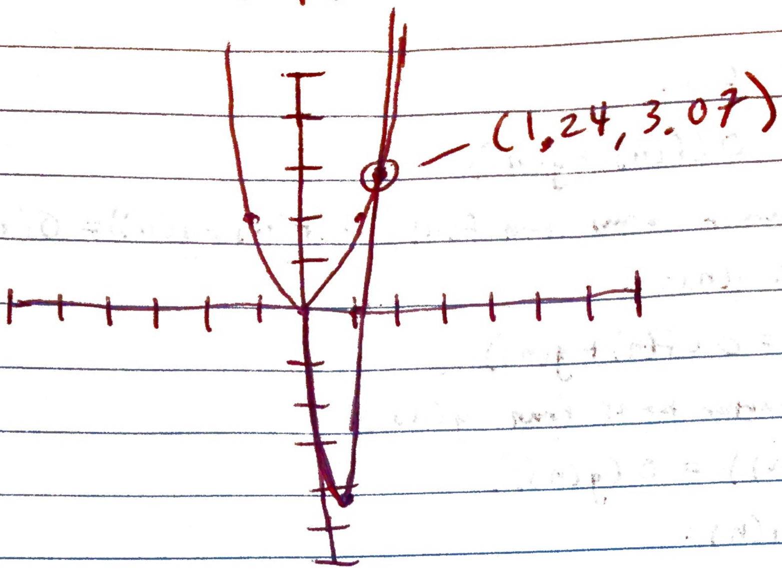
3.1)

On - Graph paper.

3.2)




3.8

[(1 = slowest growing) → (9 = fastest growing)]

1.) $2^{10} = O(1)$

2.) $2^{\log_2 n} = n^{\log 2} = n = O(n)$

3.) $4n = O(n)$

4.) $3n + 100\log_2 n = O(n)$

5.) $n \log_2 n = O(n \log_2 n)$

6.) $4 n \log_2 n = O(n \log_2 n)$

7.) $n^2 + 10n = O(n^2)$

8.) $n^3 = O(n^3)$

9.) $2^n = O(2^n)$

3.10) Def Big-O:

$$\exists\ n_0, c_0 \mid n \geq n_0,\ f(n) \leq c \times g(n),$$

Suppose that: d(n)

$$d(n) \in O(f(n)) \text{ and } e(n) \in O(g(n)),$$

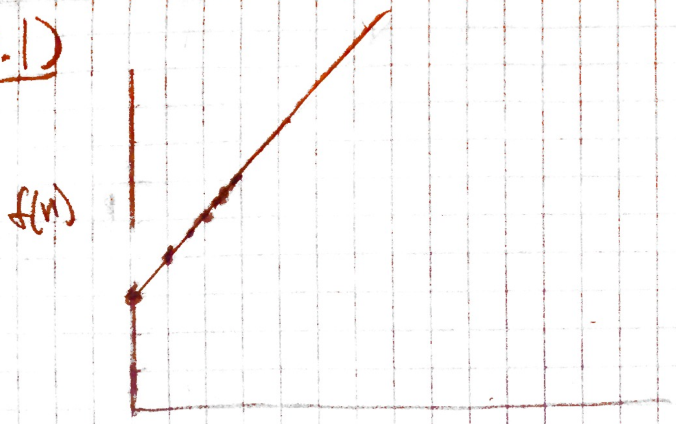
Show that: $(d(n) \times e(n)) \in O(f(n) \times g(n))$;

if $(d(n) \times e(n)) \leq c_1 f(n)\, c_2 g(n)$ for some $c_1, c_2$,

then $c_3 = c_1 \cdot c_2$ and $n_3 = n_1 \cdot n_2$ ∴ So,

$$\exists\ n_3, c_3 \mid n \geq n_3,\ (d(n) \times e(n)) \leq c_3 (f(n) \times g(n))$$

3.1)

