Computing the Median

Douglas Wikström
KTH Royal Institute of Technology
dog@kth.se

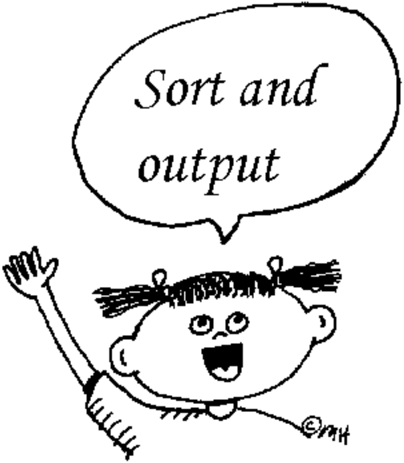
October 2020

Definition of Median (Distinct Numbers)

Definition. Given distinct numbers $x_1, x_2, \ldots, x_n$, the **median** is the $\frac{n}{2}th$ smallest of these numbers.

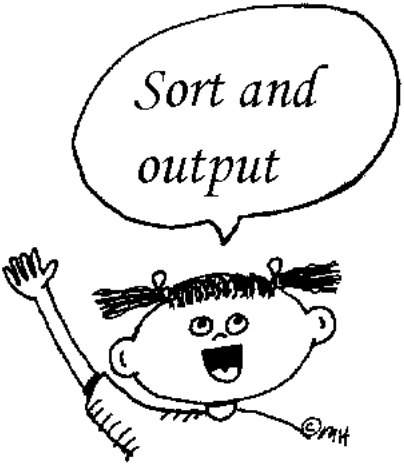
Example. Given $4, 2, 8, 5, 0$, the median is 4.

Sort and Output



Example. $4, 2, 8, 5, 0$

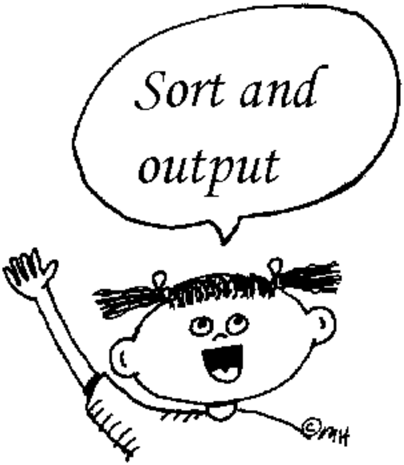
Sort and Output



Example. $4, 2, 8, 5, 0$

1. Sort gives $0, 2, 4, 5, 8$.

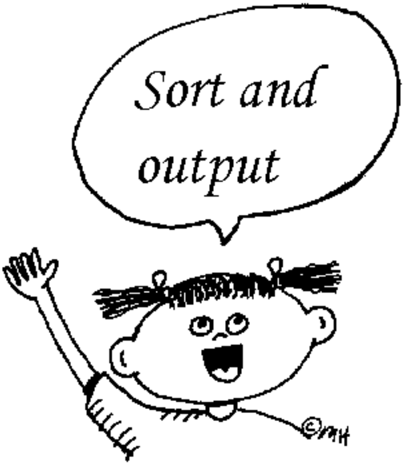
Sort and Output



Example. $4, 2, 8, 5, 0$

1. Sort gives $0, 2, 4, 5, 8$.
2. Output 4.

Sort and Output



Example. $4, 2, 8, 5, 0$

1. Sort gives $0, 2, 4, 5, 8$.
2. Output 4.

Runs in time $\Theta(n \log n)$

Generalize the median finding problem!

***k*-Selection Problem**

Given $X = [x_1, x_2, \ldots, x_n]$, output kth smallest value in X.

Generalize the median finding problem!

k-**Selection Problem**

Given $X = [x_1, x_2, \ldots, x_n]$, output kth smallest value in X.

$\textbf{MEDIAN}(X) = \textbf{SELECT}(X, n/2)$

Randomized k-Select Algorithm (1/3)

$k = 9$ 61 96 58 30 65 98 69 56 29 26 40 67 88 86 31 12 92

Randomized k-Select Algorithm (1/3)

$k = 9$ 61 96 58 30 65 98 69 56 29 26 40 67 88 86 31 12 92

Until median is found:

1. Choose random pivot.

Randomized k-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	[88]	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	[88]	96	98	92

Until median is found:

1. Choose random pivot.
2. Partition around pivot.

Randomized k-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	88	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	88	96	98	92
$k = 9$	61	58	30	65	69	56	29	26	40	67	86	31	12				

