UIU QUESTION BANK

MID-TERM QUESTION SOLUTIONS

DATA STRUCTURE AND ALGORITHM I

CSE 2215

SOLUTION BY

NURUL ALAM ADOR

UPDATED TILL ***FALL 2023***

nurulalamador.github.io/**UIUQuestionBank**

Index

Trimester	Page
Fall 2023	3
Summer 2023	11
Spring 2023	21
Fall 2022	30
Summer 2022	37

1. a) How does the **Descending Order Insertion Sort** work on the following data?

y p z x r s

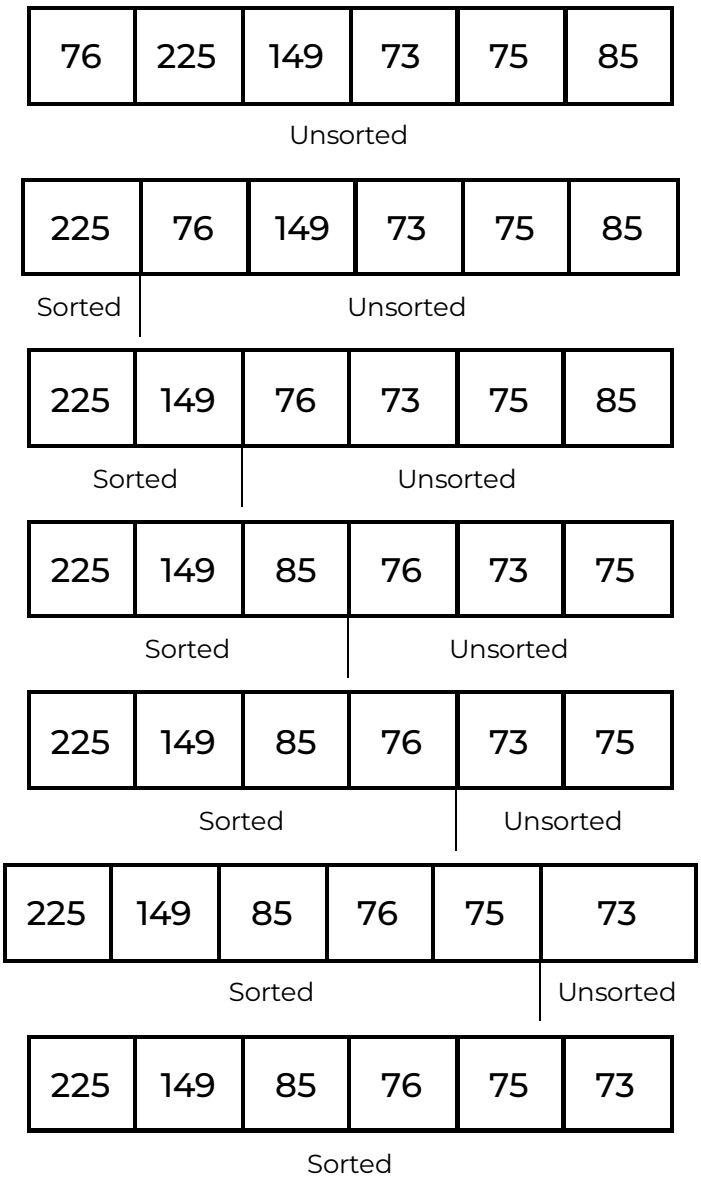
Here, x=last two digits of your student id+3, y=x+3, z=x+y, p=y+z, r=x+2, s=y+9

Solution:

Here, x = 70+3 = 73 p = 76+149 = 225
y = 73+3 = 76 r = 73+2 = 75
z = 73+76 = 149 s = 76+9 = 85

76	225	149	73	75	85
y	p	z	x	r	s

Descending Insertion Sort:



∴ Final sorted data is: 225, 149, 85, 76, 75, 73.

1. b) Find a recurrence for time using the recursive **Merge Sort** and solve the recurrence.

Solution:

In Merge Sort, we divide the array into two nearly equal halves and solve and merge them recursively.