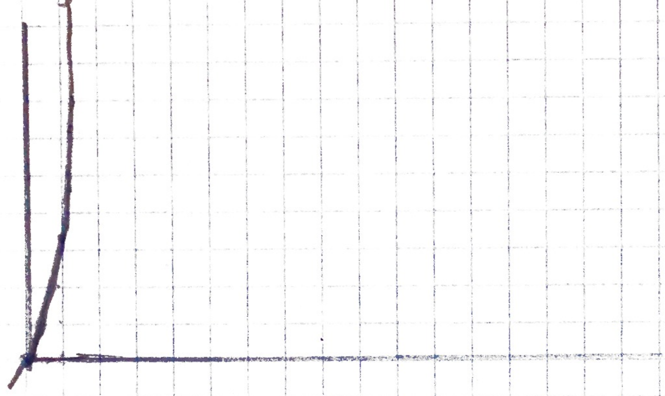



fn = 8n

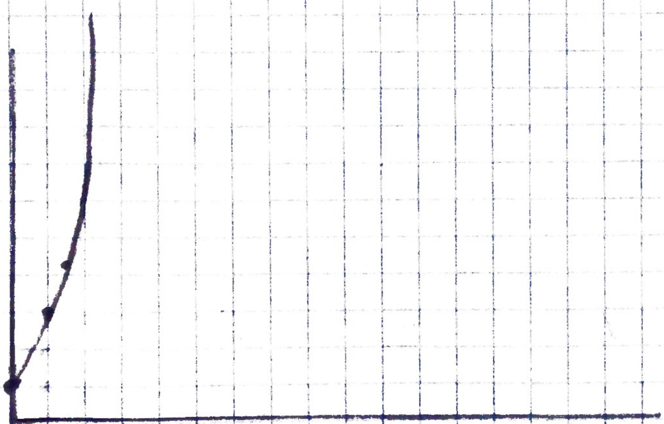
(1,8), (2,16), (3,24), (4,32), (5,40), (6,48)

↓ (0,3), (1,4), (1.58, 4.58), (2,5), (2.3,5.3), (2.5,5.5)