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	**[88]**	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	**[88]**	96	98	92
$k = 9$	61	58	30	65	69	**[56]**	29	26	40	67	86	31	12				

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized *k*-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	[88]	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	[88]	96	98	92
$k = 9$	61	58	30	65	69	[56]	29	26	40	67	86	31	12				
	30	29	26	40	31	12	[56]	61	58	65	69	67	86				

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	88	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	88	96	98	92
$k = 9$	61	58	30	65	69	56	29	26	40	67	86	31	12				
	30	29	26	40	31	12	56	61	58	65	69	67	86				
							$k = 2$	61	58	65	69	67	86				

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	**88**	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	**88**	96	98	92
$k = 9$	61	58	30	65	69	**56**	29	26	40	67	86	31	12				
	30	29	26	40	31	12	**56**	61	58	65	69	67	86				
						$k = 2$		61	58	65	69	**67**	86				

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	**88**	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	**88**	96	98	92
$k = 9$	61	58	30	65	69	**56**	29	26	40	67	86	31	12				
	30	29	26	40	31	12	**56**	61	58	65	69	67	86				
							$k = 2$	61	58	65	69	**67**	86				
								61	58	65	**67**	69	86				

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	**88**	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	**88**	96	98	92
$k = 9$	61	58	30	65	69	**56**	29	26	40	67	86	31	12				
	30	29	26	40	31	12	**56**	61	58	65	69	67	86				
						$k = 2$		61	58	65	69	**67**	86				
								61	58	65	**67**	69	86				
						$k = 2$		61	58	65							

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	88	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	88	96	98	92
$k = 9$	61	58	30	65	69	56	29	26	40	67	86	31	12				
	30	29	26	40	31	12	56	61	58	65	69	67	86				
						$k = 2$		61	58	65	69	67	86				
								61	58	65	67	69	86				
						$k = 2$		61	58	65							

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k = 9$ 61 96 58 30 65 98 69 56 29 26 40 67 88 86 31 12 92

61 58 30 65 69 56 29 26 40 67 86 31 12 88 96 98 92

$k = 9$ 61 58 30 65 69 56 29 26 40 67 86 31 12

30 29 26 40 31 12 56 61 58 65 69 67 86

$k = 2$ 61 58 65 69 67 86

61 58 65 67 69 86

$k = 2$ 61 58 65

61 58 65

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	**88**	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	**88**	96	98	92
$k = 9$	61	58	30	65	69	**56**	29	26	40	67	86	31	12				
	30	29	26	40	31	12	**56**	61	58	65	69	67	86				
							$k = 2$	61	58	65	69	**67**	86				
								61	58	65	**67**	69	86				
							$k = 2$	61	58	**65**							
								61	58	**65**							
							$k = 2$	61	58								

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	**88**	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	**88**	96	98	92
$k = 9$	61	58	30	65	69	**56**	29	26	40	67	86	31	12				
	30	29	26	40	31	12	**56**	61	58	65	69	67	86				
							$k = 2$	61	58	65	69	**67**	86				
								61	58	65	**67**	69	86				
							$k = 2$	61	58	**65**							
								61	58	**65**							
							$k = 2$	61	**58**								

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k=9$	61	96	58	30	65	98	69	56	29	26	40	67	**88**	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	**88**	96	98	92
$k=9$	61	58	30	65	69	**56**	29	26	40	67	86	31	12				
	30	29	26	40	31	12	**56**	61	58	65	69	67	86				
$k=2$								61	58	65	69	**67**	86				
								61	58	65	**67**	69	86				
$k=2$								61	58	**65**							
								61	58	**65**							
$k=2$								61	**58**								
								58	61								

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	88	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	88	96	98	92
$k = 9$	61	58	30	65	69	56	29	26	40	67	86	31	12				
	30	29	26	40	31	12	56	61	58	65	69	67	86				
$k = 2$								61	58	65	69	67	86				
								61	58	65	67	69	86				
$k = 2$								61	58	65							
								61	58	65							
$k = 2$								61	58								
								58	61								
$k = 1$									61								

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k = 9$	61	96	58	30	65	98	69	56	29	26	40	67	88	86	31	12	92
	61	58	30	65	69	56	29	26	40	67	86	31	12	88	96	98	92
$k = 9$	61	58	30	65	69	56	29	26	40	67	86	31	12				
	30	29	26	40	31	12	56	61	58	65	69	67	86				
$k = 2$								61	58	65	69	67	86				
								61	58	65	67	69	86				
$k = 2$								61	58	65							
								61	58	65							
$k = 2$								61	58								
								58	61								
$k = 1$									61								

