприводимые многочлены в $\mathbb{F}[x]$ со старшим коэффициентом 1. Положим $r > 1$. Для некоторого промежуточного поля $\mathbb{F}'$ [то есть $\mathbb{Z}/p\mathbb{Z} \subset \mathbb{F}' \subset \mathbb{F}$] будем называть $S = \{\lambda_1, \dots \lambda_s\}$ [где $\lambda_u \in \mathbb{F}[x]$, $\deg(\lambda_u) < \deg(f)$] *разделяющим множеством* f *над* $\mathbb{F}'$, если

- для $i \in [1; r]$ и $u \in [1; s]$ существует $c_{ui} \in \mathbb{F}'$ такое, что $c_{ui} = \lambda_u \mod f_i$
- для каждой пары i, j, $1 \leqslant i < j \leqslant r$ существует $u \in [1; s]$ такое, что $c_{ui} \neq c_{uj}$

Покажите, что если S разделяющее множество для f над $\mathbb{Z}/p\mathbb{Z}$, то следующий алгоритм факторизует f, совершая $O(p|S|\deg(f)^2)$ операций.

$H \leftarrow \{f\}$
for $\lambda \in S$ **do**
    for $a \in \mathbb{Z}/p\mathbb{Z}$ **do**
        $H' \leftarrow \varnothing$
        for $h \in H$ **do**

```
            d ← gcd(λ − a, h)
            if d ∈ {1, h} then
                H′ ← H′ ∪ {h}
            else
                H′ ← H′ ∪ {d, h/d}
            end if
        end for
    end for
end for
return H
```

Задача B.26. Пусть p — простое, c — положительное целое, а f — многочлен. Положим, что у нас есть решение x, которое удовлетворяет

$$f(x) = 0 \bmod p^c \qquad , \qquad f'(x) \neq 0 \bmod p$$

Покажите, что x можно поднять до решения $x^\times$ такого, что

$$f(x^\times) = 0 \bmod p^{2c} \qquad , \qquad x^\times = x \bmod p^c$$

Задача B.27. Пусть кольцо R факториально, а $\mathfrak{a} \subset R$ — идеал. Положим, что у нас есть разложение $f = gh \bmod \mathfrak{a}$ такое, что $s, t \in R$ «взаимнопросты» по модулю $\mathfrak{a}$, то есть $sg + th = 1 \bmod \mathfrak{a}$. Тогда мы можем построить $g^\times, h^\times, s^\times, t^\times$ такие, что

$$\begin{aligned} g^\times &= g \bmod \mathfrak{a} \\ h^\times &= h \bmod \mathfrak{a} \\ f &= g^\times h^\times \bmod \mathfrak{a}^2 \\ s^\times g^\times + t^\times h^\times &= 1 \bmod \mathfrak{a}^2 \end{aligned}$$

Более того, для любых g', h', удовлетворяющих свойствам выше, существует единсвтвенный $u \in \mathfrak{a}$ такой, что

$$\begin{aligned} g' &= g^\times(1 + u) \bmod \mathfrak{a}^2 \\ h' &= h^\times(1 - u) \bmod \mathfrak{a}^2 \end{aligned}$$

Нормальное замыкание подгруппы H в G — наименьшая нормальная подгруппа в G, содержащая H. Обозначим нормальное замыкание H в G как $\langle H^G \rangle$:

$$H \leqslant \langle H^G \rangle \lhd G$$

Нормализатор H, обозначаемый $N^G(H)$, является самой большой подгруппой G, в котором H нормально.

$$H \lhd N^G(H) \leqslant G$$

Подгруппа H в G называется *субнормальной* в G, если существует башня подгрупп, скованная нормальностью (т. е.)

$$H = G_r \lhd G_{r-1} \lhd \cdots \lhd G_0 = G$$

Это обозначается как $H \lhd \lhd G$.

Задача B.28. Покажите, что $H \lhd \lhd G$ титтк $H \lhd \lhd \langle H^G \rangle$. Выведите отсюда, что алгоритм

$K_1 = G; K_2 = H$
while $\langle K_2^{K_1} \rangle \neq K_2$ **do**
    if $\langle K_2^{K_1} \rangle = K_1$ **then**
        return false
    else
        $K_1 = \langle K_2^{K_1} \rangle$
    end if
end while
output true

разрешает полиномиально задачу SUBNORMAL: для заданных групп $H = \langle B \rangle$и $G = \langle A \rangle$ проверить, если $H \lhd \lhd G$.

Задача B.29. Пусть $H = \langle B \rangle$ и $G = \langle A \rangle$ — подгруппы Sym_n. Покажите, что если $H \lhd \lhd \langle G, H \rangle$, то существует полиномиальный алгоритм, вычисляющий малое множество порождающих для $G \cap H$. Оцените порядок полученного множества порождающих.