So, we have,

$$T\left(\frac{n_L}{2}\right) + T\left(\frac{n_R}{2}\right) = 2T(\frac{n}{2})$$

Here, n_L = Left Half
n_R = Right Half
$n_L = n_R$ or $n_L \approx n_R$

Finally, we merge these two sub arrays using merge procedure which takes O(n) time,

$$T(n) = 2T\left(\frac{n}{2}\right) + O(n)$$
$$\therefore T(n) = 2T\left(\frac{n}{2}\right) + n$$

This is our recurrence relation for recursive Merge Sort.

Now, solving the recurrence:

$$T(n) = 2T\left(\frac{n}{2}\right) + n$$
or, $T(n) = 2[2T\left(\frac{n}{2} \times \frac{1}{2}\right) + \frac{n}{2}] + n$
or, $T(n) = 4T\left(\frac{n}{4}\right) + 2n$ (i)
or, $T(n) = 4[2T\left(\frac{n}{4} \times \frac{1}{2}\right) + \frac{n}{4}] + 2n$
or, $T(n) = 8T\left(\frac{n}{8}\right) + 3n$ (ii)

Considering equation (i) and (ii),

$$T(n) = 4T\left(\frac{n}{4}\right) + 2n = 2^2T\left(\frac{n}{2^2}\right) + 2n = 2^kT\left(\frac{n}{2^k}\right) + kn$$

$$T(n) = 8T\left(\frac{n}{8}\right) + 3n = 2^3T\left(\frac{n}{2^3}\right) + 3n = 2^kT\left(\frac{n}{2^k}\right) + kn$$

∴ General equation for recurrence is: $T(n) = 2^kT\left(\frac{n}{2^k}\right) + kn$

Let, $\frac{n}{2^k} = 1$
or, $n = 2^k$
or, $\log n = k$

∴ New general equation is,

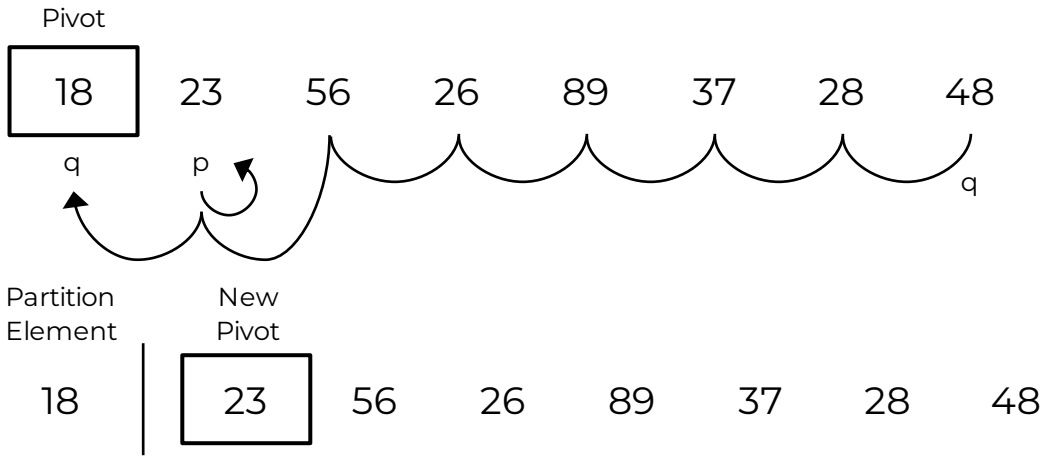
$$T(n) = nT(1) + logn \times n$$
or, $T(n) = n + nlogn$
or, $T(n) = O(nlogn)$

∴ The solution of this recurrence is O(nlogn).

1. c) How many element comparisons are needed for the following instance of the **Ascending Order Quick Sort** to find the first portioning element?

18 23 56 26 89 37 28 48

Solution:



Here, p need element comparison 1 time and q need element comparison 8 times for finding first partition.

∴ Total 1+8 = 9 element comparisons are needed to find partition element.