Until median is found:

1. Choose random pivot.
2. Partition around pivot.
3. Continue on part containing median.

Randomized k-Select Algorithm (1/3)

$k = 9$ 61 96 58 30 65 98 69 56 29 26 40 67 88 86 31 12 92

61 58 30 65 69 56 29 26 40 67 86 31 12 88 96 98 92

$k = 9$ 61 58 30 65 69 56 29 26 40 67 86 31 12

30 29 26 40 31 12 56 61 58 65 69 67 86

$k = 2$ 61 58 65 69 67 86

61 58 65 67 69 86

Until median is found: $k = 2$ 61 58 65

61 58 65

$k = 2$ 61 58

58 61

$k = 1$ 61

61

Randomized k-Select Algorithm (1/3)

$k = 9$ 61 96 58 30 65 98 69 56 29 26 40 67 88 86 31 12 92

61 58 30 65 69 56 29 26 40 67 86 31 12 88 96 98 92

$k = 9$ 61 58 30 65 69 56 29 26 40 67 86 31 12

30 29 26 40 31 12 56 61 58 65 69 67 86

$k = 2$ 61 58 65 69 67 86

61 58 65 67 69 86

Until median is found:

$k = 2$ 61 58 65

61 58 65

$k = 2$ 61 58

58 61

$k = 1$ 61

30 29 26 40 31 12 56 58 61 65 67 69 86 88 96 98 92

Randomized k-Select Algorithm (2/3)

Select(X, k)
Input: Set of distinct numbers X and index k
Output: kth largest value in X

$z \leftarrow$ **RandomElement**(X)

$(L, R) \leftarrow$ **Partition**(X, z)

If $|L| + 1 = k$ **Then**
  return z
Else If $k \leq |L|$ **Then**
  return Select(L, k)
Else
  return Select$(R, k - (|L| + 1))$

Randomized k-Select Algorithm (3/3)

- A pivot is **good** if it divides the elements in two parts of roughly the same size.

Randomized *k*-Select Algorithm (3/3)

- A pivot is **good** if it divides the elements in two parts of roughly the same size.
- Most pivots are good with high probability.

Randomized k-Select Algorithm (3/3)

- A pivot is **good** if it divides the elements in two parts of roughly the same size.
- Most pivots are good with high probability.
- The algorithm runs in **expected linear** time.

Randomized k-Select Algorithm (3/3)

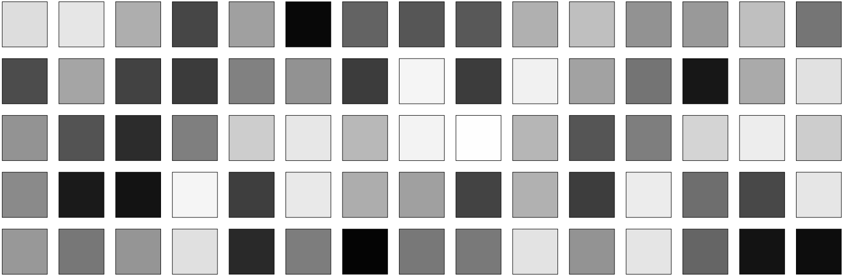
- A pivot is **good** if it divides the elements in two parts of roughly the same size.
- Most pivots are good with high probability.
- The algorithm runs in **expected linear** time.
- A more sophisticated algorithm runs in **expected time** $\approx 1.5n$.

Randomized k-Select Algorithm (3/3)

- A pivot is **good** if it divides the elements in two parts of roughly the same size.
- Most pivots are good with high probability.
- The algorithm runs in **expected linear** time.
- A more sophisticated algorithm runs in **expected time** $\approx 1.5n$.

What about deterministic linear time?

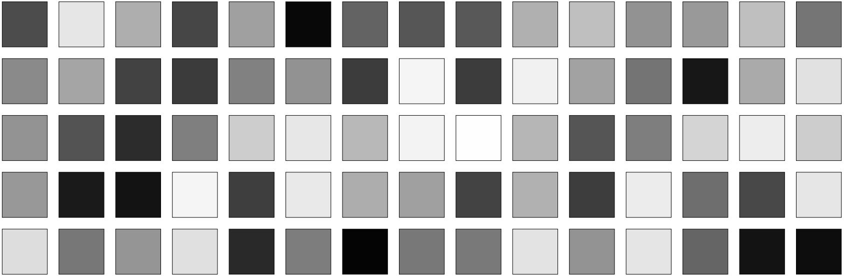
Median of Medians [Blum et al]



We are given n numbers, illustrated as squares. The bigger a number is, the **darker** its square is.