Задача B.30. Рассмотрим задачу Разрешимость: для данной группы $G = \langle A \rangle$ установить ее разрешимость. Постройте полиномиальный алгоритм, разрешающий Разрешимость.

Задача B.31. Постройте полиномиальные алгоритмы, разрешающие следующие задачи:

- **(a)** Подгруппа: даны подгруппы $H = \langle B \rangle$ и $G = \langle A \rangle$ группы Sym_n. Проверить, верно лт $H \leqslant G$.
- **(b)** Нормальность: даны подгруппы $H = \langle B \rangle$ и $G = \langle A \rangle$ группы Sym_n. Проверьте, является ли H нормальной подгруппой в G.
- **(c)** Нормальное замыкание: даны подгруппы $H = \langle B \rangle$ и $G = \langle A \rangle$ группы Sym_n, найти $\langle H^G \rangle$.

Задача B.32 ($*$). **(a)** Покажите, что если $(i, j) \in E$, то условие корректности раскраски имеет вид

$$x_i^{k-1} + x_i^{k-2} x_j + \ldots + x_j^{k-1} = 0 \tag{1}$$

(b) Теперь рассмотрим идеал

$$I_{G,k} = I_k + \langle x_i^{k-1} + x_i^{k-2} x_j + \ldots + x_j^{k-1} = 0,\ (i, j) \in E \rangle \tag{2}$$

Покажите, что $I_{G,k}$ радикален и мощность множества нулей есть число корректных раскрасок.

Графы и алгоритмы

Задача C.1. (1) Покажите, что при поиске в глубину в неориентированном графе любое ребро является либо ребром дерева, либо обратным ребром.

(2) Постройте алгоритм, который для заданного *неориентированного* графа G определяет, есть в нем простой цикл, за $O\big(|V|\big)$ операций.

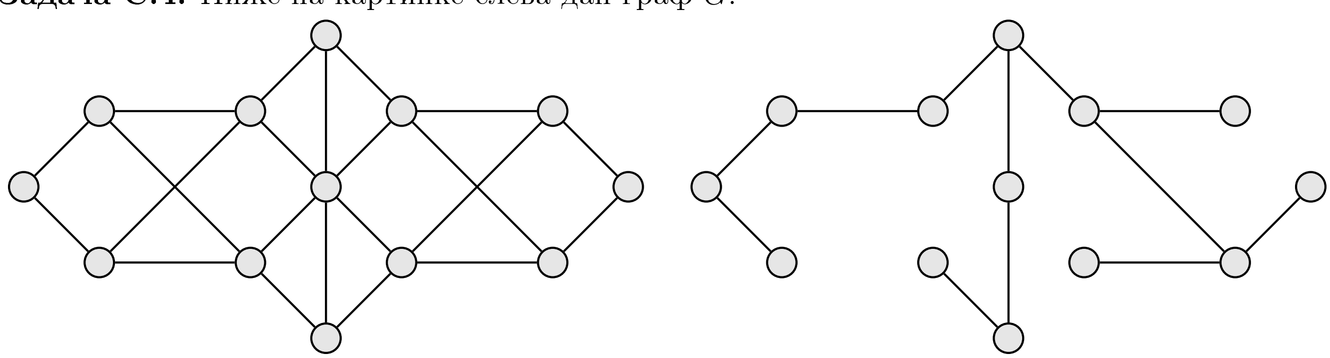
Задача C.2. Докажите или опровергните, что следующее условие дает критерий, когда остовное дерево $F \subseteq G$ является результатом работы некоторого поиска в ширину связного неориентированного графа G:

> Остовное дерево $T \subseteq G$ является деревом некоторого поиска в ширину связного неориентированного графа G титтк в нем можно выбрать одну из вершин s за корень так, чтобы T было деревом кратчайших путей из s в графе G. Иными словами, путь по дереву из s в произвольную вершину t содержит не больше ребер, чем кратчайший путь между s и t в G.

Если в настоящем виде критерий неверен, то модифицируйте его до корректного.

Задача C.3. Постройте линейный по входу алгоритм, который, имея на входе граф G и некоторое его остовное дерево T, определяет, является ли T некоторым BFS-деревом G.

Задача C.4. Ниже на картинке слева дан граф G.



Какие из следующих утверждений верны?

(A) Хотя бы три ребра G — мосты.
(B) В любом DFS-дереве графа G есть хотя бы одна вершина степени не менее 2.
(C) В G любые две вершины могут быть соединены путем, состоящим из не более 7 ребер.
(D) Подграф $G' \subset G$, данный на правой картинке, является некоторым BFS-деревом.