2. a) Find the memory location of A[70][80] if loc(A[15][20])=x+1400, where x=last four digits of your student ID. Assume column-wise memory is allocated in double type array A[90][100], where each double data is 8 bytes.

Solution:

Here, $l_1 = 0$ $l_2 = 0$

$u_1 = 90 - 1 = 89$ $u_2 = 100 - 1 = 99$

$L = 8$ $x = 0170 = 170$

$b = ?$

We know,

For column-wise memory allocated 2D array,

$$loc(A[i][j]) = b + (j - l_2) \times (u_1 - l_1 + 1) \times L + (i - l_1) \times L$$

Given,

$$loc(A[15][20]) = x + 1400$$

or, $b + (20 - 0) \times (89 - 0 + 1) \times 8 + (15 - 0) \times 8 = 170 + 1400$

or, $b + 14400 + 120 = 1570$

or, $b = 1570 - 14520$

∴ $b = -12950$

Now,

$$loc(A[70][80]) = -12950 + (80 - 0) \times (99 - 0 + 1) \times 8 + (70 - 0) \times 8$$

∴ $loc(A[70][80]) = 51610$

∴ Memory location of A[70][80] is 51610.

2. b) If $f(n) = kn^2 - 3n + 5$, prove that $f(n) = \Theta(n^2)$. Here, k=last digit of your student id+4

Solution:

Here, $k = 0 + 4 = 4$

$f(n) = 4n^2 - 3n + 5$

$g(n) = n^2$

Proving $f(n) \leq C_1 \cdot g(n)$ and $C_2 \cdot g(n) \leq f(n)$ is enough to prove $f(n) = \Theta(n^2)$.

Let, $C_1 = 100$

Now, $f(n) \leq C_1 \cdot g(n)$

or, $4n^2 - 3n + 5 \leq 100n^2$

Here, $f(n) \leq C_1 \cdot g(n)$ is true for all value of $n_0 \geq 1$.

Let, $C_2 = 1$

Now, $C_2 \cdot g(n) \leq f(n)$

or, $1n^2 \leq 4n^2 - 3n + 5$

Here, $C_2 \cdot g(n) \leq f(n)$ is true for all value of $n_0 \geq 1$.

$\therefore f(n) \leq C_1 \cdot g(n)$ and $C_2 \cdot g(n) \leq f(n)$ is true for all value of $n \geq 1$ over $C_1 = 100, C_2 = 1$.

$\therefore f(n) = \Theta(n^2)$. (Proved)

2. c) How does the **Binary Search Algorithm** work on the following data?

Input Data: t r p z y x
Search Key = y

Here, x=last two digits of your student ID, y=x+4, z=x+y, p=y+z, r=z+p, and t=p+r
Also find the total element comparisons for the given instance of the Binary Search.

Solution:

Here, x = 70 z = 70+74 = 144 r = 144+218 = 362
y = 70+4 = 74 p = 74+144 = 218 t = 218+362 = 580

t	r	p	z	y	X
580	362	218	144	74	70
0	1	2	3	4	5

Search Key = y = 74

Comparison 1: Lower Limit, $l = 0$
Higher Limit, $h = 5$
Mid Value, $m = \frac{l+h}{2} = \frac{0+5}{2} = 2.5 \approx 2$

Here, $Data[2] = 218$, $218 > 74$

$\therefore$ New Lower Limit, $l = m + 1 = 2 + 1 = 3$

Comparison 2: Lower Limit, $l = 3$

Higher Limit, $h = 5$

Mid Value, $m = \frac{l+h}{2} = \frac{3+5}{2} = 4$

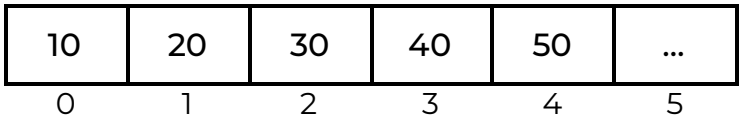
Here, $Data[4] = 74, \quad 74 == 74$

∴ 'y' founded in Data

∴ Total 2 element comparisons needed for the given instance of the Binary Search.