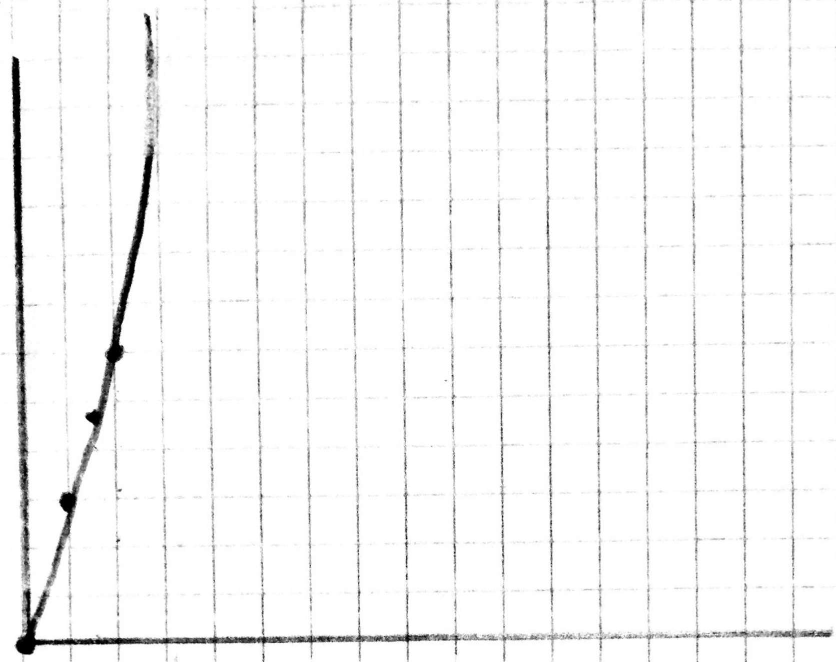
$f(n) = 4\, n \log n$



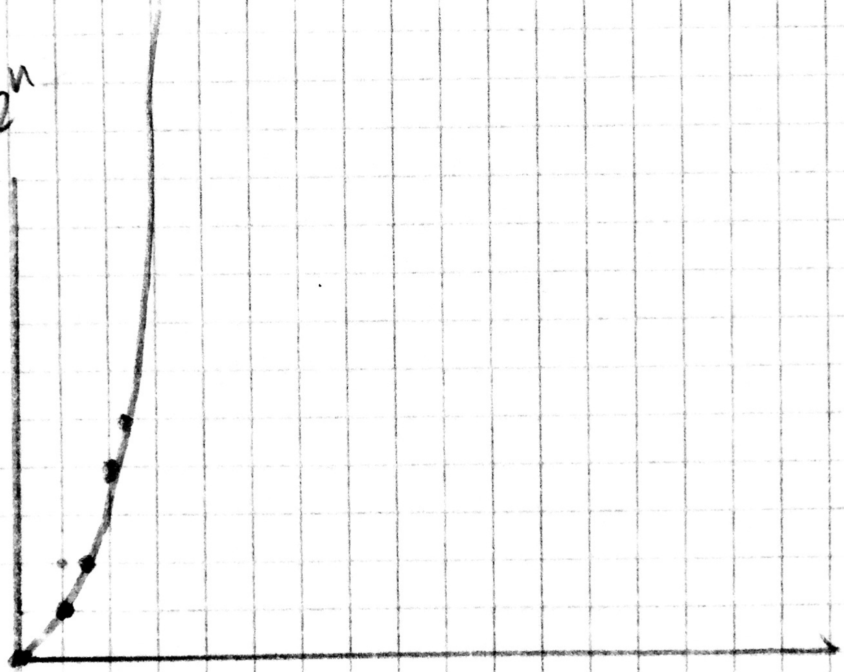
$f(n) = 2n^2$



$f(n) = n^3$



$f(n) = 2^n$



3.14 : Show that $O(\max\{f(n), g(n)\}) = O(f(n)+g(n))$ ?

Suppose that $f(n) > g(n)$:

$O(f(n)) = O(f(n)+g(n))$

- $g(n)$ has smaller terms than $f(n)$.
- So, $O(f(n)+g(n)) = O(f(n))$.

[• Since $f(n) > g(n)$, and we care about big-O, $g(n)$ gets thrown away.]

Suppose that $f(n) < g(n)$:

$O(g(n)) = O(f(n)+g(n))$

- $f(n)$ has smaller ~~[illegible]~~ terms than $g(n)$,

So $O(f(n)+g(n)) = O(g(n))$.

Suppose that $f(n) = g(n)$:

$O(f(n)) = O(f(n)+f(n))$

$= O(2(f(n)))$

- constants don't matter in Big-O

$= O(f(n))$

3.17.) Show that $(n+1)^5$ is $O(n^5)$:

By proposition 3.9, if there are no negative integers in the expanded notation

$(n+1)^5 = n^5 + 5n^4 + 10n^2 + 5n + 1 \leq (1+5+10+5+1)n^5 = c \cdot n^5$ for $c = 22$

3.18.) $2^{n+1}$ can be simplified as $2^n + 2^1$. Hence we can have $c = 2$ and $n_0 = 1$ and by prop. 3.9 and big-O definition $2^{n+1} = O(2^n)$

3.27) Each for loop adds 1 degree to the exponent of $n$. There is one for-loop that contains another 2 nested for-loops inside, thus examples is $O(n^3)$.