Задача C.5. Постройте алгоритм, имеющий сложность $O(|V||E|)$ и находящий в графе $G = (V, E)$

(a) хотя бы один цикл;
(b) кратчайший цикл.

Задача С.6. Дан [неориентированный] граф $G = (V, E)$. Постройте алгоритм, имеющий сложность $O(V + E)$ и проверяющий двудольность графа.

Задача С.7. Постройте алгоритм, имеющий сложность $O(|V||E|)$ и находящий множество кратчайших путей p_{ij} в графе $G = (V, E)$ для каждой пары вершин (i, j).

Задача С.8. Дан граф $G = (V, E)$, в котором отмечены две вершины v_1 и v_2. Постройте эффективный алгоритм поиска кратчайтчайших путей между всеми парами вершин, проходящих через хотя бы одну из отмеченных вершин.

Задача С.9. Постройте алгоритм, который по входному графу $G = (V, E)$ определяет, содержится ли в нем простой цикл длины 4. Сложность алгоритма должна быть $O(|V|^3)$.

Задача С.10. Дан орграф (V, E). Предложите эффективный алгоритм, вычисляющий минимально необходимое количество ребер, которые нужно добавить, чтобы сделать граф сильно связным. Оцените асимпотически сложность вашего алгоритма.

Задача С.11. Дан орграф $G = (V, E)$, в котором каждая вершина $u \in V$ имеет некоторую целочисленную стоимость p_u. Определим $c : V \to \mathbb{Z}$:

$$c(u) = \min\{p_v \mid v \text{ достижима из } u\}$$

(a) Постройте алгоритм, имеющий сложность $O(|V| + |E|)$ и вычисляющий функцию c на всем графе, если граф G ацикличен.

(b) Обобщите построенный алгоритм на все неориентированные графы. Оцените сложность полученного алгоритма.

Задача С.12. *Конечный автомат* (или *недетерминированный конечный автомат*) — пятерка $A = \langle Q, \Sigma, S, F, \delta\rangle$, где

- Q — конечное *множество состояний*,
- Σ — алфавит,
- $S \subseteq Q$ — *множество стартовых состояний*,
- $F \subseteq Q$ — *множество финальных состояний*,
- $\delta \subset Q \times \Sigma \times Q$

Элементы множества δ мы будем называть *переходами* конечного автомата. Будем говорить, что конечный автомат *детерминированный*, если из каждого состояния по каждой букве не более одного перехода.

Слово w *принимается* конечным автоматом A, если существует последовательность состояний $q_0, \dots q_{|w|} \in Q$ такая, что $q_0 \in S$, $q_{|w|} \in F$ и $(q_i, w[i], q_{i+1}) \in \delta$, или же если $w = \epsilon$ и $S \cap F \neq \varnothing$. *Язык автомата* $L(A)$ — множество всех слов, принимаемых автоматом; будем говорить, что конечный автомат A *принимает* язык L, если $L(A) = L$.

(a) Постройте эффективный алгоритм, проверяющий, пуст ли язык данного автомата.

(b) Будем говорить, что состояние q *достижимо из состояния* p, если существует слово w такое, что $p \xrightarrow{w} q$. Постройте эффективный алгоритм, находящий все состояния, достижимые из стартового.

(c) Определим *полный конечный автомат* как автомат $\langle Q, \Sigma, S, F, \delta \rangle$, в котором для любого $p \in Q$ и $a \in \Sigma$ множество $\{q \in Q \mid (p, a, q) \in \delta\}$ непусто. Постройте эффективный алгоритм, находящий хотя бы один полный конечный автомат, эквивалентный [принимающий тот же язык] входному.

Задача C.13. Множество *регулярных выражений* над алфавитом Σ — множество строчек над алфавитом $\Sigma \cup \{\epsilon, +, \cdot, *, (,)\}$, удовлетворяющее следующим правилам:

- $0, \epsilon$, x для любой $x \in \Sigma$ — *атомарные* регулярные выражения;
- если α и β — регулярные выражения, то $\alpha + \beta, \alpha\beta, \alpha^*, (\alpha)$ — регулярные выражения;
- никаких других регулярных выражений нет, то есть любое регулярное выражение может быть получено из атомарных с помощью операций $+, \cdot, *$ и использования скобок.

Язык $L(\alpha)$ *регулярного выражения* α определяется индукцией по построению:

- $L(0) = \varnothing$, $L(\epsilon) = \{\epsilon\}$, $L(x) = \{x\}$ для любой $x \in \Sigma$;
- $L(\alpha + \beta) = L(\alpha) \cup L(\beta)$;
- $L(\alpha\beta) = L(\alpha)L(\beta)$;
- $L(\alpha^*) = (L(\alpha))^*$.

