Berkcan EKİCİ
171044015
CSE 321 – HW1

1.a) $(2^n+n^3) \in O(4^n)$

Definition: $f(n)$ is $O(g(n))$ if and only if there exists positive constants $c$ and $k$ such that

$|f(n)| \leq c \cdot |g(n)|$ for all $n \geq k$

$f(n) = 2^n+n^3$
$g(n) = 4^n$

$2^n+n^3 \leq c \cdot 4^n$ for all $n \geq k$

$2^n+n^3 \leq c \cdot 2^{2n}$ for all $n \geq k$

$\frac{2^n}{2^{2n}} + \frac{n^3}{2^{2n}} \leq c$ for all $n \geq k$ → $2^{-n} + \frac{n^3}{2^{2n}} \leq c$

choose $c=3$ $k=1$

$2^{-1} + \frac{1}{2} \leq 3$

Finding only one $c$ and $k$ is sufficient. [TRUE]

---

b) $\sqrt{10n^2+7n+3} \in \Omega(n)$

Definition: $f(n)$ is $\Omega(g(n))$ if $f(n) \geq c \cdot g(n)$ for all $n \geq k$ where $c>0$ and $k>0$

$f(n)$: $\sqrt{10n^2+7n+3}$
$g(n)$: $n$

$\sqrt{10n^2+7n+3} \geq c \cdot n$ for all $n \geq k$

$10n^2+7n+3 \geq c^2 \cdot n^2$ for all $n \geq k$

$\frac{10n^2}{n^2} + \frac{7n}{n^2} + \frac{3}{n^2} \geq c^2$ for all $n \geq k$ → $10 + \frac{7}{n} + \frac{3}{n^2} \geq c^2$

Choose $c=1$, $k=1$.

$10+7+3 \geq 1$
$20 \geq 1$ [TRUE]

---

c) $(n^2+n) \in o(n^2)$

Definition: If for any $c>0$ there exists an integer $k$ such that $f(n) < c \cdot g(n)$ for all $n \geq k$

$n^2+n < n^2 \cdot c$

$\frac{n^2}{n^2} + \frac{n}{n^2} < n^2 \cdot c$

$1 + \frac{1}{n} < c$ → Choose $c=1$, $k=1$
$2 < 1$
[FALSE]

$f(n) \neq o(n^2)$ because:
$|f(n)| / n^2 \not\to 0$ (the limit is 1)

Berkcan Ekici
171044015
CSE 321 - HW 1

d) $3 \cdot \log_2^2 n \in \Theta(\log_2^{n^2})$

$0 \leq c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$ for all $n \geq k$

$0 \leq c_1 \cdot \log_2^{n^2} \leq 3\log_2^2 n \leq c_2 \cdot \log_2 n^2$

$$0 \leq \frac{c_1 \cdot \log_2 n^2}{\log_2 n} \leq \frac{3\log_2^2 n}{\log_2 n} \leq \frac{c_2 \cdot \log_2 n^2}{\log_2 n}$$

$0 \leq c_1 \cdot 2 \leq 3\log n \leq c_2 \cdot 2$ [FALSE]

Is not true because $c_1$ and $c_2$ are constant.

---

e) $(n^3+1)^6 \in O(n^3)$ } iff $f(n) <= c \cdot g(n)$ for all $n >= k$ $c > 0$ and $k > 0$

$= (n^3+1)^6 \leq c \cdot n^3$

$= (n^3+1) \cdot (n^3+1)^5 \leq c \cdot n^3$

$= \frac{(n^3+1)}{n^3} \cdot (n^3+1)^5 \leq c$

$\boxed{= \left(1 + \frac{1}{n^3}\right) \cdot (n^3+1)^5 \leq c \quad \text{[FALSE]}}$ A contradiction

$n^{18} \notin O(n^3)$

CamScanner ile tarandı

Berkcan EKİCİ
171044015
CSE-321 HW I

2)

a) $2n\log(n+2)^2 + (n+2)^2 \cdot \log n/2$

$= 4 \cdot n\log(n+2) + (n^2+4n+4) \cdot (\log n - \log 2)$

$= 4n\log(n+2) + n^2\log n - n^2\log 2 + 4n\log n - 4n\log 2 + 4\log n - 4\log 2$

* Since this term is the highest degree, we must calculate the notation according to this term.

So $\boxed{g(n) \in \Theta(n^2 \log n)}$

b) $0.001n^4 + 3n^3 + 1$

→ As in the above example, we need to find the highest degree and calculate our notation according to this term.