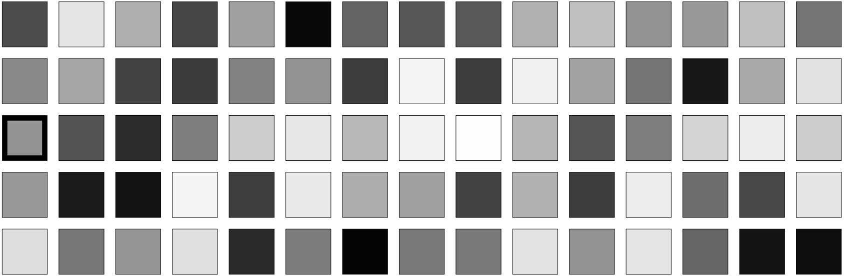
We must choose a pivot that partitions the numbers roughly evenly.

Median of Medians [Blum et al]



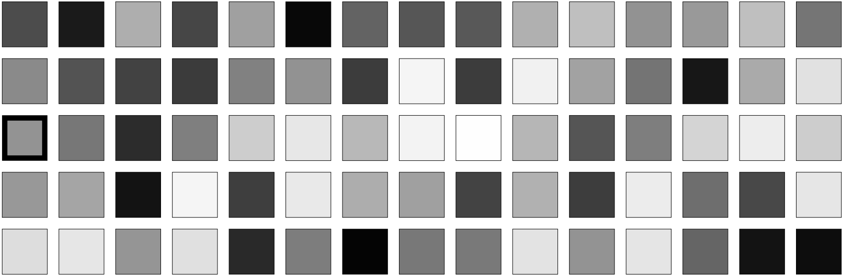
1. Sort each column and find its median.

Median of Medians [Blum et al]



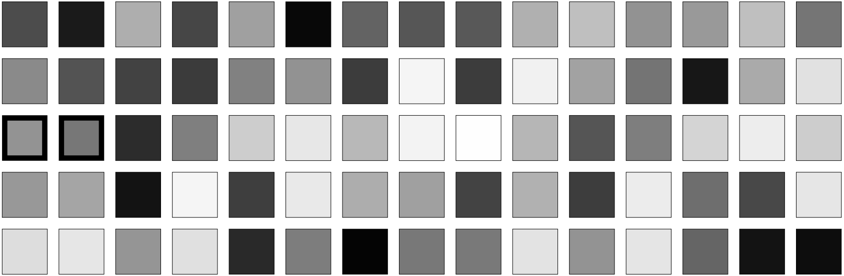
1. Sort each column and find its median.

Median of Medians [Blum et al]



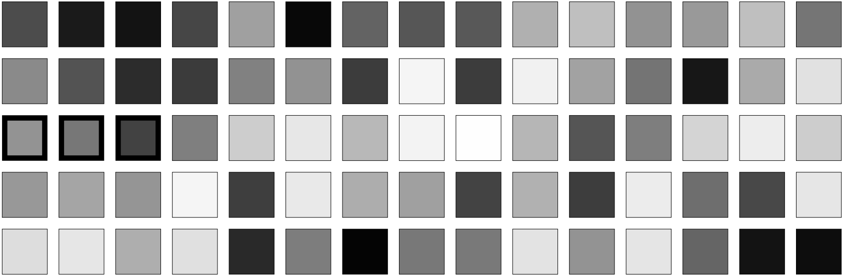
1. Sort each column and find its median.

Median of Medians [Blum et al]



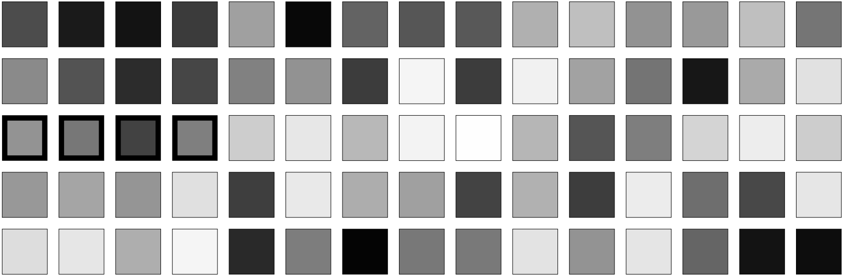
1. Sort each column and find its median.