Например, язык $L \subset \{a, b\}^*$ всех слов, содержащих три буквы b, может быть задан регулярным выражением $a^*ba^*ba^*ba^*$: слово w, содержащее ровно три буквы b, должно иметь вид $a^xba^yba^zba^t$ для некоторых $x, y, z, t \in \mathbb{Z}_{\geqslant 0}$.

(a) Постройте алгоритм, вычисляющий язык входного конечного автомата A. На выходе алгоритма должно быть регулярное выражение. Оцените сложность построенного алгоритма.

(b) Покажите, что любой детерминированный конечный автомат, принимающий язык, заданный выражением

$$(a+b)^*b\underbrace{(a+b)\dots(a+b)}_{n \text{ раз}},$$

имеет $\Omega(2^n)$ состояний.

(c) Проверьте, что тем не менее существует полиномиальный алгоритм, строящий конечный автомат [не обязательно детерминированный] по входному регулярному выражению.

Задача C.14. Пусть $A = \langle Q, \Sigma, S, F, \delta \rangle$ — детерминированный конечный автомат. Введем на его состояниях отношение $\approx \in Q \times Q$: $q_1 \approx q_2$ титтк из q_1 и q_2 по любому слову можно попасть либо одновременно в некоторые финальные, либо одновременно в некоторые нефинальные состояния.

(a) Постройте эффективный алгоритм, находящий по входному A классы эквивалентности отношения $\approx$.

(b) Покажите, что все состояния в каждом классе эквивалентности можно «склеить» вместе с переходами, и полученный автомат $\hat{A}$ будет принимать тот же язык, что и A.

(c) Пусть A_1 и A_2 — два различных детерминированных конечных автомата, принимающие один и тот же язык. Верно ли, что $\hat{A}_1$ и $\hat{A}_2$ будут совпадать?

Задача C.15 (**∗∗**)**.** В контексте этой задачи *пальмой* графа (V, E) будем называть орграф $(V, E_1 \sqcup E_2)$ с выделенной вершиной s, где (V, E_1) — DFS-дерево с корнем s, E_2 — множество $E \setminus E_1$ с ориентацией «к корню», то есть ребра имеют вид $(v_i v_j)$, где v_i была посещена DFS после v_j.

Рассмотрим скетч процедуры

```
function Planarity(G = (V, E))
    if |E| > 3|V| − 3 then
        return nonplanar
    end if{G_1, ... G_k, X} = Block-Cut(G)
    for G_i ∈ {G_1, ... G_k} do
        P ← пальма графа G_i
        c — цикл в P
        вложить c в плоскость
        for e ∈ G_i \ c do
            проверить, является ли c ∪ {e} планарным
            if c ∪ {e} планарен и e можно добавить в планарное представление then
                добавить e
            else
                return nonplanar
            end if
        end for
    end for
    return planar
end function
```

Укажите, как достроить данный алгоритм, чтобы он корректно определял планарность входного графа G за $O(|V| + |E|)$.

Задача C.16. Дан орграф G. Используя функцию $L(v)$, которую мы ввели, находя блокирующие вершины в графе, постройте алгоритм, который обходит G в глубину ровно один раз и находит компоненты сильной связности.

Задача C.17. Дано дерево T. Постройте эффективный алгоритм, находящий минимальное вершинное покрытие T.

Задача C.18. Определим *грид-граф* как взвешенный орграф (V, E), в котором $V(G) = \{(i, j),\ i = 0, \ldots, m,\ j = 0, \ldots, n\}$, а дуги соединяют соседние точки: $E(G) = \{[(i, j) \to (i + 1, j)],\ i = 0, \ldots, m - 1,\ j = 0, \ldots, n$ или $[(i, j) \to (i, j + 1)],\ i = 0, \ldots, m,\ j = 0, \ldots, n - 1\}$, причем дугам G приписаны целочисленные веса. Вес пути определим как сумму весов входящих в него ребер.

Рассмотрим задачу поиска экстремального (самого «тяжелого» или самого «легкого») пути между вершинами $(0,0)$ и (m,n). Постройте $O(mn)$-алгоритм поиска экстремального пути.

Задача C.19. Пусть $N=(G,w)$ — взвешенный граф, где $G=(V,E)$ — ацикличный орграф с корнем s. Покажите, что существует алгоритм, имеющий сложность $O(E)$ и находящий множество длиннейших путей из s в остальные вершины [то есть для каждой $v\in V$ алгоритм находит самый длинный путь из s в v].