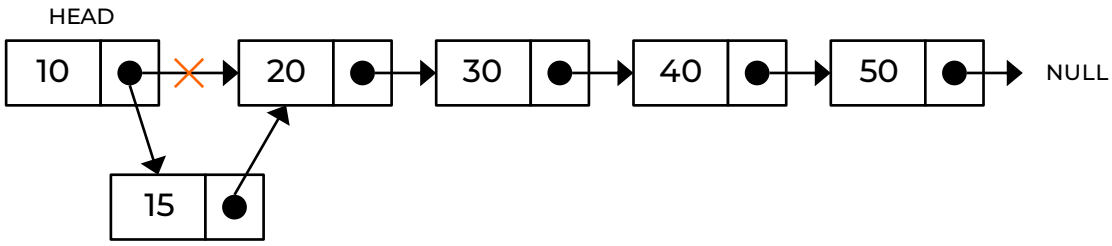
3. a) An array contains 10, 20, 30, 40, 50. Now we want to insert 15 in-between 10 and 20. Remember that it will maintain the ascendency after insertion. What is the difficulty for this insertion? How this problem can be resolved by a linked list easily?

Solution:



If we want to insert 15 between 10 and 20 in the array, we need to shift 20, 30, 40, and 50 to one index right. Beside this difficulty, if our array size is declared as 5, we can neither shift these data to one index right nor insert the 15 between 10 and 20.

But we can resolve these difficulties in linked list easily. We just need to make new node of 15 and connect 10's node next to 15's node and 15's node next to 20's node.

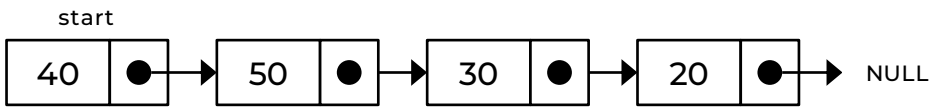


3. b) Suppose a linear linked list headed with "start" contains four nodes whose data values are 40, 50, 30, 20, respectively. Show the following operations.

i) Draw a diagram for the linear linked list.
ii) Find a name for each of the nodes with respect to "start" that contain 40, 50, 30, 20, respectively?
iii) Write statements to represent 40, 50, 30, 20, respectively.
iv) Write a statement to set NULL at the end of the linked list.
v) Write statements to delete the node that contains 30.
vi) Write statements to insert a node "temp" in-between 50 and 20 that contains 28.

Solution:

i) The diagram has been drawn below:



ii) Name for each of the nodes with respect to "start" has been written below:

```
Node first = start;
Node second = start->next;
Node third = start->next->next;
Node fourth = start->next->next->next;
```

iii) Statements to represent 40, 50, 30, 20, respectively are written below.

40:
```
start->value;
```

50:
```
start->next->value;
```

30:
```
start->next->next->value;
```

20:
```
start->next->next->next->value;
```

iv) Statement to set NULL at the end of the linked list are written below:

```
start->next->next->next->next = NULL;
```

v) Statements to delete the node that contains 30 are written below:

```
Node* temp = start->next->next;
start->next->next = temp->next;

delete temp;
```

vi) Statements to insert a node "temp" in-between 50 and 20 are written below:

```
Node* temp = new Node();
temp->value = 28;
temp->next = start->next->next;

start->next->next = temp;
```