Median of Medians [Blum et al]



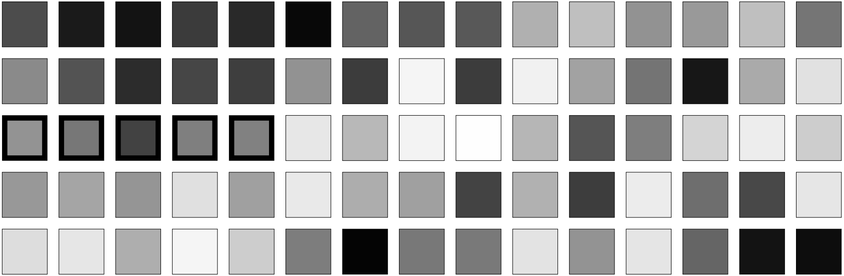
1. Sort each column and find its median.

Median of Medians [Blum et al]



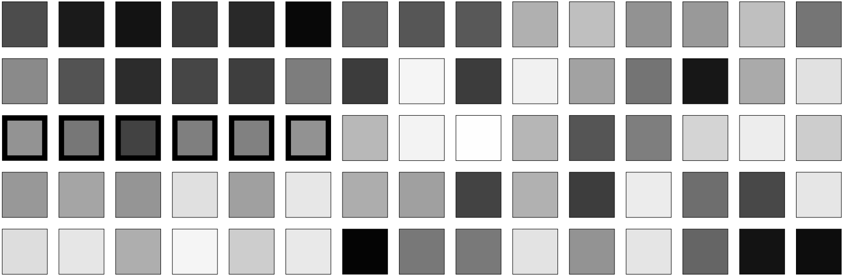
1. Sort each column and find its median.

Median of Medians [Blum et al]



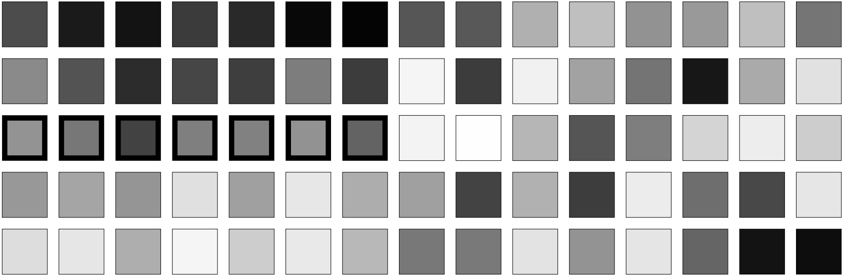
1. Sort each column and find its median.

Median of Medians [Blum et al]



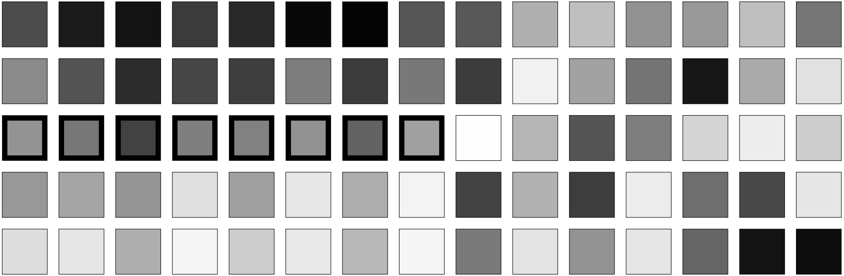
1. Sort each column and find its median.

Median of Medians [Blum et al]



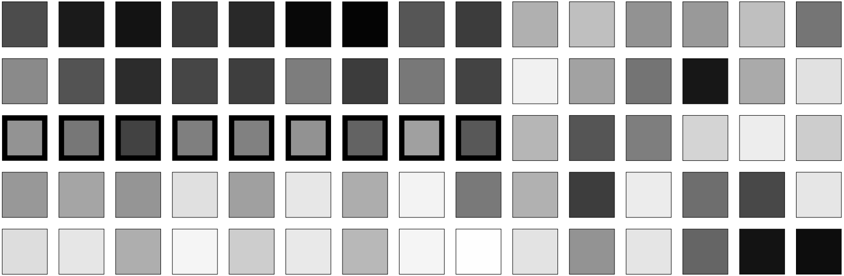
1. Sort each column and find its median.