Задача C.20. Пусть известно, что веса ребер входного графа $G=(V,E)$ целочисленны, положительны и не превышают W. Покажите, как можно реализовать алгоритм Дейкстры так, чтобы сложность была

(a) $O(|W||V|+|E|)$
(b) $O((|V|+|E|)\log|W|)$

Задача C.21. Дан граф $G=(V,E)$, веса l_e всех ребер положительны, отмечена также вершина s. Постройте алгоритм, имеющий сложность $O((|V|+|E|)\log|V|)$ и вычисляющий $u:V\to\{0,1\}$ такую, что $u(v)=1$ титтк существует ровно один кратчайший путь из s в v.

Задача C.22. Пусть (G,c) — взвешенный граф, содержащий ребро отрицательного веса. Рассмотрим следующий алгоритм $\textsc{Shortpath}(G,s,c)$:

$C=\max(|c|)$
$\textsc{Dijkstra}(G,s,c+2C)$

Корректен ли такой алгоритм?

Задача C.23. Покажите, как модифицировать алгоритм Беллмана-Форда, чтобы он находил, содержится ли во входном (G,w) цикл отрицательного веса.

Задача C.24. Пусть дан ацикличный граф G. Рассмотрим операцию *транзитивного редуцирования*: если существует ребро $(uv)\in G$ и путь из u в v длины хотя бы 2, мы выбрасываем (uv). Граф $\hat{G}$, который получится после всевозможных транзитивных редуцирований, назовем *транзитивной редукцией* G. Предъявите алгоритм, имеющий сложность $O(|V|^3)$ и находящий транзитивную редукцию G.

Задача C.25. Пусть (G,w) — взвешенный граф, а v — некоторая вершина. Покажите, что любое минимальное остовное дерево содержит минимальное по весу инцидентное v ребро.

Задача C.26. Пусть (G,w) — взвешенный граф, а $\mathcal{T}$ — множество всех остовных деревьев (G,w). Положим

$$w(T):=\max\{w(e):e\in TT\in\mathcal{T}\}$$

Докажите, что все минимальные остовные деревья минимизируют функцию $w(T)$.

Задача C.27. Покажите на примере, что ориентированное остовное дерево максимального веса может не содержать путей [от корня до вершины] максимального веса.

Задача C.28. Пусть T — остовное дерево неориентированного взвешенного графа (G, w), в которым для любых вершин u и v соединяющий их в T путь [единственный] имеет максимальный вес среди всех путей из u в v в G. Верно ли, что T — максимальное остовное дерево?

Задача C.29. Мы показали, что целочисленная сеть всегда имеет максимальный целочисленный поток. Верно ли, что любой максимальный поток должен быть целым?

Задача C.30. Пусть f — поток в сети N. Будем называть *носителем потока* множество $\operatorname{supp}(f) = \{e \in E \mid f(e) \neq 0\}$. Поток f называется *элементарным*, если его носитель — путь [дерево, где степень каждой вершины не более 2]. Можно ли представить каждый максимальный поток как сумму элементарных потоков?

Задача C.31. Пусть $N = (G, c, s, t)$ — сеть, а (S_1, T_1) и (S_2, T_2) — минимальные разрезы N. Покажите, что $(S_1 \cup S_2, T_1 \cap T_2)$ и $(S_1 \cap S_2, T_1 \cup T_2)$ также являются минимальными разрезами T.

Задача C.32. Определим граф G_n на множестве натуральных чисел чисел $\{1, 2 \ldots, n\}$. в котором вершины — числа, и две вершины соединены тогда и только тогда, когда одно их них делится нацело на другое. Верно ли, что язык $L = \{(G_n, k)\}$, состоящий из кодировок графов G_n, имеющих независимое множество вершин размера k, является **NP**–полным? Считайте, что арифметические операции выполняются за $O(1)$.

Задача C.33 (∗). На клетчатой бумаге закрасили несколько клеток (координаты закрашенных клеток передаются на вход). Существует ли полиномиальный алгоритм, проверяющий, можно ли покрыть закрашенную область доминошками 1×2 так, чтобы каждая клетка покрывалась ровно два раза?

Задача C.34. Покажите, что в любом графе существует разрез, содержащий не менее половины всех ребер.

Задача C.35. Рассмотрим следующую задачу. В потоковой сети нет ограничений пропускной способности на дугах, но есть ограничения пропускной способности вершин. Формально, для каждой вершины v, отличной от истока и стока, задано целое неотрицательное число $c(v)$, и для потока в сети должно выполняться $\sum\limits_u f(u, v) = \sum\limits_u f(v, u) \leq c(v)$. Опишите алгоритм нахождения максимального потока в такой сети.

