Алгоритмы и структуры данных. Домашнее задание №1

Выполнил студент Наседкин Дмитрий Сергеевич (группа 242)

Письменная часть

№ 1

Псевдокод:

```
func genRandomTriple(n, a, b, c) {
    t1 = genRandomInteger(1, n)
    t2 = genRandomInteger(1, n - 1)
    t3 = genRandomInteger(1, n - 2)

    if (t2 >= t1) {
        t2 += 1
    }

    if (t3 >= t1) {
        t3 += 1
    }
    if (t3 >= t2) {
        t3 += 1
    }

    a = min(t1, t2, t3)
    c = max(t1, t2, t3)
    b = t1 + t2 + t3 - a - b
}
```

Очевидно, что алгоритм работает за $O(1)$ (Не матожидание)

Теперь докажем, что алгоритм генерирует каждую тройку равновероятно:

- Для начала скажем, что a, b, c полученные после выполнения алгоритма будут различны и $1 \leq a < b < c \leq n$
- Очевидно, что всего существует $\binom{n}{3}$ таких троек(не от алгоритма), то есть раскрыв бином получим $\frac{n(n-1)(n-2)}{6}$
- То есть нам нужно доказать, троек от алгоритма возможно столько же, и правда мы генирируем $n(n-1)(n-2)$ различных троек, при этом после сортировки их число сокращается до:

$$\frac{n(n-1)(n-2)}{3!} = \frac{n(n-1)(n-2)}{6}$$

ч. т. д.

№ 2

Из школы знаем, что cnt_divisors(n) $= \prod_{i=1}^{m}(\alpha_i + 1)$

Для начала заметим, что если x_i не отсортированные, то непонятно как считать матожидание, так как появляются зависимости.

Поэтому будем считать, что xi(вместе с pi) отсортированы, тогда они разбиваются на непересекающиеся отрезки, внутри которых равные x.

Также заметим, так как отрезки непересекаются, то матожидание коэфицентов a при x станут независимыми, поэтому теперь

$$E(\xi) = \prod_{i=1}^{m}(E(\xi_i) + 1),$$

где:

- m - кол-во отрезков
- $E(\xi_i)$ - мат. ожидание коэфицента α при x для текущего отрезка

Введем следующую индикаторную величину:

$$I_k = \begin{cases} 1, & \text{если число выбрано} \\ 0, & \text{иначе} \end{cases}$$

Тогда матожидание значения коэфицента для отрезка (l, r):

$$E(\xi_i) = \sum_{l}^{r} E(I_l) = \sum_{l}^{r} p_l$$

Алгоритм

1. Отсортируем x_i вместе с p_i

2. Будем двигаться двумя указателями поддерживая текущий отрезок на котором равные числа, и как дошли до конца посчитаем $E(\xi_i) + 1$

3. Перемножим все посчитанные матожидания и получим искомое $E(\xi)$

Время работы алгоритма -

$$O(\text{время сортировки}) + O(\text{проход двумя указателями}) = O(n \log n) + O(n) = O(n \log n)$$

№ 3

Назовем цикл длины три **треугольником**.

Для начала скажем, что пусть у нас есть произовльный разрез $A \subset V$, тогда очевидно, что треугольники лежащие в A, V/A, так и вне разреза никак не влияют друг на друга(другими словами **независимы**).

Теперь рассмотрим произвольный треугольник (v, u, w), тогда существует всего 2 варианта когда он полностью войдет в компоненту:

$$(v, u, w) \in A \ \vee (v, u, w) \in V/A$$

Введем следующую индикаторную величину:

$$I_i = \begin{cases} 1, & i\text{-й треугольник в одной компоненте} \\ 0, & \text{иначе} \end{cases}$$

Тогда матожидание кол-ва треугольников:

$$E(\xi) = \sum_{i=1}^{k} E(I_i) = \sum_{i=1}^{k} p_i = \sum_{i=1}^{k} \frac{1}{2^3} + \frac{1}{2^3} = \sum_{i=1}^{k} \frac{1}{4} = \frac{k}{4}$$

Устная и письменная часть

№ 4

Как мы все прекрасно знаем матожидание равно:

$$E(\xi) = \sum_{\omega \in \Omega} \xi(\omega) * p(\omega),$$

в случае с перестановками получаем:

$$E(\xi) = \sum_{p} \xi(p) * \frac{1}{n!} = \frac{1}{n!} * \sum_{p} \xi(p)$$

Замечание:

Значение перестановки равно сумме всех чисел вплодь до первого отрицательного, то есть можно отдельно рассматривать все перестановки, обьединяя их по принципу **i-е число первое отрицательное**, назовем такие перестановки pi