Median of Medians [Blum et al]



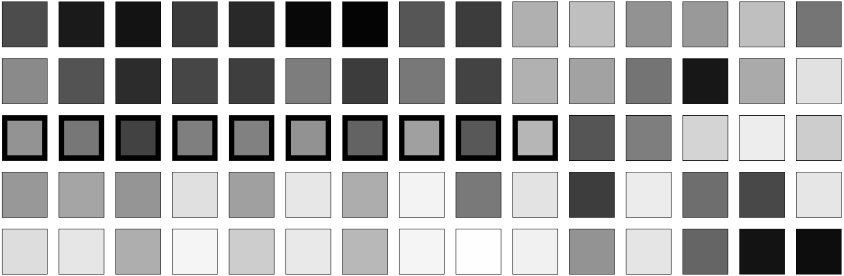
1. Sort each column and find its median.

Median of Medians [Blum et al]



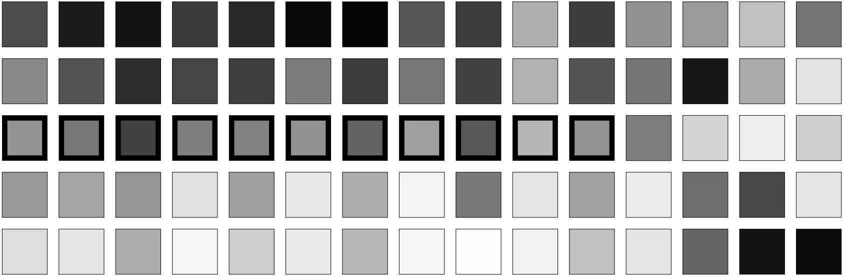
1. Sort each column and find its median.

Median of Medians [Blum et al]



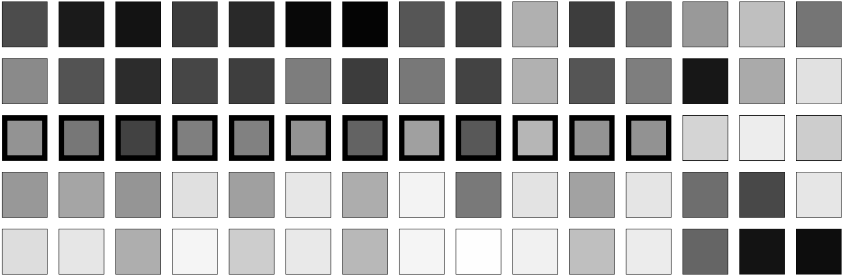
1. Sort each column and find its median.

Median of Medians [Blum et al]



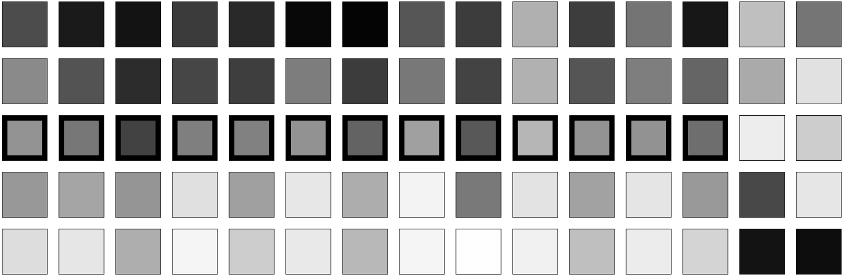
1. Sort each column and find its median.

Median of Medians [Blum et al]



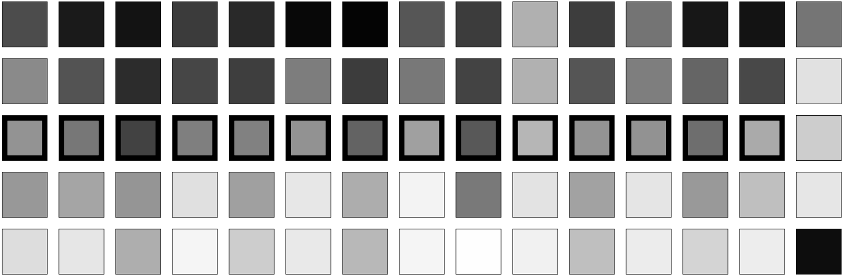
1. Sort each column and find its median.

Median of Medians [Blum et al]



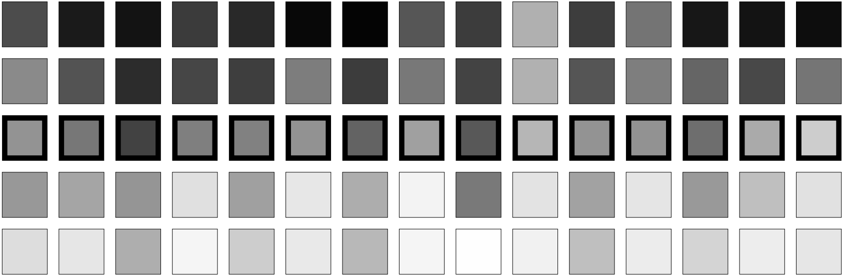
1. Sort each column and find its median.