Задача C.36. Рассмотрим следующую задачу Сеть. Дан ориентированный граф $G = (V, E)$, дугам которого приписаны неотрицательные числа $l_i \leqslant u_i$, $i \in E$. Нужно проверить, можно ли приписать ребрам числа $F = \{f_i,\ i \in E\}$, чтобы в любой вершине v выполнялось равенство Кирхгоффа:

$$\operatorname{div} F = \sum_{(xv) \in E} f_i - \sum_{(vy) \in E} f_j = 0,$$

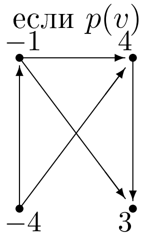
а также выполнялись неравенства $l_i \leqslant f_i \leqslant u_i, i \in E$. Покажите, как можно решить задачу Сеть с помощью решения подходящей задачи о максимальном потоке в сети и наоборот.

Задача C.37. Последовательность выполнения проектов задана ациклическим орграфом $G = (V, E)$ (если в орграфе есть ребро (u, v), то проект v не может начаться, пока не будет выполнен проект u). Выполнение проекта v приносит прибыль $p(v)$ (она может быть и отрицательна). Требуется выбрать подмножество проектов, приносящих максимальную суммарную прибыль, то есть найти такое подмножество проектов $M \subseteq V$, что

$$M = \operatorname{argmax}_{S \subseteq V}\{p(S) \stackrel{def}{=} \sum_{v \in S} p(v)\}.$$

Оказывается, что эту задачу можно свести к задаче о минимальном разрезе. Дополняем граф G источником s и стоком t, и задаем **бесконечные** пропускные способности на ребрах G. Далее, для всех вершин $v \in V$

- если $p(v) < 0$, то задаем ребро (s, v) с пропускной способностью $-p(v)$;
- если $p(v) > 0$, то задаем ребро (v, t) с пропускной способностью $p(v)$.



(a) С помощью алгоритма Форда-Фалкерсона найдите максимальный поток в полученной сети. Начальный поток нулевой.

(b) Затем, используя алгоритм Форда-Фалкерсона, найдите минимальный разрез.

(c) Обоснуйте конструкцию в общем случае.

Задача C.38. Рассмотрим один из вариантов алгоритма Каргера:

function FastKarger(G)
  if $|V| \leq 6$ **then**
    найти минимальный разрез C перебором;
    return C;
  else
    $t := \lceil |V|/\sqrt{2} + 1 \rceil$;
    независимо стянуть две копии G в графы H_1 и H_2 с t вершинами в каждом;
    $C_1 :=$ FastKarger(H_1);
    $C_2 :=$ FastKarger(H_2);
    if $w(C_1) < w(C_2)$ **then**
      return C_1;
    else
      return C_2;
    end if

end if
end function

(a) Оцените сложность алгоритма.

(b) Покажите, что данный алгоритм находит минимальный разрез с вероятностью $\Omega\left(\frac{1}{\log n}\right)$. В частности, это означает, что для нахождения минимального разреза с вероятностью $1-\varepsilon$ требуется независимо запустить алгоритм $O\left(\log n \cdot \log \frac{1}{\varepsilon}\right)$ раз.

Задача C.39 $(*)$**.** На вход задачи подаётся ориентированный граф $G = \langle V, E\rangle$ без контуров (ориентированных циклов). Необходимо покрыть его наименьшим числом простых путей, то есть найти наименьшее количество не пересекающихся по вершинам простых путей, чтобы каждая вершина принадлежала одному из них. Допускаются пути нулевой длины (состоящие из одной вершины). Предложите полиномиальный алгоритм.

Задача C.40 $(*)$**.** Пусть дан связный неориентированный граф $G = (V, E)$. Рассмотрим следующий алгоритм для определения двусвязности графа.

(1) Проводим DFS на графе G из произвольной вершины. Ориентируем ребра дерева к корню, а обратные — от корня. Каждое обратное ребро e образует единственный контур $c(e)$, состоящий из этого ребра и некоторых ребер дерева.

(2) Все вершины G помечаем как непросмотренные.

(3) Обходим вершины в порядке возрастания временных меток $u.d$. Пусть выбрана вершина v, $e_1, \ldots, e_k$ — выходящие из нее обратные ребра. Помечаем v как просмотренную, затем для каждого ребра e_i обходим контур $c(e_i)$, начиная с v, помечая вершины, пока не встретим просмотренную вершину. Полученный обход назовем **цепью**.

(4) Получаем список цепей в порядке их образования: $C = \{C_1, \ldots, C_p\}$. Этот список назовем **цепным разложением**.

(5)
- Если цепное разложение содержит не все ребра, то граф не 2-реберно-связный.
- Иначе, если в цепном разложении есть цикл, отличный от C_1, то граф 2-реберно-связный, но не двусвязный.
- Иначе, граф двусвязный.

Докажите корректность приведенного алгоритма.

