# Лабиринтики и quickunion

# ПРОСЫПАЕМСЯ!!!

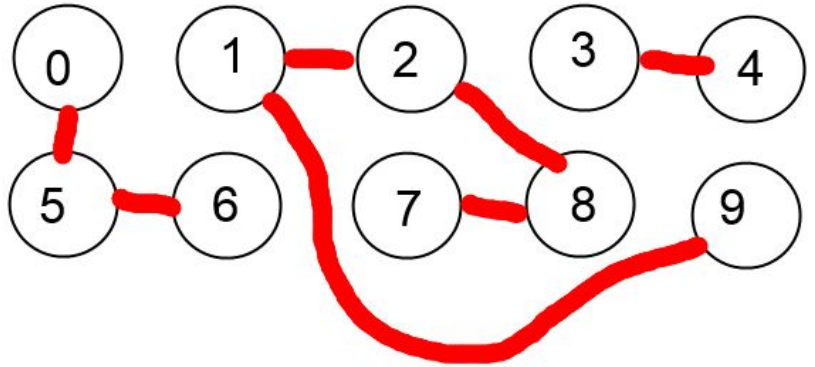
Сколько связанных компонент можно получить при выполнении следующей последовательности команд union на наборе из 10 элементов?

1-2; 3-4; 5-6; 7-8; 7-9; 2-8; 0-5; 1-9

Варианты:

- 1
- 2
- 3 < Прав ответ
- 4

решение:




P.S. Рисуем-соединяем, нумерация не имеет значения, так как ответ всегда один

# ПРОСЫПАЕМСЯ!!!

Какое максимальное число элементов массива id[] может быть изменено при выполнении одной команды union когда мы используем quick-find, а размер массива N?

Варианты:

- 1
- LgN
- N-1 < Прав ответ
- N

Предельный случай
0 1 2 3 4 5 6 7 8 9
0 1 0 0 0 0 0 0 0 0
Union(0,1)
Все элементы равные 0 меняются на 1,
их 9шт (10 -1 или N-1)

# ПРОСЫПАЕМСЯ!!!

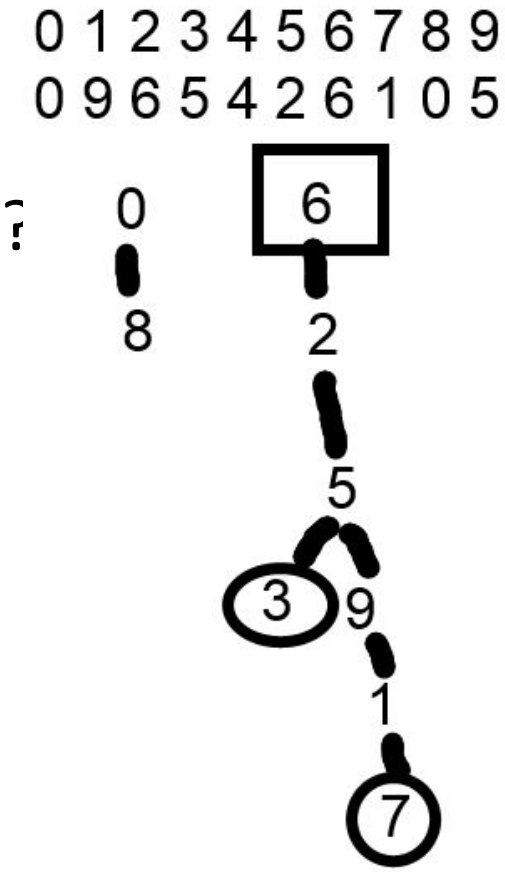
Есть структура данных quick-union из 10 элементов и их id[] соответственно равны

0 9 6 5 4 2 6 1 0 5

Чему равны корни 3 и 7, соответственно?

Варианты:

- 3 и 7
- 4 и 4
- 6 и 4
- 6 и 6 <Прав ответ



# ПРОСЫПАЕМСЯ!!!

Каким может быть максимальное число запросов к массиву во время операции поиска (connected) для алгоритма Quick-union от числа элементов N в массиве

Варианты:

- постоянно
- логарифмично
- Линейно < пр. отв
- Квадратично

Операция connected сравнивает корни 2ух чисел, возможна ситуация, когда числа находятся в самом низу
необходимо пройти весь массив от начала до конца, чтобы
узнать корни

