Contents

Some Notes on Complexity

Have a look at the Big-O complexity chart:

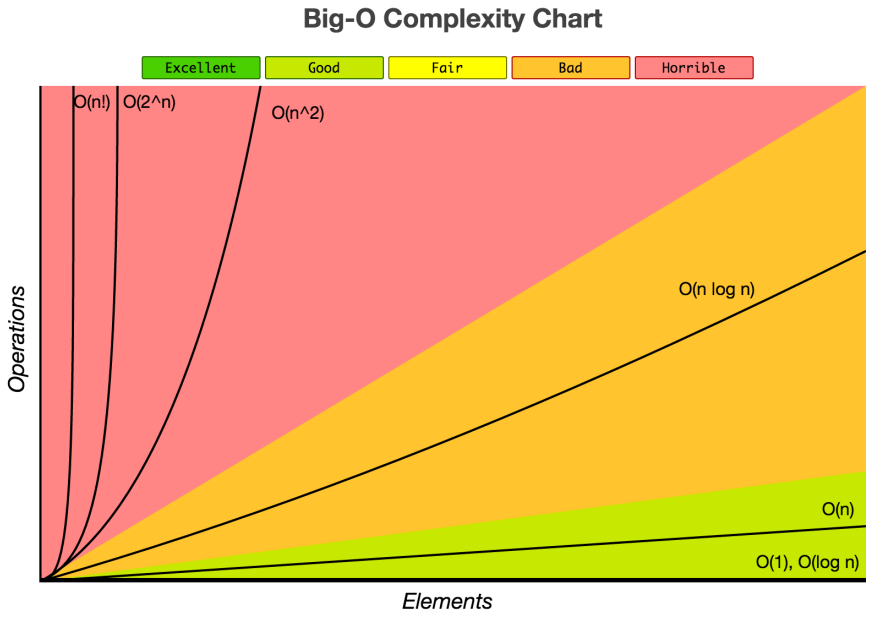

Figure 1: Big-O Complexity Chart

A function has an order, it can be for example

- constant (O(c)),
- logarithmic (O(log n)),
- linear (O(n)),
- linearithmic (O(n log n)),
- quadratic (O(n^2)),
- cubic (O(n^3)),
- exponential (O(c^n)),
- factorial (O(n!)).

This can make a *very* significant difference, as exemplified in the following code:

```
using System;
using System.IO;

class Program
{
    static int factorial(int n)
    {
        if (n == 0)
            return 1;
        else
            return checked(factorial(n - 1) * n);
    }

    static void Main()
    {
        Console.WriteLine(int.MaxValue);

        int x = int.MaxValue;
        int y = x + 1;
        Console.WriteLine(y);

        try
        {
            y = checked(x + 1);
        }
        catch (System.OverflowException)
        {
            Console.WriteLine("An overflow exception was
↪   thrown.");
        }

        int count0 = 0;
        int n0 = 3;
        try
        {
            while (true)
            {
                count0++;
                n0 = checked(n0 +1 );
            }
        }
        catch (System.OverflowException)
        {
            Console.WriteLine("An overflow exception was
↪   thrown after " + count0 + " iterations.");
```

```
            // An overflow exception was thrown after
             ↪  2147483645 iterations.
        }

        int count1 = 0;
        int n1 = 3;
        try
        {
            while (true)
            {
                count1++;
                n1 = checked(n1 * n1);
            }
        }
        catch (System.OverflowException)
        {
            Console.WriteLine("An overflow exception was
↪  thrown after " + count1 + " iterations.");
            // An overflow exception was thrown after 5
             ↪  iterations.
        }

        int count2 = 0;
        int n2 = 3;
        try
        {
            while (true)
            {
                count2++;
                n2 = factorial(n2);
            }
        }
        catch (System.OverflowException)
        {
            Console.WriteLine("An overflow exception was
↪  thrown after " + count2 + " iterations.");
            // An overflow exception was thrown after 3
             ↪  iterations.
        }

        int count3 = 0;
        int n3 = 100;
        try
        {
            while (true)
            {
```

```
                count3++;
                n3 = checked(n3 * (int)Math.Log(n3, 2));
            }
        }
        catch (System.OverflowException)
        {
            Console.WriteLine("An overflow exception was
↪   thrown after " + count3 + " iterations.");
            // An overflow exception was thrown after 7
             ↪  iterations.
        }
    }
}
```