Median of Medians [Blum et al]



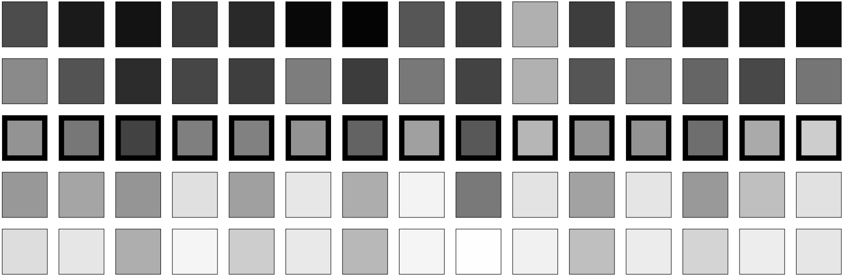
1. Sort each column and find its median.

Median of Medians [Blum et al]



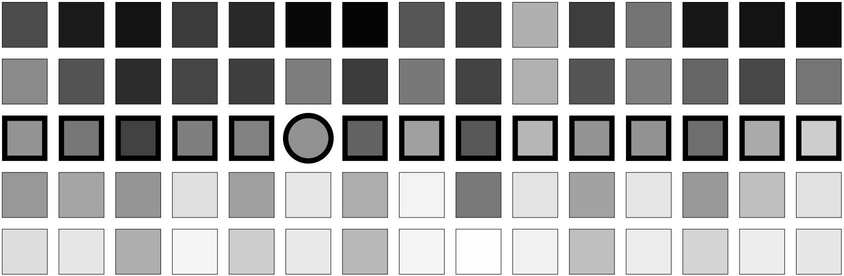
1. Sort each column and find its median.

Median of Medians [Blum et al]



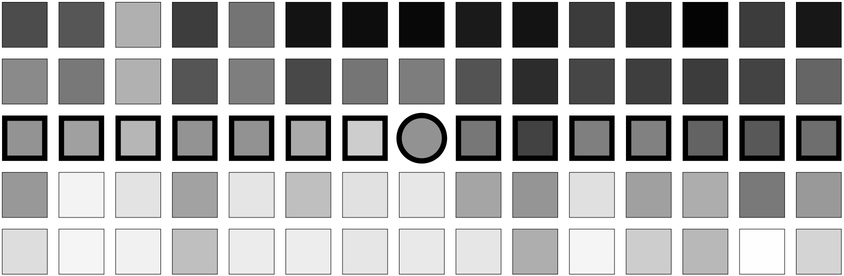
1. Sort each column and find its median.

Median of Medians [Blum et al]



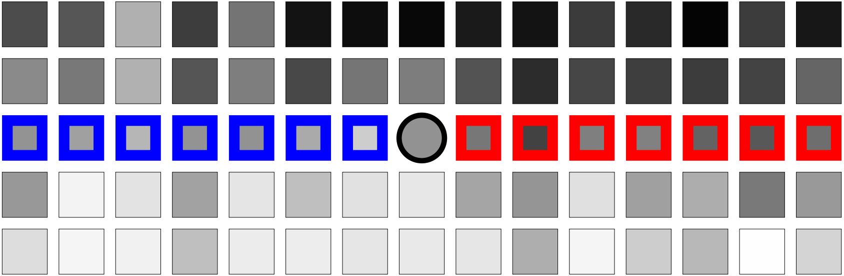
1. Sort each column and find its median.
2. Call **SELECT** on the $n/5$ medians to return the median of medians.

Quality of Median of Medians [Blum et al]



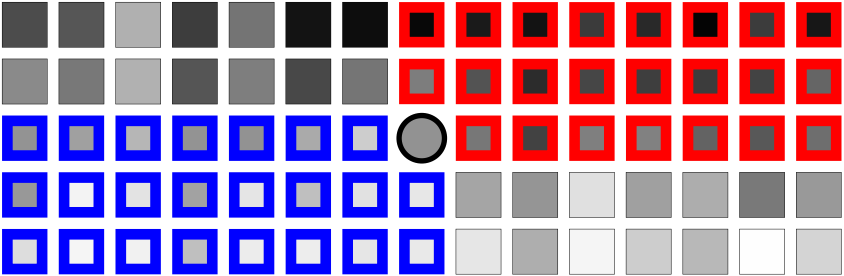
1. Partition columns around the pivot.

