Algorithms and Datastructures

Assignment 1

Lucas van der Laan
s1047485

1

(a)

$$
\begin{aligned}
n^2 &= 60 \times 10^6 \\
n &= \sqrt{60 \times 10^6} \\
&= 2000\sqrt{15} \\
&\approx 7745.97
\end{aligned}
$$

So we can process 7745 elements in 1 minute.

(b)

$$
\begin{aligned}
n \log_{10}(n) &= 60 \times 10^6 \\
n &= e^{W(60\times 10^6 \times \log(10))} \\
&\approx 8.64929600822... \times 10^6 \\
&\approx 8,649,296.0
\end{aligned}
$$

So we can process 8,649,296 elements in 1 minute.

(c)

$$
\begin{aligned}
2^n &= 60 \times 10^6 \\
n &= \log_2(60 \times 10^6) \\
&\approx 25.8
\end{aligned}
$$

So we can process 25 elements per minute.

(d)

$$
\begin{aligned}
n\sqrt{n} &= 60 \times 10^6 \\
n^3 &= 60 \times 10^8 \\
n &= \sqrt[3]{60 \times 10^8} \\
&= 1000
\end{aligned}
$$

So we can process 1000 elements per minute.

(e)

$$
\begin{aligned}
n^{100} &= 60 \times 10^6 \\
n &= \sqrt[100]{60 \times 10^6} \\
&\approx 1.196
\end{aligned}
$$

So we can process 1 element per minute.

(f)

$$
\begin{aligned}
4^n &= 60 \times 10^6 \\
\log_4(4^n) &= \log_4(60 \times 10^6) \\
n &= \log_4(60 \times 10^6) \\
&= \log_4(60 \times 10^6) \\
&= \frac{\log(60 \times 10^6)}{log(4)} \\
&\approx 12.919
\end{aligned}
$$

So we can process 12 elements per minute.

(g)

$$
n = 60 \times 10^6
$$

Because $n = 60 \times 10^6$, we can simply process 60 million elements per minute.

(h)

$$
\begin{aligned}
n^3 &= 60 \times 10^6 \\
n &= \sqrt[3]{60 \times 10^6} \\
&\approx 391.49
\end{aligned}
$$

So we can process 391 elements per minute.

(i)

$$
\begin{aligned}
n! &= 60 \times 10^6 \\
&\approx 11.17
\end{aligned}
$$

So we can process 11 elements per minute.

2

(a) True

(b) True

(c) True

(d) False

(e) False

(f) True

(g) False

3

(a) We have:

$$n + 1 \in \mathcal{O}(n)$$

Take:

$$c = 2, \qquad n_0 = 1$$

Then for $n \geq n_0$ we have

$$\begin{aligned} n + 1 &\leq c \cdot n \\ &\leq 2 \cdot n \end{aligned}$$

(b) We have:

$$n^3 + n + 2 \in \mathcal{O}(n^3)$$

Take:

$$c = 2, \qquad n_0 = 2$$

Then for $n \geq n_0$ we have

$$\begin{aligned} n^3 + n + 2 &\leq c \cdot n^3 \\ &\leq 2 \cdot n^3 \end{aligned}$$

(c) Didn't know how.

(d) Didn't know how.

4

The algorithm below is linear time, for $n \geq 0$, it will run n times, thus it is $T(n) = n$.

Algorithm 1 Factorial

Require: $n \geq 0$

procedure FACTORIAL(n)
    if $n == 0$ **then**
        return 1
    else
        $n \leftarrow n \cdot \text{FACTORIAL}(n - 1)$
    end if
    return n
end procedure

5

(a) $\mathcal{O}(n)$
$c_1 + c_2 + n(c_3 + c_4 + c_5) \in \mathcal{O}(n)$
$a \times n + b \in \mathcal{O}(n)$
$n \in \mathcal{O}(n)$
So: $a \times n \in \mathcal{O}(n)$
$b \in \mathcal{O}(n)$
So: $a \times n + b \in \mathcal{O}(n)$

(b) $\mathcal{O}(n^2)$
$c_1 + c_2 + n(c_3 + c_4 + n(c_5 + c_6 + c_7)) + c_8 \in \mathcal{O}(n^2)$
$n(b + n(a)) + c \in \mathcal{O}(n^2)$
$a \times n^2 + b \times n + c \in \mathcal{O}(n^2)$
$n^2 \in \mathcal{O}(n^2)$
So: $a \times n^2 \in \mathcal{O}(n^2)$
$n \in \mathcal{O}(n^2)$
So: $b \times n \in \mathcal{O}(n^2)$
So: $a \times n^2 + b \times n \in \mathcal{O}(n^2)$
$c \in \mathcal{O}(n^2)$
So: $a \times n^2 + b \times n + c \in \mathcal{O}(n^2)$

(c) $\mathcal{O}(n^3)$ $c_1 + c_2 + n(c_3 + c_4 + n^2(c_5 + c_6 + c_7)) + c_8 \in \mathcal{O}(n^3)$
$n(b + n^2(a)) + c \in \mathcal{O}(n^3)$
$a \times n^3 + b \times n^2 + c \in \mathcal{O}(n^3)$
$n^3 \in \mathcal{O}(n^3)$
So: $a \times n^3 \in \mathcal{O}(n^3)$
$n^2 \in \mathcal{O}(n^3)$
So: $b \times n^2 \in \mathcal{O}(n^3)$
So: $a \times n^3 + b \times n^2 \in \mathcal{O}(n^3)$
$c \in \mathcal{O}(n^3)$
So: $a \times n^3 + b \times n^2 + c \in \mathcal{O}(n^3)$

(d) $\Omega(1)$
This is because only with $n = 0$ is it not an inifite loop, thus only the Ω of $n = 0$ can be determined, which is constant time because $n = 0$ and the loop is never initiated.

(e) $\mathcal{O}(n^3)$
Worst case:
while i < n = n times
while j < i = n - 1 times
while k < j = n - 2 times

$c_1 + c_2 + n(c_3 + c_4 + (n-1)(c_5 + c_6 + (n-2)(c_7 + c_8 + c_9) + c_{10})) + c_{11} \in \mathcal{O}(n^3)$
$n(c + (n-1)((n-2)(a) + b)) + d \in \mathcal{O}(n^3)$
$n(c + (n-1)(an - 2a + b)) + d \in \mathcal{O}(n^3)$
$n(c + an^2 - an - 2an + 2a + bn - b) + d \in \mathcal{O}(n^3)$
$an^3 - an^2 - 2an^2 + bn^2 + 2an - bn + cn + d \in \mathcal{O}(n^3)$
$an^3 - 3an^2 + bn^2 + 2an - bn + cn + d \in \mathcal{O}(n^3)$
Just like before, all the other ns and constants can fit in the maximum of $\mathcal{O}(n^3)$.