Connie Ma
56047129 // p6z8
CPSC 221 202
7.2.17

THEORY ASSIGNMENT #1

pg 1

PART 1 — Induction

1 Let P(n) be statement that $1^3 + 2^3 + \dots + n^3 = \left(\frac{n(n+1)}{2}\right)^2$ for any +ve integer n.

a) $P(1)$: $1^3 = \left(\frac{(1)((1)+1)}{2}\right)^2$

$n = 1$

b) $P(1) = \left(\frac{(1)((1)+1)}{2}\right)^2 = \left(\frac{1(1+1)}{2}\right)^2 = \left(\frac{2}{2}\right)^2 = 1^2 = 1^3$ ✓

c) Inductive hypothesis

for some +ve int k, $P(k) \Rightarrow P(k+1)$ i.e., $1^3 + 2^3 + \dots + k^3 = \left(\frac{k(k+1)}{2}\right)^2 \Rightarrow 1^3 + 2^3 + \dots + k^3 + (k+1)^3 = \left(\frac{(k+1)((k+1)+1)}{2}\right)^2$

d) To prove the inductive step, prove that the LHS = RHS in the above statement from P(k+1).

e) i. Given $1^3 + 2^3 + \dots + k^3 = \left(\frac{k(k+1)}{2}\right)^2$ ★

ii. Substitute into $\underbrace{1^3 + 2^3 + \dots + k^3}_{★} + (k+1)^3$ ← expanded exp

iii. $\Rightarrow \left(\frac{k(k+1)}{2}\right)^2 + (k+1)^3 = \frac{k^2(k+1)^2}{4} + \frac{4(k+1)^3}{4}$ ← multiplied expression by $1 = \frac{4}{4}$

$= \frac{k^2(k+1)^2 + 4(k+1)^3}{4}$ add together

$= \frac{(k+1)^2(k^2 + 4k + 4)}{4}$ factored out $(k+1)^2$ from numerator

$= \frac{(k+1)^2(k+2)^2}{4}$ $(k+2)^2 = k^2 + 4k + 4$

$= \left(\frac{(k+1)(k+2)}{2}\right)^2$ factor out exponent

$= \left(\frac{(k+1)((k+1)+1)}{2}\right)^2 = \text{RHS}$ ✓ ▨

2 Prove the statement:

for all integers $n >= 1$, $1 + 6 + 11 + 16 + \dots + (5n-4) = \frac{n(5n-3)}{2}$

→ Base case $n = 1$: $1 = \frac{(1)(5(1)-3)}{2}$

$= \frac{1(5-3)}{2} = \frac{2}{2} = 1$ ✓

Connie Ma
56047129 // p6z8
CPSC 221 202
7.2.17

PART ①

THEORY ASSIGNMENT #1

② (cont'd) for ~~some~~ ⊕ve integer k,

Induction hypothesis: $P(k) \Rightarrow P(k+1)$ i.e. $1+6+11+16+\ldots+(5k-4) = \frac{k(5k-3)}{2}$

$$\Rightarrow 1+6+11+16+\ldots+(5k-4)+(5(k+1)-4) = \frac{(k+1)(5(k+1)-3)}{2}$$

Proof: i. Given $1+6+11+16+\ldots+(5k-4) = \frac{k(5k-3)}{2}$ ★

ii. Substitute into $\underbrace{1+6+11+16+\ldots+(5k-4)}_{★}+(5(k+1)-4)$

iii. $\Rightarrow \frac{k(5k-3)}{2} + 5((k+1)-4) = \frac{5k^2-3k}{2} + 5k+5-4$ expanded expressions

$= \frac{5k^2-3k}{2} + \frac{10k+10-8}{2}$ multiplied by $1 = \frac{2}{2}$

$= \frac{5k^2+7k+2}{2}$ added both expressions together

expand

$= \frac{5k^2+5k+2k+2}{2}$

$= \frac{5k(k+1)+2(k+1)}{2}$ factored out (k+1)

