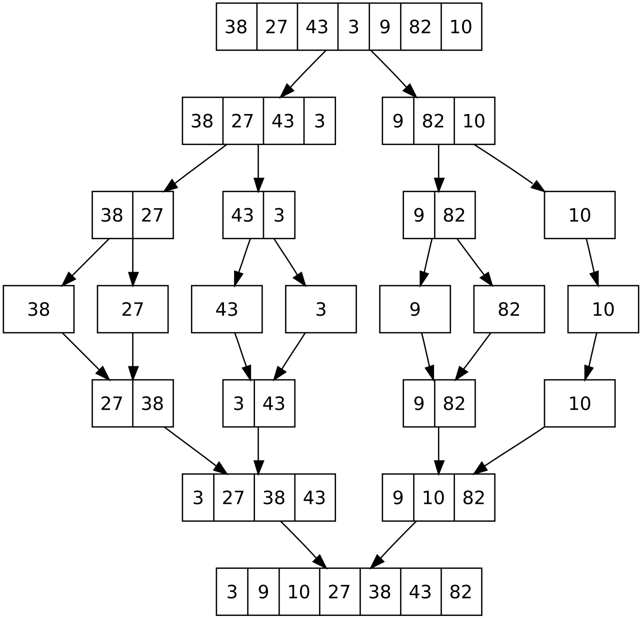
Programmeren (Ectrie)

Lecture 7: Sorting continued, searching

Tommi Tervonen

Econometric Institute, Erasmus University Rotterdam

Mergesort


38 27 43 3 9 82 10
38 27 43 3
9 82 10
38 27
43 3
9 82
10
38
27
43
3
9
82
10
27 38
3 43
9 82
10
3 27 38 43
9 10 82
3 9 10 27 38 43 82

Video: Merge-sort with Transylvanian-saxon (German) folk dance

Mergesort: analysis

- Each step, two recursion steps with half size input (divide)
- After division, merge the lists: O(n)

$$\begin{aligned} T(n) &= 2T(n/2) + O(n) \\ &= O(n \log n) \end{aligned}$$

- Does NOT sort in place, but requires O(n) memory

Quicksort

1. Choose a pivot element (e.g. first)
2. Partition array so, that elements left are $\leq$ pivot, and elements right are $>$ pivot
3. Sort recursively until size < 2

Video: Quicksort with Hungarian folk dancers

QS complexity analysis: worst case

$$\begin{aligned} T(n) &= T(n-1) + O(n) \\ &= \sum_{k=1}^{n} O(k) \\ &= O(\sum_{k=1}^{n} k) \\ &= O(n^2) \end{aligned}$$

QS complexity analysis: base case

$$\begin{aligned} T(n) &= 2T(n/2) + O(n) \\ &= O(n \log_2 n) \end{aligned}$$

QS analysis: guaranteed partitioning

$$\begin{aligned} T(n) &= T(9n/10) + T(n/10) + n \\ &= O(n \log_{10/9} n) \\ &= O(n \log n) \end{aligned}$$

- Similarly for any 1-to-x partitioning, where x is a constant

Randomized quicksort

```
function A = randomizedPartition(A, p, r)
    i = p + (round(rand(1) * (r-p)));
    swap(A, i, p); % pseudo-code
    A = partition(A, p, r);
end
```

Binary search

1	4	5	6	7	8

- Min-heap
- BST
- Array

Where we started

Elementary imperative programming:

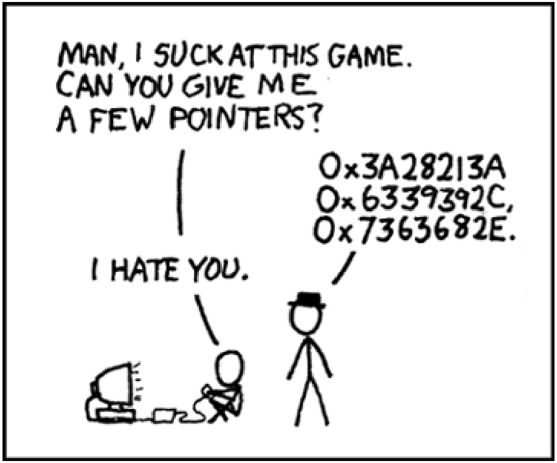
- Variables and methods
- Program flow
- Decisions and branching
- Control structures
- Bitwise operators
- Arithmetic operators
- Scoping

Where we are now

- Computational complexity
- Memory organization
- Fundamental algorithms: sorting (insertion, bubble, heap, merge, quick)
- Fundamental data structures: array, stack, queue, linked list, (binary) tree, heap
- Program correctness: pre- and post-conditions, unit testing, halting problem
- ... Matlab!

What next?

- Last exercise
- Exam
- Other courses: use it or lose it
- Voortgezet Programmeren: object oriented programming, software development


MAN, I SUCK AT THIS GAME. CAN YOU GIVE ME A FEW POINTERS?
0x3A28213A
0x6339392C,
0x7363682E.
I HATE YOU.