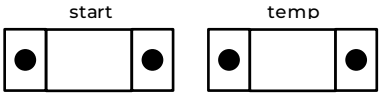
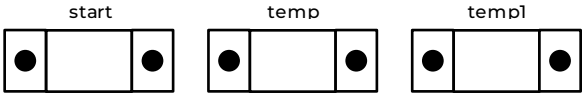
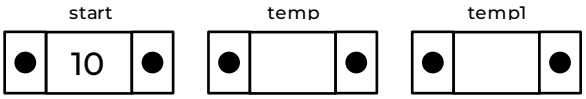
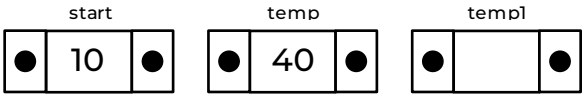
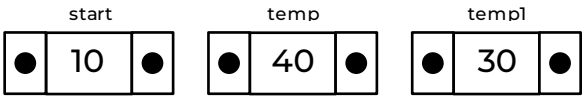
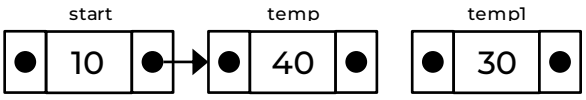
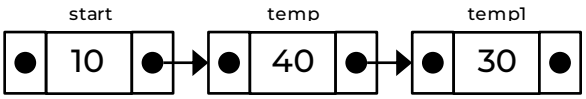
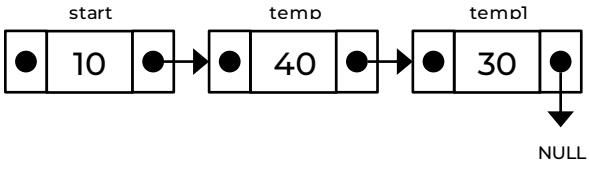
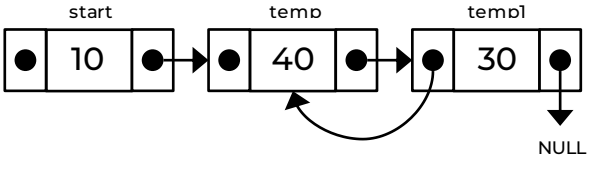
4. a) Show the effect of each of the statements given in the following code segment. Assume, each of the nodes in the doubly linked list has fields' **data, next and back**, where **data** is of integer type, and **next** and **back** will contain the addresses of the next and previous nodes, respectively.

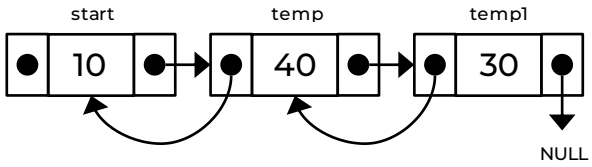
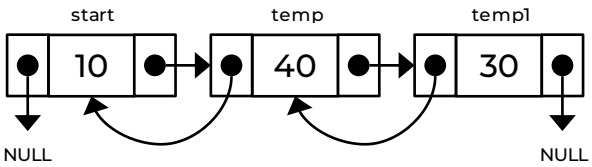
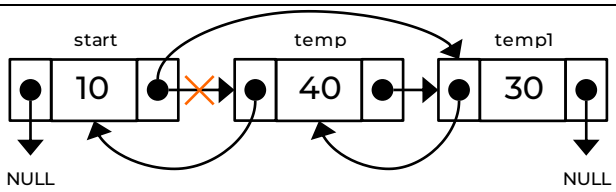
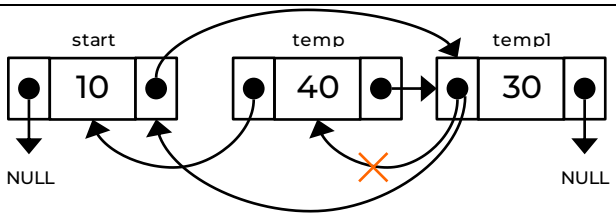
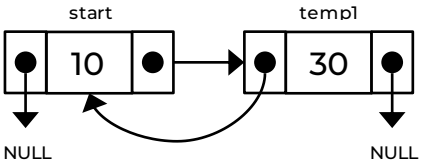
```
start=(node*)malloc(sizeof(node));
temp=(node*)malloc(sizeof(node));
temp1=(node*)malloc(sizeof(node));

start->data=10;
temp->data=40;
temp1->data=30;
```

```
start->next=temp;
temp->next=temp1;
temp1->next=NULL;

temp1->back=temp;
temp->back=start;
start->back=NULL;

temp->back->next=temp->next;
temp->next->back=temp->back;
free(temp);
```