Задача C.41. Для любых $k, d, m \in \mathbb{N}$ таких, что $0 < k \leqslant m \leqslant d$, постройте хотя бы один граф G, для которого $\kappa(G) = k$, $\lambda(G) = m$, $\delta(G) = d$.

Задача C.42. Пусть G — k-связный граф, где $k \geqslant 2$. Докажите, что если G не содержит независимого множества мощности $k+1$, то G гамильтонов.

Задача C.43. Постройте полиномиальный алгоритм, находящий в графе $G = (V, E)$ минор $K_{3,3}$ или минор K_5. Оцените его сложность.

Задача C.44. Определим *матрицу Татта* неориентированного графа $G = (\{1, \dots n\}, E)$ как матрицу $T \in \mathrm{Mat}_{n\times n}$, где

$$T_{ij} = \begin{cases} x_{ij}, & (ij) \in E, i < j \\ -x_{ji}, & (ij) \in E, i > j \\ 0, & (ij) \notin E \end{cases}$$

Здесь x_{ij} — переменные. Покажите, что $\det T \neq 0$ титтк существует совершенное паросочетание в G.

Задача C.45. Пусть дан планарный неориентированный граф $G = (\{1, \dots n\}, E)$ в виде списков смежности $A_1, \dots A_n$. Рассмотрим следующий алгоритм:

```
function 6-Color(G)
    for i ∈ [1; n − 1] do
        V_i ← {v ∈ V | deg(v) = i} (двусвязный список)
    end for
    for i ∈ {n, ... 1} do
        j ← min{m | V_m ≠ ∅}
        v_i ← первая вершина V_j
        V_i ← V_i \ {v_i}
        for u ∈ A_{v_i} do
            взять k так, что u ∈ V_k
            V_k ← V_k \ {u}, V_{k−1} ← V_{k−1} ∪ {u}
        end for
    end for
    for i ∈ [1; n] do
        c(v_i) ← min{x ∈ {1, 2, 3, 4, 5, 6} | ∄u ∈ A_{v_i}, c(u) = x}
    end for
    return c
end function
```

Покажите, что 6-Color корректно находит раскраску G в 6 цветов за $O(n)$ операций.

Задача C.46. Будем называть неориентированный граф $G = (V, E)$ *внешнепланарным*, если существует такое вложение G в плоскость, что все вершины графа оказываются смежными бесконечной грани.

(a) Покажите, что граф внешнепланарен титтк он не содержит миноров $K_{3,2}$ или K_4.

(b) Существует ли полиномиальный алгоритм раскраски внешнепланарного графа в 3 цвета?

Задача C.47 (∗∗)**.** Пусть G — неориентированный граф. *Древесное разложение* графа G — пара (T, ϕ), где T — дерево, а $\phi : V(T) \to 2^{V(G)}$ — отображение, удовлетворяющее следующим свойствам:

- для любого $e \in E(G)$ существует $t \in V(T)$ такое, что $e \subset \phi(t)$;
- для любого $v \in V(G)$ множество $\{t \in V(T) \mid v \in \phi(t)\}$ связно в T.

Шириной древесного разложения (T, ϕ) назовем $\max_{t \in V(T)} |\phi(t)| - 1$. *Древесной шириной* графа G будем называть минимально возможную ширину древесного разложения G.

(a) Докажите, что связный граф имеет древесную ширину 1 титтк он является деревом.

(b) Докажите, что следующие утверждения эквивалентны:
- древесная ширина G не более 2;
- G не содержит минор K_4;
- G может быть получен из пустого графа последовательным добавлением мостов и раздвоением и разделением ребер. (Раздвоение ребра $e = (vw) \in E(G)$ — добавление второго ребра с концами в v и w; разделение ребра $e = (vw) \in E(G)$ — добавление вершины x замена e на ребра (vx), (xw).)

Задача C.48. Расмотрим жадный алгоритм для задачи нахождения наибольшего независимого множества в неориентированном графе G. На каждом шаге обрабатывается пара $(I_i; G_i)$; $i \in \{1, 2, \ldots\}$ и $I_1 = \varnothing$; $G_1 = G$. На i-м шаге в в графе G_i выбирается произвольная вершина v_i минимальной степени и добавляется в $I_{i+1} = Ii \cup \{v_i\}$, а G_{i+1} образуется из G_i путем удаления v_i и всех смежных с ней вершин. Постройте граф, имеющий $O(n)$ вершин и независимое множество размера $O(n)$, в котором жадный алгоритм находит независимое множество размера $O(1)$.