→ The constant of the highest degree term is not important to us.

$0.001\, n^4$ → highest degree

→ Constant

So, $c_1 \cdot g(n) \leq 0{,}001n^4 + 3n^3 + 1 \leq c_2 \cdot g(n)$

$\boxed{g(n) \in \Theta(n^4)}$

Berkcan EKİCİ
171044015
CSE 321-HW I

3) Definiton : Rates of growth as $x \to \infty$

- f grows faster than g as $x \to \infty$ if

$$\lim_{x \to \infty} \frac{f(x)}{g(x)} = \infty$$

- equivalently, if

$$\lim_{x \to \infty} \frac{g(x)}{f(x)} = 0$$ } g grows slower than f as $x \to \infty$

- f and g row at the same rate as $x \to \infty$

$$\lim_{x \to \infty} \frac{f(x)}{g(x)} = L$$

a) $\log n$, $n^{\log n}$, $n^{1.5}$

$\log n$ vs $n^{\log n}$ :

$$\lim_{n \to \infty} \left( \frac{\ln(x)}{x^{\ln(x)}} \right) \xrightarrow{\text{Apply L'Hopital's Rule}} = \lim_{n \to \infty} \left( \frac{\frac{1}{n}}{\frac{2e^{\ln^2(n)} \ln(n)}{n}} \right) = \frac{\frac{1}{n}}{2e^{\ln^2(n)} \ln(n)} \cdot {}^{x} = \frac{1}{2 \cdot e^{\ln^2(n)} \ln(n)}$$

$$\frac{1}{2} \cdot \lim_{x \to \infty} \left( \frac{1}{e^{\ln^2 x} \cdot \ln x} \right) = \frac{1}{2} \cdot \frac{\lim_{n \to \infty} (1)}{\lim_{n \to \infty} (e^{\ln^2(n)} \cdot \ln(n))}$$

$\lim_{n \to \infty} 1 = 1$

$\lim_{n \to \infty} \left( e^{\ln^2(n)} \cdot \ln(n) \right) = \infty$

$$= \frac{1}{2} \cdot \underbrace{\frac{1}{\infty}}_{0} = \frac{1}{2} \cdot 0 = 0, \Rightarrow n^{\log n} > \log n$$

We know $n^{1.5} > \log n$

$n^{1.5}$ vs $n^{\log n}$ :

$$\lim_{n \to \infty} \frac{n^{1.5}}{n^{\log n}} = \lim_{n \to \infty} n^{1.5 - \log(n)} = \lim_{n \to \infty} e(\log(n)(1.5 - \log(n)))$$

$$= (e(\lim_{n \to \infty} \log(n)(1.5 - \log(n))))$$

CamScanner ile tarandı

$$= e\left(\infty\left(\lim_{n\to\infty} -\log(n) + 1.5\right)\right) = e\left(\infty\left(1.5 - \lim_{n\to\infty}\log(n)\right)\right)$$

$* \lim_{n\to\infty} \log(n) = \infty$

$$= e^{(1.5-\infty)\infty} = e^{-\infty \cdot +\infty} = 0$$

$$\boxed{n^{\log n} > n^{1.5} > \log n}$$

---

b) $n!, 2^n, n^2$

$$\rightarrow \lim_{n\to\infty} \frac{n!}{2^n} = \lim_{n\to\infty} \frac{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n}{2^n}$$ (Stirling's formula)

$$= \lim_{n\to\infty} \sqrt{2\pi n} \cdot \underbrace{\left(\frac{n}{2e}\right)^n}_{\infty} = \infty$$ So, $n! > 2^n$

$$\rightarrow \lim_{n\to\infty} \frac{2^n}{n^2} = \lim_{n\to\infty} \frac{2^n \cdot \log 2}{2 \cdot n} = \infty$$

$$\boxed{n! > 2^n > n^2}$$

---

c) $n\log n, \sqrt{n}$

$$\rightarrow \lim_{n\to\infty} \frac{n.\log n}{\sqrt{n}} = \lim_{n\to\infty} \sqrt{n}\log n = \underbrace{\lim_{n\to\infty}\sqrt{n}}_{\infty} \cdot \underbrace{\lim_{n\to\infty}\log(n)}_{\infty} = \infty$$

$$\boxed{n.\log n > \sqrt{n}}$$

CamScanner ile tarandı

Berkcan EKİCİ
171044015
CSE 321 – HW1

3.)

