

# Profiling report

Názov projektu: (j)Elitná Kalkulačka

Apríl 2020

## 1 Profiling report

Na vypočítanie výberovej smerodatnej odchylky sme pre náš projekt použili súbor s 10, 100 a 1000 číslami - kladnými aj zápornými.

Na základe pozorovania môžeme vidieť, že program najviac vyvoláva funkcie Pow a Add – teda mocnenie a sčítavanie.

Pre optimalizáciu by sme sa mali zamerať najviac práve na tieto dve funkcie.

## 2 Výsledky profilingu

Výsledok profilingu s 10 číslami:

|            |                  |                   |   |
|------------|------------------|-------------------|---|
| ▲ 100,00 % | CalcStdDeviation | • 2 ms • 1 call   | • Kalkulator.StdDeviation.CalcStdDeviation(List)            |
| ▼ 5,94 %   | MoveNext         | • 0 ms • 11 calls | • System.Collections.Generic.List+Enumerator`1.MoveNext     |
| ▶ 5,54 %   | Pow              | • 0 ms • 11 calls | • Kalkulator.Calculator.Math.Math.Pow(Double, Double)       |
| 0,04 %     | Add              | • 0 ms • 20 calls | • Kalkulator.Calculator.Math.Math.Add(Double, Double)       |
| ▶ 0,04 %   | Root             | • 0 ms • 1 call   | • Kalkulator.Calculator.Math.Math.Root(Double, Double)      |
| 0,02 %     | get_Current      | • 0 ms • 10 calls | • System.Collections.Generic.List+Enumerator`1.get_Current  |
| 0,02 %     | GetEnumerator    | • 0 ms • 1 call   | • System.Collections.Generic.List`1.GetEnumerator           |
| 0,02 %     | Divide           | • 0 ms • 2 calls  | • Kalkulator.Calculator.Math.Math.Divide(Double, Double)    |
| 0,01 %     | Substract        | • 0 ms • 2 calls  | • Kalkulator.Calculator.Math.Math.Substract(Double, Double) |
| 0,01 %     | get_Count        | • 0 ms • 1 call   | • System.Collections.Generic.List`1.get_Count               |
| 0,01 %     | Math..ctor       | • 0 ms • 1 call   | • Kalkulator.Calculator.Math.Math..ctor                     |
| 0,01 %     | Multiply         | • 0 ms • 1 call   | • Kalkulator.Calculator.Math.Math.Multiply(Double, Double)  |
| 0,01 %     | Dispose          | • 0 ms • 1 call   | • System.Collections.Generic.List+Enumerator`1.Dispose      |

Výsledok profilingu so 100 číslami:

|            |                  |                    |   |
|------------|------------------|--------------------|---|
| ▲ 100,00 % | CalcStdDeviation | • 2 ms • 1 call    | • Kalkulator.StdDeviation.CalcStdDeviation(List)            |
| ▶ 6,37 %   | Pow              | • 0 ms • 101 calls | • Kalkulator.Calculator.Math.Math.Pow(Double, Double)       |
| ▼ 5,74 %   | MoveNext         | • 0 ms • 101 calls | • System.Collections.Generic.List+Enumerator`1.MoveNext     |
| 0,33 %     | Add              | • 0 ms • 200 calls | • Kalkulator.Calculator.Math.Math.Add(Double, Double)       |
| 0,16 %     | get_Current      | • 0 ms • 100 calls | • System.Collections.Generic.List+Enumerator`1.get_Current  |
| ▶ 0,04 %   | Root             | • 0 ms • 1 call    | • Kalkulator.Calculator.Math.Math.Root(Double, Double)      |
| 0,02 %     | get_Count        | • 0 ms • 1 call    | • System.Collections.Generic.List`1.get_Count               |
| 0,02 %     | GetEnumerator    | • 0 ms • 1 call    | • System.Collections.Generic.List`1.GetEnumerator           |
| 0,02 %     | Divide           | • 0 ms • 2 calls   | • Kalkulator.Calculator.Math.Math.Divide(Double, Double)    |
| 0,01 %     | Substract        | • 0 ms • 2 calls   | • Kalkulator.Calculator.Math.Math.Substract(Double, Double) |
| 0,01 %     | Math..ctor       | • 0 ms • 1 call    | • Kalkulator.Calculator.Math.Math..ctor                     |
| 0,01 %     | Multiply         | • 0 ms • 1 call    | • Kalkulator.Calculator.Math.Math.Multiply(Double, Double)  |
| 0,01 %     | Dispose          | • 0 ms • 1 call    | • System.Collections.Generic.List+Enumerator`1.Dispose      |

Výsledok profilingu so 1000 číslami:

|            |                  |                      |   |
|------------|------------------|----------------------|---|
| ▲ 100,00 % | CalcStdDeviation | • 2 ms • 1 call      | • Kalkulator.StdDeviation.CalcStdDeviation(List)            |
| ▶ 11,14 %  | Pow              | • 0 ms • 1 001 calls | • Kalkulator.Calculator.Math.Math.Pow(Double, Double)       |
| ▶ 5,34 %   | MoveNext         | • 0 ms • 1 001 calls | • System.Collections.Generic.List+Enumerator`1.MoveNext     |
| 2,21 %     | Add              | • 0 ms • 2 000 calls | • Kalkulator.Calculator.Math.Math.Add(Double, Double)       |
| 1,07 %     | get_Current      | • 0 ms • 1 000 calls | • System.Collections.Generic.List+Enumerator`1.get_Current  |
| 0,03 %     | GetEnumerator    | • 0 ms • 1 call      | • System.Collections.Generic.List`1.GetEnumerator           |
| ▶ 0,03 %   | Root             | • 0 ms • 1 call      | • Kalkulator.Calculator.Math.Math.Root(Double, Double)      |
| 0,01 %     | Divide           | • 0 ms • 2 calls     | • Kalkulator.Calculator.Math.Math.Divide(Double, Double)    |
| 0,01 %     | Substract        | • 0 ms • 2 calls     | • Kalkulator.Calculator.Math.Math.Substract(Double, Double) |
| 0,01 %     | get_Count        | • 0 ms • 1 call      | • System.Collections.Generic.List`1.get_Count               |
| 0,01 %     | Math..ctor       | • 0 ms • 1 call      | • Kalkulator.Calculator.Math.Math..ctor                     |
| 0,01 %     | Dispose          | • 0 ms • 1 call      | • System.Collections.Generic.List+Enumerator`1.Dispose      |
| 0,01 %     | Multiply         | • 0 ms • 1 call      | • Kalkulator.Calculator.Math.Math.Multiply(Double, Double)  |