6

3.20) Def big-Omega:

$$\Omega(g(n)) = \{ f(n) \mid \exists n_0, c.\ \forall n \geq n_0.\ g(n) \leq (c \times f(n)) \}$$

Show that $n^2 \in \{$ this set $\}$:

$$\exists n_0, c.\ \forall n \geq n_0.\ c \times (n \lg n) \leq c \times n^2$$

Choose a $n_0$ and $c$ : $(n_0 = 0, c = 1)$

Let $k$ be an arbitrary int $\geq 0$,

$$c \times (k \lg k) \leq c \times k^2$$

$$\frac{k \lg k}{k} \leq \frac{k^2}{k} \Rightarrow \lg k \leq k \text{ for all } k \geq 0.$$

3.35) (Show that the function can be written in $O(n \lg n)$ time)

```
def disjoint3(A, B, C):
    new = []
    new += (A)   - n cost
    new += (B)   - n cost
    new += (c)   - n cost
    new.sort()   - n log n cost
    if len(set([x for x in new if new.count(x) > 2])) == 0:
        return True:
    else:
        return False
```

Cost $= 3n + n \log n$

$n \log n > 3n$, so $O(3n + n \log n) = O(n \log n)$.

3.41) By induction | $F(n) = \Omega((\frac{3}{2})^n)$

$c = (\frac{2}{3})^2 = 4/9$, so $f(n) \ge c(\frac{3}{2})^n$ for $n_0 = 1$

| $n$ | $f(n)$ | $(\frac{3}{2})^n$ | $c\,g(n)$ |
|---|---|---|---|
| 0 | 0 | 1 | 4/9 |
| 1 | 1 | 3/2 | 2/3 |
| 2 | 1 | 9/4 | 1 |
| 3 | 2 | 27/8 | 3/2 |

• Assume that induction hypothesis: $f(k) \ge c(\frac{3}{2})^k$ for some $k \ge n_0$

Show that $f(k+1) \ge c(\frac{3}{2})^{k+1}$

$f(k+1) \ge f(k) + f(k-1)$

$\ge c(\frac{3}{2})^k + c(\frac{3}{2})^{k-1}$ by inductive hypothesis,

$\ge c(\frac{3}{2}+1)(\frac{3}{2})^{k-1}$

$\ge c(\frac{3}{2})^2(\frac{3}{2})^{k-1}$, because $\frac{5}{2} > \frac{9}{4}$

$\ge c(\frac{3}{2})^{k+1}$

Hence, $f(n)$ is $\Omega((\frac{3}{2})^n)$.

4.3)

power(2,18) — $512 \cdot 512 = 262,144$

power(2,9) — $16 \cdot 16 \cdot 2 = 512$

power(2,4) — $4 \cdot 4 = 16$

power(2,2) — return $2 \cdot 2 = 4$

p(2,1) — return $1 \cdot 1 \cdot 2 = 2$

p(2,0) — return 1

4.5) puzzlesolve(3, (1, {a,b,c,d})

ps(2,a,{b,c,d}) ps(2,b,{a,c,d}) ps(2,c,{a,b,d}) ps(2,d,{a,b,c})

ps(1,ab,{c,d})
ps(1,ac,{b,d})
ps(1,ad,{b,c})
abc, abd
acb, acd
adb, adc

ps(1,ba,{c,d})
ps(1,bc,{a,d})
ps(1,bd,{a,c})
bda, bdc
bca, bcd
bac, bad

ps(1,ca,{b,d})
ps(1,cb,{a,d})
ps(1,cd,{a,b})
cda, cdb
cba, cbd
cab, cad

ps(1,da,{b,c})
ps(1,db,{a,c})
ps(1,dc,{a,b})
dca, dcb
dba, dbc
dab, dac

should be big-0

4.7)
```
def stringConverter(string, cnt):
    if not String:
        return Cnt
    return stringConverter(string[:1], 10 * cnt + ord(string[0]) - 48)
```

4.11
```
def uniqueElement(list):
    if len(list) == 1:
        return True
    else:
        dupe = list[0]
        other = list[1:]
        ~~unique~~ if dupe not in other:
          unique = dupe
        ~~recur if uniqueElement(~~
        recur = uniqueElement(other)
        return (uniqueness and recur)
```