$= \frac{(k+1)(5k+2)}{2}$

add 0 = +3−3 to expression

$= \frac{(k+1)(5k+2+3-3)}{2}$

$= \frac{(k+1)(5k+5-3)}{2} = \frac{(k+1)(5(k+1)-3)}{2}$ = RHS ✓ ▨

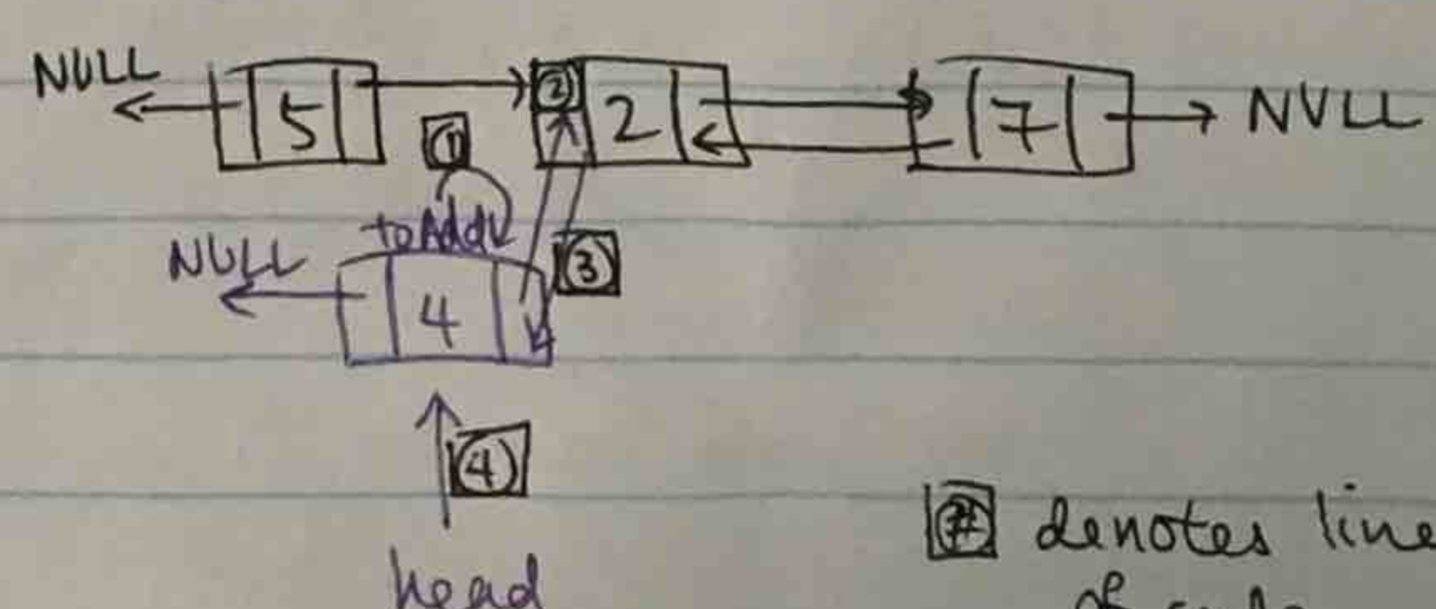
PART ② — POINTERS

doubly-linked list

```
class Node {
  public:
    Node (int v) {data = v;}
    int data;
    Node * prev;
    Node * next;
};
```

③ Diagram after following code has executed

```
[1] Node * toAdd = new Node (4);
[2] toAdd -> next = head -> next;
[3] head -> next -> prev = toAdd;
[4] head = toAdd;
```