Solution:

Effect of each of the given statements are showed below:

Statements	Effect
`start=(node*)malloc(sizeof(node));`	
`temp=(node*)malloc(sizeof(node));`	
`temp1=(node*)malloc(sizeof(node));`	
`start->data=10;`	
`temp->data=40;`	
`temp1->data=30;`	
`start->next=temp;`	
`temp->next=temp1;`	
`temp1->next=NULL;`	
`temp1->back=temp;`	

`temp->back=start;`	
`start->back=NULL;`	
`temp->back->next=temp->next;`	
`temp->next->back=temp->back;`	
`free(temp);`	

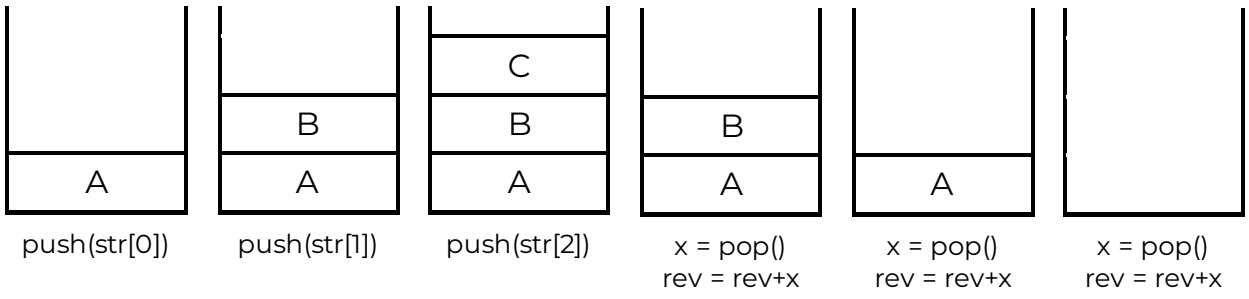
4. b) How can you reverse a string using a STACK implemented by an array? Show push() and pop() operations in this regard.

Solution:

To reverse a string using a Stack, at first, we need to push all characters of that string to the Stack from first to last. Then we should make separate empty string and pop() all character from Stack and add these character in that empty string one by one.

All push() and pop() operations for reversing **`string str = "ABC"`** are shown below:

Let, **`string rev = ""`**



After these operations,
Our string **`rev`** will became **`"CBA"`**, which is reverse of **`str = "ABC".`**

Summer 2023

1. a) Demonstrate how **Descending Order Merge Sort** will work on the following data?