# ПРОСЫПАЕМСЯ!!!

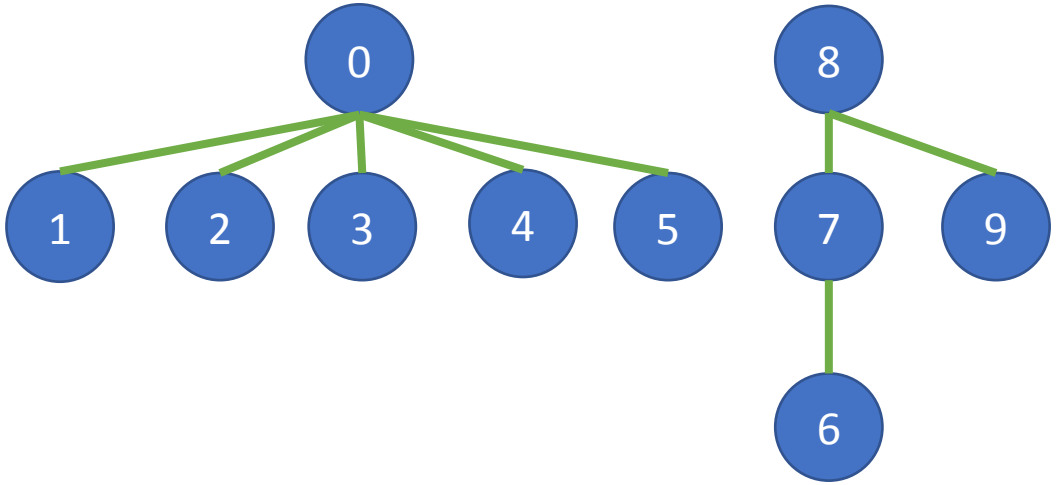
Дан массив id[] для WQU алгоритма

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 7 | 8 | 8 | 8 |

Какой id[] изменится при выполнении операции Union(3,6)

Варианты:

- ID[0]
- ID[3]
- ID[6]
- ID[8] < см. сл слайд



# Ответ

- Ответ – id[8]. Так как сравниваем алгоритмы по числу элементов, а не по высоте.
- У дерева с 0 вес ~ 6, а у дерева 8 вес ~ 4. По правилу присоединяем меньшее дерево к большему, сравнивая по весу => меняется id[8]

# Вопрос для суперигры

Когда открывается новый узел, сколько раз вызывается команда UNION()?

Варианты:

- 0,1,2,3 или 4 < пр. ответ
- 1,2,3 или 4
- 2,3 или 4
- 4

Сначала проверяем состояние соседей
(открыты или закрыты). С открытыми
Делаем команду union, с закрытыми – ничего не делаем. Поэтому вызовов
Union может быть от 0 до 4

# Анализ алгоритмов

# ПРОСЫПАЕМСЯ!

Дано: N = 1,000,000 (1 миллион)

Во сколько быстрее работает алгоритм $N*\log_2 N$ по сравнению с $N^2$?

- в 20
- в 1000
- в 50,000 < пр отв
- в 1,000,000

$$\frac{N*N}{N*\log N} = \frac{N}{\log N} = 10^6 / 20 = 50k$$

# Просыпаемся

- Вы измеряете время работы программы T(N) (сек), как функции размера входных данных N. Какая из этих функций лучше всего описывает время работы.

- $3.3 * 10^{-4} * N$
- $N^2$
- $5.0 * 10^{-9} * N^2$ < Пр отв.
- $6.25 * 10^{-9} * N^2$

Подставляем самый большой N
В каждую из функций и считаем T
Там где он будет точнее, та и функция

| N | T(N) |
|---|---|
| 1,000 | 0.0 |
| 2,000 | 0.0 |
| 4,000 | 0.1 |
| 8,000 | 0.3 |
| 16,000 | 1.3 |
| 32,000 | 5.1 |
| 64,000 | 20.5 |

# Просыпаемся!

- Сколько обращений к массиву происходит в следующем фрагменте кода?

$\sim 3N^2$

$\sim \frac{3}{2} N^2 \lg N$

$\sim \frac{3}{2} N^3$

$\sim 3N^3$

```
int sum = 0;
for (int i = 0; i < N; i++)
    for (int j = i+1; j < N; j++)
        for (int k = 1; k < N; k = k*2)
            if (a[i] + a[j] >= a[k]) sum++;
```

см. сл слайд

# Ответ

- Не все тройные циклы работают за кубическое время. В данном случае в цикл с параметром k работает 3lgN раз, вместо N раз, так как каждую итерацию k увеличивается вдвое.
- Первые 2 цикла работают по N раз каждый
- Итого 3/2 * N * N * lgN

# Просыпаемся!

Чему равно максимальное число обращений к массиву в алгоритме бинарного поиска в отсортированном массиве длины N

- Постоянно
- Логарифмично <пр отв.
- Линейно
- N log N (Linearithmic)

Каждый раз область поиска (обрабатываемый массив) Уменьшается в 2 раза при нахождении середины.
А это подобно логарифму по основанию 2

# Просыпаемся!

- Какая из следующих функций $O(N^3)$?