[#] denotes line of code

Connie Ma
56047129 // p6z8
CPSC 221 202
7.2.17

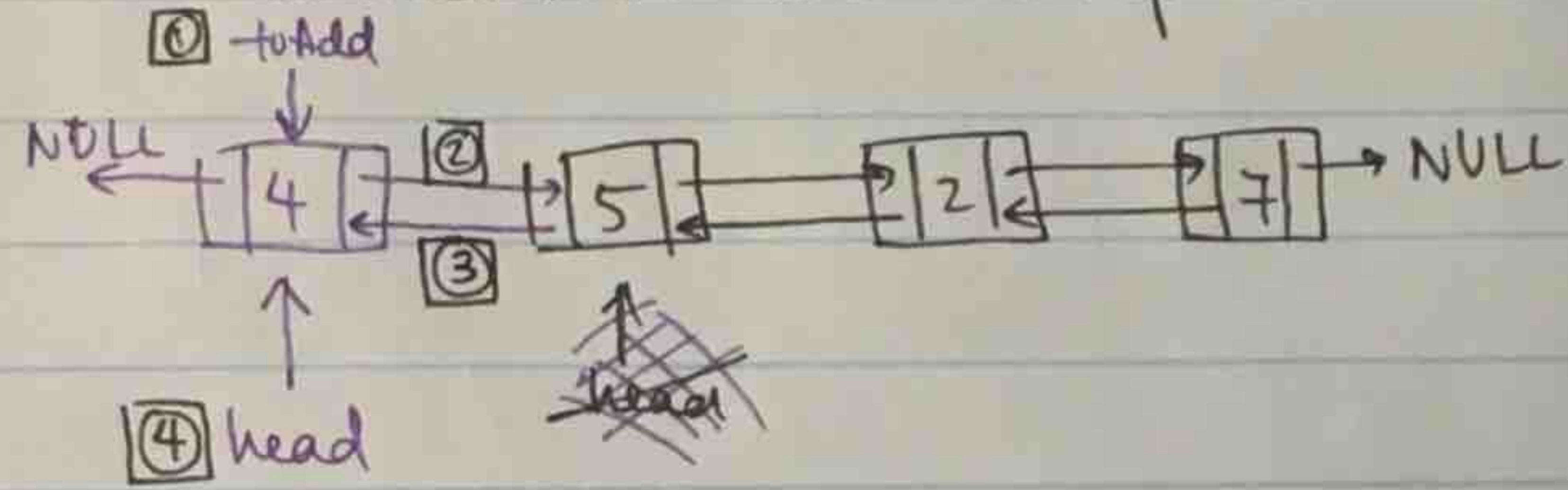
THEORY ASSIGNMENT #1

PART 2

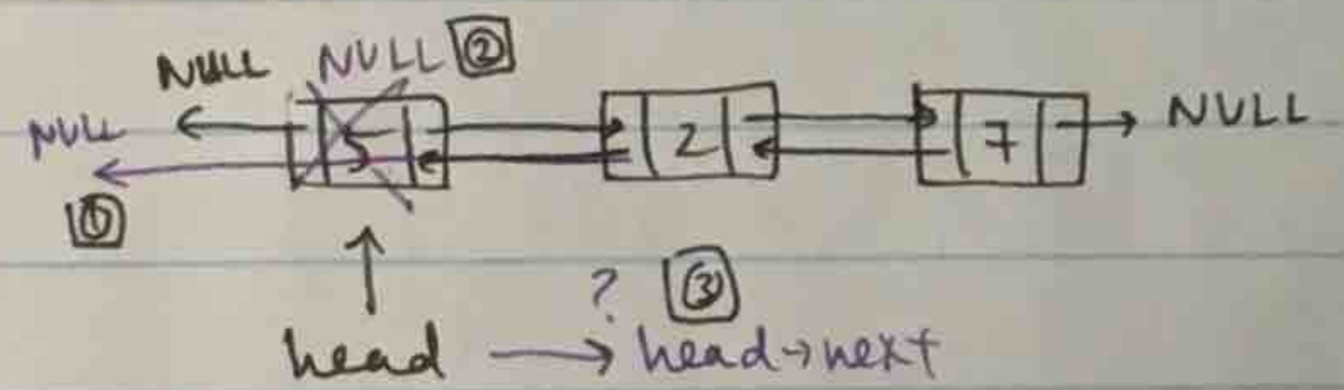
4. The code in (3) is incorrect.
Correct code is as follows:

```
//trying to insert a node at the front (head) of the list
[1] Node *toAdd = new Node(4);
[2] toAdd -> next = head;
[3] head -> prev = toAdd;
[4] head = toAdd;
```

5. Diagram of the state of memory when the corrected code completes:



6. Using the original diagram, draw memory diagram after following lines of code executed:



//// = original
//// = after code execution

```
[1] head -> next -> prev = head -> prev;
[2] delete head;
[3] head = head -> next;
```

7. Explain why code from 6 is not safe.

Just by looking at above diagram you can tell that we will lose track of the head ptr data by invoking delete head, which is not safe because the program will segfault at line 3 of the code because it is trying to find information from memory that has already been deallocated.

8. Fix the code so that it properly removes the node from front of the list.

```
//trying to remove a node from front of the list
Node* temp = head;
head -> next -> prev = temp;      <- or can just assign to NULL
head = head -> next;   <- this way we can keep track of head
delete temp;   <- then safely deallocate memory
```

Connie Ma
56047129 // p6z8
CPSC 221 202
7.2.12

THEORY ASSIGNMENT #1

PART 3 — Algorithm Analysis

9 Find smallest int x s.t. $f(n)$ is $O(n^x)$ & ~~the~~ constants c & n_0 to satisfy Big-O, for each of the following:

a) $f(n) = n^2 + 25n - 4$

$T(n) = n^2 + 25n - \overset{\text{constant}}{\cancel{4}} \le n^2 + 25n^2 \le 26n^2$ (arbitrarily satifies)

$\therefore T(n) \in O(n^2) \Rightarrow \boxed{x = 2}$ w/ $\boxed{c = 26,\ n_0 = 1}$

$26n^2 = cn^2$

b) $f(n) = 5n^3 + 3n^2 \log n$

$T(n) = 5n^3 + 3n^2 \log n \le 5n^3 + 3n^3 \le 8n^3$

$\therefore T(n) \in O(n^3) \Rightarrow \boxed{x = 3}$ w/ $\boxed{c = 8,\ n_0 = 1}$

$8n^3 = cn^3$

c) $f(n) = \dfrac{n^4 + n^2 + 1}{n^3 + 1}$

$T(n) = \dfrac{n^4 + n^2 + 1}{n^3 + 1} \le \dfrac{n^4 + n^2 + 1}{n^3} \le n + \dfrac{1}{n} + \dfrac{1}{n^3} \le 3n$