d) $n2^n, 3^n$

$$= \lim_{n\to\infty} \frac{n \cdot 2^n}{3^n} = \lim_{n\to\infty} \left(\frac{2}{3}\right)^n \cdot n$$

L'Hopital's rule:

$$\lim_{n\to\infty} \frac{n}{(\frac{3}{2})^n} \Rightarrow \lim_{n\to\infty} \frac{1}{(\frac{3}{2})^n \cdot \log(\frac{3}{2})} \Rightarrow \lim_{n\to\infty} \frac{(\frac{3}{2})^{-n}}{\log(\frac{3}{2})}$$

$$= \frac{\lim_{n\to\infty} (\frac{3}{2})^{-n}}{\log(\frac{3}{2})}$$

Since $\frac{3}{2} > 1$, $\lim_{n\to\infty} \left(\frac{3}{2}\right)^{-n} = 0$

$$= \frac{0}{\log(\frac{3}{2})} = 0 //$$

$$\boxed{3^n > n.2^n}$$

---

e) $\sqrt{n+10}$ , $n^3$

- Since $\sqrt{n+10}$ grows asymptotically slower than the polynomial $n^3$ as $n$ approaches $\infty$.

$$\lim_{n\to\infty} \frac{\sqrt{n+10}}{n^3} = 0$$

$$\boxed{n^3 > \sqrt{n+10}}$$

Berkcan EKİCİ
171044015
CSE 321 - HW1

4) algorithm1 (B[0... n-1, 0... n-1])

```
for i=0 to n-2 do
    for j=i+1 to n-1 do
        if B[i,j] != B[j,i]
            return false;
return true;
```

a) "The thing to do identify the most important operation of the algorithm, called the <u>basic operation</u>, the operation contributing the most to the total running time, and compute the number of times the basic operation is executed."

The basic operation is the: if $\boxed{A[i,j] \neq A[j,i]}$

b) $\sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} 1$

$= \sum_{i=0}^{n-2} [(n-1)-(i+1)+1] = \sum_{i=0}^{n-1} n-1-i$

$= n-1 \sum_{i=0}^{n-1} 1 - \sum_{i=0}^{n-2} i \Rightarrow (n-1).(n-1) - \frac{(n-1).(n-2)}{2}$

$= n^2-2n+1 - \frac{n^2-3n+2}{2}$

$= \frac{2n^2-4n+2-n^2+3n-2}{2} = \frac{n^2-n}{2}$

c) $\Theta(n^2)$

Berkcan EKİCİ
171044015
CSE 321 - HW1

5) algorithm 2(A[0...n-1, 0...n-1], B[0...n-1, 0...n-1])

```
for i=0 to n-1 do
    for j=0 to n-1 do
        C[i,j] = 0.0
        for k=0 to n-1 do
            C[i,j] = C[i,j] + A[i,k] * B[k,j]
return C
```

a) Multiplication and addition. On each repetition of the innermost loop each of the two is executed exactly once. We consider "multiplication" as the basic operation.

b) The number of multiplications made for every pair of specific values of variables i and j is: $\sum_{k=0}^{n-1} 1$ and the total number of multiplications is expressed by:

$$= \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} 1$$

$$= \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} n$$

$$= \sum_{i=0}^{n-1} n^2$$

$$= n^3$$

c) $\Theta(n^3)$

Berkcan EKİCİ
171044015
CSE 321 - HW1

6) Code:

```
for (int i=0; i<arr.length; ++i)
    for (int j=i+1; j<arr.length; ++j)
        if (arr[i] * arr [j] == desiredNumber)
            System.out.printf("(%d,%d)%n", arr[i], arr[j]);
```

Pseudo code:

```
// Input: An array A[0...n-1], a decimal integer value
// Output: Print pairs.

for i←0 to n-1 do
   for j←i+1 to n-1 do
      if A[i] * A[j] = desiredNumber
         print "A[i], A[j]"
      end-if
   end-for
end-for
```

- Algorithm is to run two loops to consider all possible pairs. For every pair, check if the A[i] * A[j] is equal to desiredNumber or not.

Time complexity:

$$C_{worst}(n) = \sum_{i=0}^{n-1} \sum_{j=i+1}^{n-1} 1 = \sum_{i=0}^{n-1} [(n-1)-(i+1)+1] = \sum_{i=0}^{n-1} n-1-i$$

Sum formula

$$= \sum_{i=0}^{n-1} (n-1-i) = (n-1)+(n-2)+\dots+1 = \frac{(n-1).n}{2}$$

$$= \Theta(n^2)$$