$11N + 15 \lg N + 100$

$\frac{1}{3} N^2$

$25,000 N^3$

сл.слайд

All of the above

# Ответ

- O() определяет верхнюю границу роста функции, кубичность характерна для них всех.

# Просыпаемся!

- Сколько памяти (в байтах) использует взвешенный QuickUnionUF состоящий из массива в N элементов?

○ $\sim N$

○ $\sim 2N$

○ $\sim 4N$

○ $\sim 8N$

Вещественное число (int) занимает в памяти 4 байта
Если массив состоит из N веществ чисел,
то занимает 4*N байт памяти
В QuickUnion находится 2 целочисленных массива
значит он занимает ~ 8N байт

- Hint: 2 integer arrays size of N

# Стеки и очереди

# Просыпаемся!

Какой из следующих входных данных в стек не сможет воспроизвести следующий вывод на экран:

5 4 3 2 1

Варианты:

- 1 2 3 4 5 - - - - -
- 1 2 5 - 3 4 - - - -
- 5 - 1 2 3 -4 - - - < ответ
- 5 - 4 - 3 - 2-1

Он возвращает: 5 3 4 2 1
Почему? – смотри предыдущие слайды лекции про
стеки

# Просыпаемся!

- Дана ссылка first, ссылающаяся на 1ый элемент стека (реализация с пом списка связанных элементов), в котором минимум 2 элемента, что делает представленный код?

```
Node x = first;
while (x.next.next != null) {
    x = x.next;
}
x.next = null;
```

- Удаляет первый элемент в стеке
- Удаляет второй элемент стека
- Удаляет предпоследний элемент стека
- Удаляет последний элемент в стеке < пр отв

# Ответ

- X.next.next != null – проверяет, не является ли следующий за ним элемент последним в стеке. Так как если a.next == null, то он последний, так как уже ни на что не указывает. Соответственно, как только предпоследний элемент достигается по проходу цикла, условие в while уже не выполняется и происходит выход из цикла. После которого следующий за предпоследним элемент, то есть последний, удаляется (зануляется).

# Просыпаемся!

Предположим, что начиная с пустой структуры, мы производим N push() операций в стеке, реализованном на основе массива изменяемого размера. Как будет вызываться метод resize()?

- Постоянно
- Логарифмично < пр. отв
- Линейно
- Квадратично

Как обычно имеем степень двойки, то есть
Массив каждый раз увеличивается в 2 раза
1 2 4 8 16 32 и тд пока не дойдем до N
Типа.. Массив растет от 1 до N каждый раз
увеличиваясь в 2 раза

# Сортировка

# ПРОСЫПАЕМСЯ!

Число сравнений для уже отсортированного входного массива алгоритма сортировки с выбором…

- Линейно
- Квадратично < ответ на предыдущем слайде (сортировка #22)
- Логарифмично
- Экспоненциально

# Просыпаемся

Число сравнений для уже отсортированного массива для метода сортировки с вставкой…

- Постоянно
- Логарифмично
- Линейно < просто сравниваем соседствующие элемены, левый эл-т всегда будет меньше правого, условие всегда будет выполняться, следовательно пройдет всего N сравнений
- Квадратично

# ПРОСЫПАЕМСЯ!

Число сравнений для уже отсортированного массива для метода сортировки Шелла при инкременте 3x+1…

- Постоянно
- Логарифмично
- Линейно
- N*logN < на слайде 48 ответ

# Просыпаемся!

- Каково максимальное число раздач 52 карт?

- $2^{52}$
- $52 \cdot 52$
- $52!$
- $52^{52}$

52! На предыдущем слайде ответ

# Просыпаемся!

Максимальное число вершин на выпуклой оболочке для множества из N точек…

- Постоянно
- Логарифмично
- Линейно < пр ответ
- N*logN

В предельном случае:
Сколько точек, столько и вершин.
4 точки – квадрат
N точек – N-угольник