$\le n + n + n$

$\therefore T(n) \in O(n) \Rightarrow \boxed{x = 1}$ w/ $\boxed{c = 3,\ n_0 = 1}$

$3n = cn$

10 Explain based on expected runtime diff. b/t array & linked-list for the following

a) find (int index);

$\boxed{\text{array}}$-based implementation worst-case estimate is $\boxed{O(1)}$, since index is supplied

$\boxed{\text{linked-list}}$ worst-case is $\boxed{O(n)}$, since you must ~~still~~ iterate thru the list if index is at the very end

∴ The array-implementation is expected to run faster than LL-implementation based on Big O estimation.

b) remove (int index);

→ could be $\boxed{O(\log n)}$ w/ binary search of sorted arr

$\boxed{\text{array}}$-based implementation worst-case is $\boxed{O(n)}$, since you must still iterate thru entire list to maintain adjacent-ordering

$\boxed{\text{linked-list}}$ worst-case is also $\boxed{O(n)}$, since you must ~~check~~ traverse the entire list to find index as before in worst case (although maintaining adjacent order is only O(1)).

∴ The array-implementation and LL-implementation have expected runtime worst cases that are the same.

→ If binary search implemented for array, then array would be faster by O(log n).

Connie Ma
56047129 // p6z8
CPSC 221 202
7.2.17

THEORY ASSIGNMENT #1

PART 3 (cont'd)

11 Give tight <u>worst-case bounds</u> for the time complexity of each of the following pieces of code. How much time does each loop take? Calc $T(n)$ & express its order of growth using Θ-notation.

a) // reverses the values in array arr

```
void reverse (int arr[], int size) {
    for (int i=0; i<size; i++) {        // O(1)  O(n)  O(n)  = 2n+2
        ① int temp = arr[0];            // — O(n)   2n+2 (see above)
        for (int j=0; j<size-i-1; j++) {   // O(1)  (n-1)(n-2)...  O(n)
            ② arr[j] = arr[j+1];        // — 2·O(size)
        }
        ③ arr[size-i-1] = temp;         // — 2·O(size)
    }
}
```

i. $T(n) \Rightarrow$ line ① & ③ $\times\ 2n+2 = 4n+\cancel{2}$ const

line ② $\times \dfrac{n(n+1)}{2}\ \cancel{-1}$ const

summation because j goes to n-i-1

$\dfrac{n(n+1)}{2} - 1 + 4n+2 \leftarrow T(n)$

$T(n) = \boxed{\dfrac{n(n+1)}{2} + 4n + 1}$

ii. Big-O : $T(n) = \dfrac{n(n+1)}{2} + 4n + \cancel{1}$ (const) $\le \dfrac{n^2}{2} + \dfrac{n^2}{2} + 4n^2$ (const)

$\le 5n^2$ (5 → constant) $\quad \therefore \boxed{T(n) \in O(n^2)}$

iii. Big-Ω : $T(n) = \dfrac{n^2+n}{2} + 4n + 1 \ge \dfrac{n^2}{4}$ $\quad (n_0 \ge 2)$

$\hookrightarrow \dfrac{4+2}{2} + 4(2) + 1 \ge \dfrac{4}{4}$ ✓

$\therefore \boxed{T(n) \in \Omega(n^2)}$

Big-O = Big-Ω $\Rightarrow \boxed{T(n) \in \Theta(n^2)}$

Connie Ma
[illegible]6047129 // p6z8
CPSC 221 202
7.2.17

THEORY ASSIGNMENT #1

PART 3

11 (cont'd)

b) // searches a sorted array for the value n
// returns index of n, or -1 if array does not contain n

```
int search (int arr[], int size, int n){
    int min = 0, max = size-1;
    int mid;  ——> O(1) each = 3.
    while (min <= max) {
      ① mid = (min + ②/③ max) /2 ;
      ④ if (arr[mid] < n) { min = ⑤/⑥ mid + 1; }
      ⑦ else if ( arr[mid] > n) { max = ⑧/⑨ mid - 1; }
        else { return ⑩ mid; }
    }
    return -1;  — O(1)
}
```

binary search = $O(\log n)$

LOC × log n = 10 log n

i. → $T(n) = \boxed{4 + 10\log n}$ (while loop $T(n)$)

ii. Big-O : $T(n) = \cancel{4} + \underbrace{\cancel{10}\log n}_{\text{constant}} \leq \log n$

$\therefore \boxed{T(n) \in O(\log n)}$

iii. Big-Ω : $T(n) = 4 + \cancel{10\log n} \geq 4 \geq 1$

$\therefore \boxed{T(n) \in \Omega(1)}$

Big-O > Big-Ω $\therefore \boxed{T(n) \in \Theta(\log n)}$