Нижние оценки. Сортировки

1. Среди n камней есть один радиоактивный. Счётчиком Гейгера мы можем проверить для любой кучки камней, если ли среди них радиоактивный. За какое наименьшее количество проверок можно найти радиоактивный камень?

2. В клетках шахматной доски написали в каком-то порядке числа от 1 до 64, каждое по одному разу. Про любое множество клеток доски мы можем спросить, какие числа на них стоят, и нам выдают полный список. За какое наименьшее количество вопросов можно понять, где какие числа стоят?

3. Есть n монет разного веса. За одно взвешивание можно сравнить по весу любые две монеты.

1. Найдите самую тяжёлую и самую лёгкую за $\frac{3}{2}n + O(1)$ взвешиваний.

2. Докажите, что нельзя найти среди n монет самую тяжелую и самую лёгкую монету за менее чем $\frac{3}{2}n + c$ взвешиваний.

4. На вход задачи подаются натуральные числа n и $k < n$ и массив из n натуральных чисел, каждое из которых не превосходит k. Предложите лучший по асимптотике алгоритм, который сортирует массив.

5. Алгоритм сортировки пузырьком (BubbleSort) задан псевдокодом:

```
1 Function BubbleSort(a) :
2 |  repeat
3 |  |  for i = 1 to a.size − 1 do
4 |  |  |  if a[i − 1] > a[i] then
5 |  |  |  |  swap(a[i − 1], a[i]);
6 |  |  |  end
7 |  |  end
8 |  until не было выполнено ни одной операции swap;
9 end
```

1. Сколько операций делает сортировка пузырьком в худшем случае? Укажите вход, на котором это происходит.

2. Какой элемент окажется в ячейке $a[n-k]$ после k итераций цикла **repeat-until**?

3. Сортировка RandBubbleSort получается применением сортировки пузырьком к массиву, элементы в котором переставлены в случайном порядке. Оцените асимптотически математическое ожидание времени работы алгоритма RandBubbleSort.

6. 1. Найдите среди n монет самую тяжелую и вторую по тяжести монету за $n + \log n + c$ взвешиваний. 2^*. Докажите, что нельзя найти самую тяжелую и вторую по тяжести монету из n монет за менее чем $n + \log n + c$ взвешиваний.

7. Вам нужно построить алгоритм, который принимал бы на вход произвольную строку битов длины n, а выдавал бы 1, если в этой строке есть два последовательных бита 01, и 0 в противном случае. Есть простой алгоритм, проверяющий каждый бит входной строки, но можно ли сэкономить? Иными словами, существует ли алгоритм, проверяющий меньше n битов в худшем случае и при этом решающий задачу?

8. Есть n монет и чашечные весы. Одна из монет фальшивая — отличается по весу от остальных, но неизвестно легче ли она или тяжелее. Найдите фальшивую монету за три взвешивания для **а)** $n = 10$; **б)*** $n = 12$.