Friday Challenge #001

The task

- You are given a stream of numbers (IEnumerable<double>).
- The goal is to output a stream of medians (IEnumerable<double>).

Median – the definition

- Median is a number in the middle of the sorted population.

- For an array {1, 2, 3}, median is 2.
- For an array {1, 2, 3, 4} median is (2+3)/2.0 = 2.5.

Straightforward implementation

- Keep sorted list of numbers (an invariant).
- Find a value in the middle
  (either at N/2 or average of N/2 ceiling and floor).
- Rely on iterator block to produce output stream.

Straightforward implementation - notes

- Time complexity: O(N*N*lnN)
  (due to sorting of the list for each new number)
- Space complexity: O(N)

- Can we do better?

Insertion sort implementation

- Keep sorted list of numbers (an invariant).
  No need to sort **all** values each time,
  just find the correct place to **insert**.

- Find a value in the middle
  (either at N/2 or average of N/2 ceiling and floor).

- Rely on iterator block to produce output stream.

Insertion sort implementation- notes

- Time complexity: O(N*N)
  (due to linear insertion time of each new number)
- Space complexity: O(N)

- Can we do better?

PriorityQueue (heap) implementation

- Keep descending left queue and ascending right queue.
- Keep them balanced (an invariant)
  – max item below the middle is in left queue
  min item above the middle is in the right queue.

- Find a value in the middle
  (either the bigger queue top or an average of both queues tops).

- Rely on iterator block to produce output stream.

PriorityQueue (heap) implementation - notes

- Time complexity: O(N*lnN)
  (due to logarithmic insertion/extraction time of each new number)
- Space complexity: O(N)

- Is possible out of the box since .NET 6.

- Can we do better?

Benchmark

```
BenchmarkDotNet=v0.13.2, OS=Windows 10 (10.0.19044.2006/21H2/November2021Update)
Intel Core i7-8750H CPU 2.20GHz (Coffee Lake), 1 CPU, 12 logical and 6 physical cores
.NET SDK=7.0.100-preview.7.22377.5
  [Host]     : .NET 6.0.9 (6.0.922.41905), X64 RyuJIT AVX2
  DefaultJob : .NET 6.0.9 (6.0.922.41905), X64 RyuJIT AVX2


|                    Method |       source |         Mean |        Error |       StdDev |
|-------------------------- |------------- |-------------:|-------------:|-------------:|
|        GetMediansQuickSort | Int32[10000] | 323,722.3 us |  4,326.37 us |  4,046.89 us |
|        GetMediansQuickSort | Int32[10000] | 555,288.4 us | 10,749.86 us | 10,055.43 us |
|        GetMediansQuickSort | Int32[10000] | 230,661.3 us |  3,656.80 us |  3,420.58 us |
|        GetMediansQuickSort | Int32[10000] | 402,076.0 us |  1,980.16 us |  1,755.36 us |
|    GetMediansInsertionSort | Int32[10000] | 289,283.4 us |  2,663.22 us |  2,360.88 us |
|    GetMediansInsertionSort | Int32[10000] | 287,131.4 us |  3,971.78 us |  3,715.21 us |
|    GetMediansInsertionSort | Int32[10000] | 149,319.5 us |    784.18 us |    654.83 us |
|    GetMediansInsertionSort | Int32[10000] |  48,036.3 us |    238.92 us |    223.48 us |
|    GetMediansPriorityQueue | Int32[10000] |     937.1 us |     14.24 us |     11.89 us |
|    GetMediansPriorityQueue | Int32[10000] |     959.8 us |      5.65 us |      5.29 us |
|    GetMediansPriorityQueue | Int32[10000] |     847.8 us |      4.49 us |      3.98 us |
|    GetMediansPriorityQueue | Int32[10000] |   1,476.2 us |     17.10 us |     15.16 us |
```

The winners

- Egemen Ciftci
- Artem Dvornik

Thanks for your attention!