Quality of Median of Medians [Blum et al]



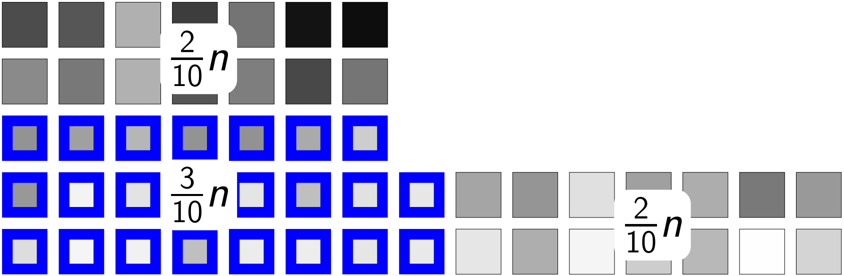
1. Partition columns around the pivot.
2. **BLUE** is smaller than pivot value and **RED** is bigger than pivot value.

Quality of Median of Medians [Blum et al]



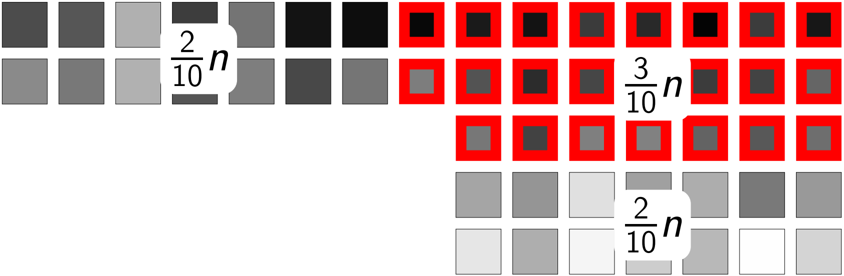
1. Partition columns around the pivot.
2. **BLUE** is smaller than pivot value and **RED** is bigger than pivot value.

Quality of Median of Medians [Blum et al]



1. Partition columns around the pivot.
2. **BLUE** is smaller than pivot value and **RED** is bigger than pivot value.
3. **Recursive call on at most $\frac{7}{10}n$ elements.**

Quality of Median of Medians [Blum et al]



1. Partition columns around the pivot.
2. **BLUE** is smaller than pivot value and **RED** is bigger than pivot value.
3. **Recursive call on at most $\frac{7}{10}n$ elements.**

Worst Case Running Time

$T(n)$ – **Worst case running time on** n **numbers**

Cost	Step
$7n$	Sort each column.
$T(n/5)$	Recursive call to find median of medians.
$4n$	Partition numbers around median of medians.
$T(\frac{7}{10}n)$	Recursive call on part containing median.

$$T(n) = T(n/5) + T(7n/10) + 11n$$

Worst Case Running Time Is Linear

$$T(n) = T(n/5) + T(7n/10) + 11n$$

1. Set $T(n) = cn$. Show that $c > 0$ exists.

Worst Case Running Time Is Linear

$$T(n) = T(n/5) + T(7n/10) + 11n$$

1. Set $T(n) = cn$. Show that $c > 0$ exists.
2. $cn = \frac{1}{5}cn + \frac{7}{10}cn + 11n$

Worst Case Running Time Is Linear

$$T(n) = T(n/5) + T(7n/10) + 11n$$

1. Set $T(n) = cn$. Show that $c > 0$ exists.
2. $cn = \frac{1}{5}cn + \frac{7}{10}cn + 11n$
3. $c = 110$ solves above, so $T(n) \leq 110n$.

What Is the True Answer?

Complexity	Year	Author
$5.43n$	1973	Blum et al
$3n + o(n)$	1976	Schönhage et al
$2.95n + o(n)$	1995	Dor & Zwick
$\vdots$	$\vdots$	$\vdots$
True answer!	**?**	**?**
$\vdots$	$\vdots$	$\vdots$
$(2 + \epsilon)n$	1996	Dor & Zwick
$\frac{79}{43}n + O(1)$	1982	Poblete & Munro
$\frac{38}{21}n + O(1)$	1976	Yap
$1.75n$	1973	Yao & Pratt
$1.5n$	1973	Blum et al

Conclusion

- We can prove that **randomness helps!**
- Clever ideas give **linear deterministic algorithm**.
- For a seemingly simple problem, it is **still unknown how many operations are needed!**