CS2210 Asmt 1

Def: $f(n) \in O(g(n))$ if a constant multiple of $g \geq f$

1. We need to find constants $c>0$ and integer $n_0 \geq 1$ st. . .

Use def. of O to prove $\frac{1}{2n-1}$ is $O(\frac{1}{n})$

Def: Let $f(n)$, $g(n)$ be functions from $\mathbb{I}$ to $\mathbb{R}$.

$f(n)$ is $O(g(n))$ if $\exists\, \underset{\text{constant}}{c} > 0,\ c \in \mathbb{R} \wedge \underset{\text{constant}}{n_0} \in \mathbb{Z}, n \geq 1 \mid$

$f(n) \leq c \overset{\text{times}}{\times} g(n)\ \forall n \geq n_0$ independent from n

We need to find constants $c>0$ and integer $n_0 \geq 1$ st.

$\frac{1}{2n-1} \leq c\frac{1}{n}\ \forall n \geq n_0$ $\quad f(n) = \frac{1}{2n-1} \quad g(n) = \frac{1}{n}$

$\underline{\frac{1}{2n-1}}$ $\quad c\frac{1}{n} \geq \frac{1}{2n-1}$ always

$\forall n \geq n_0$

$\frac{n}{1} \leq c\frac{(2n-1)}{n}$

$4n^2 + 3n \leq cn^2$

$3n \leq cn^2 - 4n^2$

$3n \leq n^2(c-4)$

$3 \leq n(c-4)$ choose a c st this is true

$\frac{1}{2n-1} \leq c\frac{1}{n}$

$1 \leq \frac{c(2n-1)}{n} = \frac{2nc - c}{n} = \frac{2c}{\cancel{}} - \frac{c}{n}$

$1 \leq 2c - \frac{c}{n}$ choose c st this is true

Let $c = 1$

$1 \leq 2 - \frac{1}{n}$ is valid for all $n \geq 1$, so we choose $n_0 = 1$

$c = 1$, $n_0 = 1$, $\therefore \frac{1}{2n-1}$ is $O(\frac{1}{n})$

1. We need to find constants $c > 0$ and integer $n_0 \geq 1$ such that

$$\frac{1}{2n-1} \leq c\frac{1}{n} \quad \forall\ n \geq n_0$$

$$1 \leq \frac{c(2n-1)}{n} \quad \forall\ n \geq n_0$$

$$1 \leq \frac{2nc}{n} - \frac{c}{n} \quad \forall\ n \geq n_0$$

$1 \leq 2c - \frac{c}{n} \quad \forall\ n \geq n_0$, let $c = 1$, then

$1 \leq 2 - \frac{1}{n} \quad \forall\ n \geq n_0$. This holds true for all $n \geq 1$,

so let $n_0 = 1$.

$\frac{1}{2n-1} \leq c\frac{1}{n}$ holds for $c = 1, n_0 = 1$, $\therefore \frac{1}{2n-1}$ is $O\left(\frac{1}{n}\right)$.

2. $\cancel{\frac{1}{n} \notin O\left(\frac{1}{n^2}\right)}$ Let $f(n), g(n) \geq 0 \mid f(n) \in O(g(n)) \wedge g(n) \geq 1\ \forall n \geq 1$.

Show $f(n) + k$ is $O(g(n))$, $k > 0$ constant

We need to find constants $c > 0$ and integer $n_0 \geq 1$ such that

$$\underbrace{|f(n) + k|}_{>0} \leq \underbrace{c}_{>0}(\underbrace{g(n)}_{\geq 0}) \quad \forall\ n \geq n_0$$

$f(n) \leq c(g(n)) - k \quad \forall\ n \geq n_0$, let $c = 1$, then

$f(n) \leq g(n) - k \quad \forall\ n \geq n_0$, this holds only if $g(n) - k \geq 0$

so ~~we~~ need a constant c st $c(g(n)) - k \geq 0$, so $c(g(n)) \geq k$.

Let $c = k$, then

$$f(n) \leq k(g(n)) - k = k(g(n) - 1) \quad \forall\ n \geq n_0.$$

$f(n) \leq k(g(n) - 1)$ This holds $\forall\ n \geq 1$, ~~so~~ $g(n) \geq 1$,

so let $n_0 = 1$.

$f(n) \leq c(g(n)) - k$ holds for $c = k$, $n_0 = 1$, $\therefore f(n) + k$ is $O(g(n))$.

3. Prove $\frac{1}{n} \notin O(\frac{1}{n^2})$.

Proof by contradiction. Assume $\frac{1}{n}$ is $O(\frac{1}{n^2})$. If true, then there are constants $c>0$ and $n_0 \geq 1$ st. $\frac{1}{n} \leq c\frac{1}{n^2}$ $\forall n \geq n_0$.

$1 \leq c\frac{1}{n}$, $\forall n \geq n_0$, since $n>0$,

$n \leq c$. This is valid only for values of n that are at most c, so it is not always true for all $n \geq n_0$. Let $n \geq c+n_0$, then $n \leq c$ does not hold.

This is a contradiction. There are no constants $c>0$ and $n_0 \geq 1$ st. $\frac{1}{n} \leq c\frac{1}{n^2}$ $\forall n \geq n_0$, and $\frac{1}{n} \notin O(\frac{1}{n^2})$

4. i) Input: Array L stores $\mathbb{Z}$s in ↑ order

x

numVals = ~~4~~ 6

third = $\frac{6}{3}$ = 2

if x = L[2], terminates

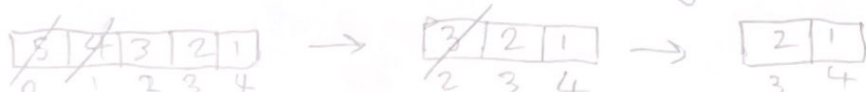
if x < L[2] or > L[2]

Let x = 7, L be the array

	5	4	3	2	1
i:	0	1	2	3	4

Then first = 0, last = 0

The condition to terminate is first > last and that never happens, because it will keep splitting the array by thirds until it reaches the last two elements in the array, and it loops infinitely. This is in the case that x is not in array L.



infinitely loops through last 2 elements looking for x.

X = 7

5. Algorithm does not output the correct answer. if $x = L[0]$ then $c \leftarrow 1$, if x is the first and only element, $c \leftarrow 1$. The algorithm will then continue to the for statement to give $c \leftarrow c+1 = 2$ and return 2. This is not the right output as it should be 1.

6. The time complexity is 1, because the while loop will always terminate after 1 loop.