Тогда получим:

$$E(\xi) = \frac{1}{n!} * \sum_{i=1}^{n} \sum_{pi} \sum_{j=1}^{i} pi[j]$$

Как считать сумму по всем перестановкам pi? Давайте считать что сумма всех неотрицательных чисел равна sum_{pos}, отрицательных - sum_{neg}, кол-во неотрицательных чисел - m(очевидно, что посчитав их вначале выполнения алгоритма их можно переиспользовать), тогда сумма по всем перестановкам pi разбивается на 2 части:

- "Позитивная"часть - сумма всех неотрицательных чисел в таких перестановках = (сумма всех подмножеств неотрицательных чисел из m элементов) * $m!$, чтобы ее посчитать левую часть давайте поссмотрим во сколько подмножеств войдет элемент x - получим $\binom{m-1}{i-1}$, тогда очевидна формула - $m! * sum_{pos} * \binom{m-1}{i-1}$

- "Отрицательная"часть - сумма всех отрицательных чисел в таких перестановках $= sum_{neg} * \binom{m}{i}$

Тогда итоговая формула:

$$E(\xi) = \frac{1}{n!} * \sum_{i=1}^{n} m! * sum_{pos} * \binom{m-1}{i-1} + sum_{neg} * \binom{m}{i}$$

Очевидно, что все элементы внутри суммы считаются за $O(1)$, кроме факториалов и биномов, но предпосчитав все факториалы до n, и это делается за $O(1)$

Время работы алгоритма -

$$O(\text{время предпосчета}) + O(\text{время подсчета формулы}) = O(n) + O(n) = O(n)$$

№ 5

Из школы мы знаем, что существует сферическая система координат, в которой каждая точка пространства образом задается с помощью 3 чисел (r, θ, ϕ), где r - расстояние от начала координат, угол θ - полярный угол, а угол ϕ - азимутальным. При этом $r \geq 0$, $0 \leq \theta \leq \pi$, $0 \leq \phi \leq 2\pi$.

Тогда идея следующая - взять равновероятно θ на отрезке $[0, \pi]$, а ϕ на отрезке $[0, 2\pi]$, ну и так как мы работаем на еденичной сфере $r = 1$.

Переход же в декартовую систему координат происходит по следующим формулам:

$$\begin{cases} x = r \sin\theta \cos\phi, \\ y = r \sin\theta \sin\phi, \\ z = r \cos\theta \end{cases}$$

Таким образом получим точку на сфере.

При этом очевидно, что алгоритм будет работать за $O(1)$ - так как требуется всего 2 генерации точек на отрезке.

Докажем **корректность**:

Сферическая система координат одназначно биективна декартовой, кроме случая когда $\sin\theta = 0$, то есть в точках, когда $\theta = 0$ или $\theta = \pi$, однако так как θ выбирается равновероятно на отрезке(на котором бесконечное!! количество точек) получим, что $p(\theta = 0 \vee \theta = \pi) = p(\sin\theta \neq 0) = 0$, то есть распределение останется равномерным.

ч. т. д.

№ 6

Воспользуемся геометрическим определением вероятности, а именно разобьем отрезок на 2 части:

- [0, p) соответствует результату 1
- [p, 1) соответствует результату 0

Тогда func от интервала $[l, r)$ будет следующая:

- Кинуть монетку
- Если выпал орел, то новый интервал $[l, \frac{l+r}{2})$, иначе $[\frac{l+r}{2}, r)$
- Далее вернуть 0/1 если p не лежит в интервале(то есть не возникает неопределенности насчет итогового вердикта), иначе запуститься еще раз, более формально:

$$\begin{cases} 1, & \text{если } r \leq \text{p} \\ 0, & \text{если } l > \text{p} \\ func(l, r), & \text{иначе} \end{cases}$$

Заметим, что единственное, что нам надо, это операции сравнения с p, что не запрещено.

Очевидно, что алгоритм **корректен**(из-за геометрической интерпритации вероятности), осталось доказать, что математическое ожидание количества бросков монеты $O(1)$.

Доказательство:

Для оценки матожидания введем следующую индикаторную величину:

$$I_k = \begin{cases} k, & \text{если текущий бросок стал последним,} \\ 0, & \text{иначе} \end{cases}$$

Тогда:

$$E(\xi) = \sum_{k=1}^{\infty} E(I_k) = 1 * \frac{1}{2} + 2 * \frac{1}{4} + \cdots + i * \frac{i}{2^i} + \ldots,$$

То есть нужно найти сумму ряда $\sum_{k}^{\infty} \frac{k}{2^k}$, тут или можно явно ее найти или показать что ряд сходится(например по признаку Даламбера), так или иначе получим, что ряд сходится к какому-то числу σ, а значит матожидание кол-ва бросков равно O(1).

ч. т. д.