Прочее

Задача D.1. Зафиксируем унарный алфавит. Пусть T_n — множество машин Тьюринга с n рабочими состояниями (то есть отличными от финальных). Для каждой машины $M \in T_n$, останавливающейся на пустом входе, определим $b(M)$ как число единиц, которые M оставит на ленте, закончив работу. Тогда

$$BB(n) = \max_{M \in T_n} b(M)$$

(1) Пусть существует машина Тьюринга M_F, имеющая N состояний и вычисляющая функцию $F(\cdot)$. Покажите, что существует машина Тьюринга, имеющая $1+n+2N$ состояний и оставляющая на ленте не менее $F(F(n))$ единиц.

(2) Выведите отсюда, что для любой вычислимой $f : \mathbb{N} \to \mathbb{N}$ существует такое $N \in \mathbb{N}$, что $BB(n) > f(n)$ для всех $n > N$. В частности, $BB(n)$ — невычислимая функция.

Задача D.2. Определим для $M \in T_n$ функцию $s(M)$ — число ячеек, которые обработала МТ до своей остановки, а $t(M)$ — число шагов, которые проделала МТ до своей остановки. Определим

$$S(n) = \max_{M \in T_n} s(M), \quad T(n) = \max_{M \in T_n} t(M)$$

Покажите, что

$$S(n) \leqslant (n+1)BB(5n)2^{BB(5n)}$$

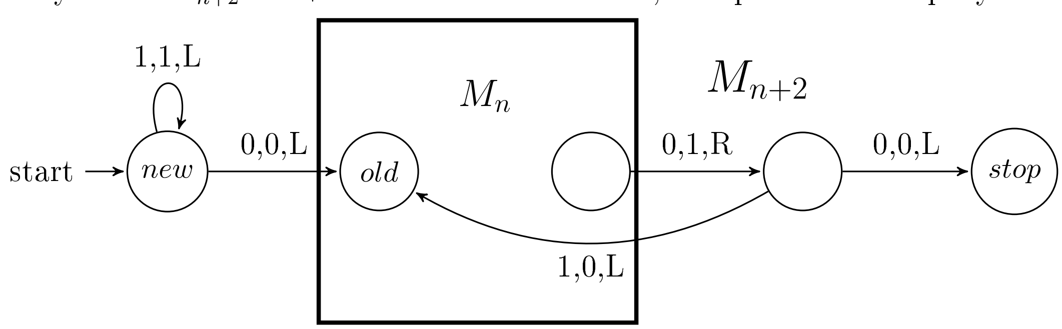
Задача D.3. Покажите, что

(a) $S(n) \leqslant BB(3n-1)$;

(b) $T(n) \leqslant (2n-1)BB(3n+3)$;

(c) $T(n) \leqslant nS(n)2^{S(n)}$.

Задача D.4. Индуктивно построим теперь по МТ M_n с n состояниями (не считая состояние останова) новую МТ M_{n+2} с $n+2$ состоянием по схеме, изображенной на рисунке.



Если к M_2 $(k-1)$ раз применить индуктивную конструкцию, то получим МТ M_{2k}. Из оценки выхода M_{2k}, т. е. длины блока единиц, который останется на ленте после того, как M_{2k} остановится, можно получить нижнюю оценку роста функции трудолюбия Радо (busy beaver).

Обозначим $A(k, m)$ число единиц, которые останутся на ленте, когда МТ M_{2k} остановится, стартуя с ленты, на которой записан блок из m единиц. Получите рекуррентую формулу для $A(k, m)$.

Задача D.5. Постройте алгоритм типа разделяй-и-властвуй для определения треугольника минимального периметра, вершины которого можно выбирать из заданных n точек плоскости. Множество точек представляет собой массив пар целых чисел. Считаем, что арифметические операции выполняются за единицу времени.

Задача D.6. Постройте алгоритм, который по двум входным словами $x = x[1]x[2]\dots x[n]$ и $y = y[1]y[2]\dots y[m]$, за $O(mn)$ операций находит длину наибольшей общей подпоследовательности, то есть максимальное k, для которого существуют $1 \leqslant i_1 < i_2 < \dots < i_k \leqslant n$ и $1 \leqslant j_1 < j_2 < \dots < j_k \leqslant m$ такие, что $x_{i_1}x_{i_2}\dots x_{i_k} = y_{j_1}y_{j_2}\dots y_{j_k}$.

Задача D.7. Дан прямоугольный кусок ткани размером $X \times Y$, где X, Y — положительные целые, $n \in \mathbb{N}$ и $\{(a_i, b_i, c_i) \mid i \in [1; n]\}$, где c_i — стоимость куска $a_i \times b_i$. Предложите эффективный алгоритм, производящий разрезание данного куска на куски размером $a_i \times b_i$ [для разных i, кусок одного размера может встречаться неоднократно, стороны кусков параллельны сторонам данного куска] так, чтобы суммарная стоимость разрезанных кусок была максимальной.