y p z x r s

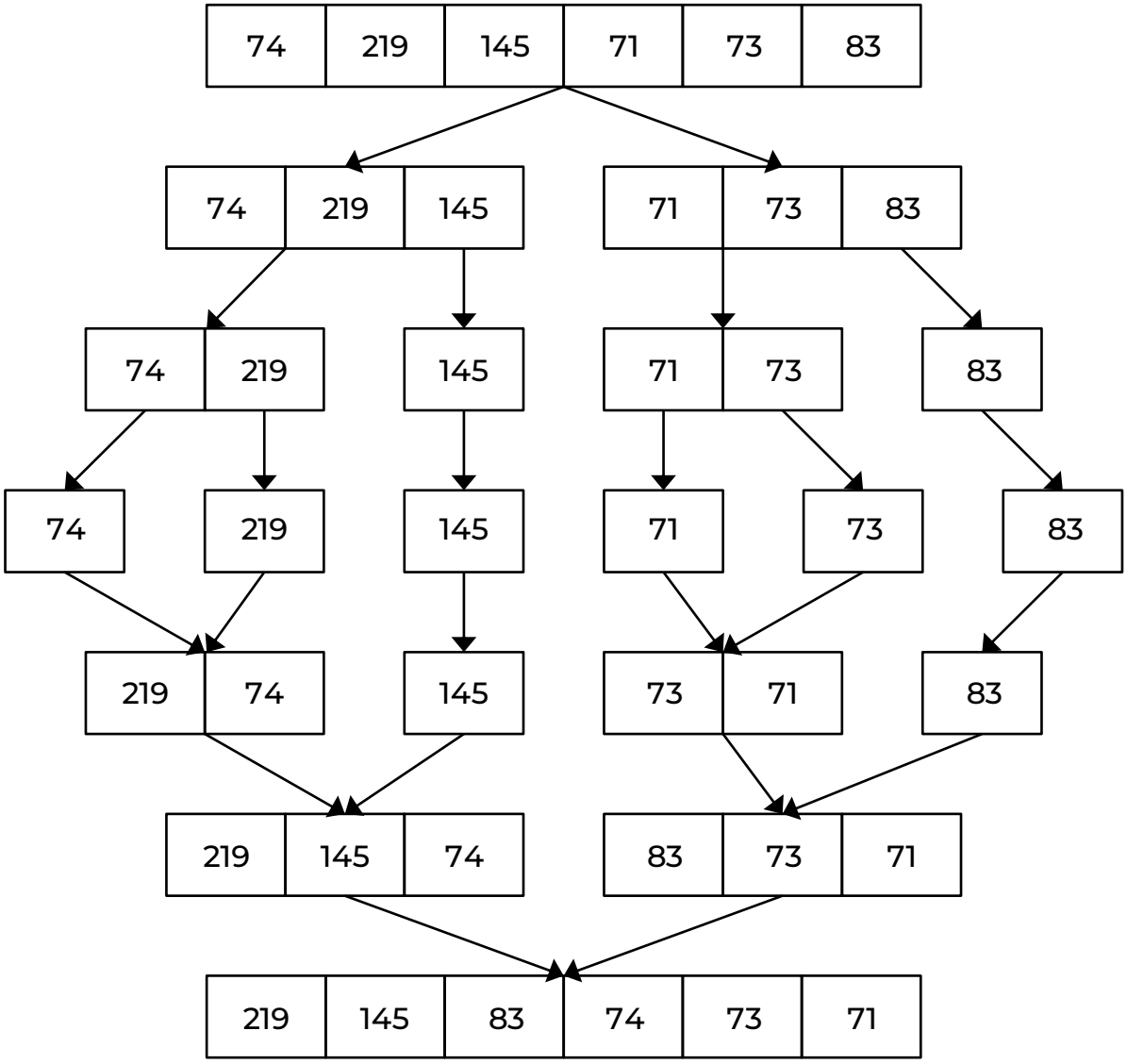
Here, x=last two digits of your student id+1, y=x+3, z=x+y, p=y+z, r=x+2, s=y+9
You must show the entire sorting procedure using a recursion tree.

Solution:

Here, x = 70+1 = 71
y = 71+3 = 74
z = 71+74 = 145
p = 74+145 = 219
r = 71+2 = 73
s = 74+9 = 83

74	219	145	71	73	83
y	p	z	x	r	s

Descending Merge Sort:



∴ Final sorted data is: 219, 145, 83, 74, 73, 71.

1. b) Discuss the time complexity of the following algorithm.

```
sum=0;
for(i=2; i<=n; i++){
    for(j=2; j<=i; j++){
        sum=sum+i+j;
    }
}
printf("%d", sum);
```

Solution:

Statements	Times
`sum=0;`	1
`for(i=2; i<=n; i++){`	n
`for(j=2; j<=i; j++){`	T
`sum=sum+i+j;`	T-1
`}`	0
`}`	0
`printf("%d", sum);`	1

Here, $T = 1 + 2 + 3 + \cdots + n = \frac{n^2 + n}{2}$

∴ Time complexity, $T(n) = 1 + n + \frac{n^2 + n}{2} + \frac{n^2 + n}{2} - 1 + 1$

$= 1 + n + n^2 + n$

$= O(n^2)$

∴ Time complexity of this algorithm is $O(n^2)$.

2. a) What will be the index returned by the partition function PARTITION (A,1,8) for the following array elements in **Ascending Order Quicksort**?

18 23 56 26 89 37 28 48

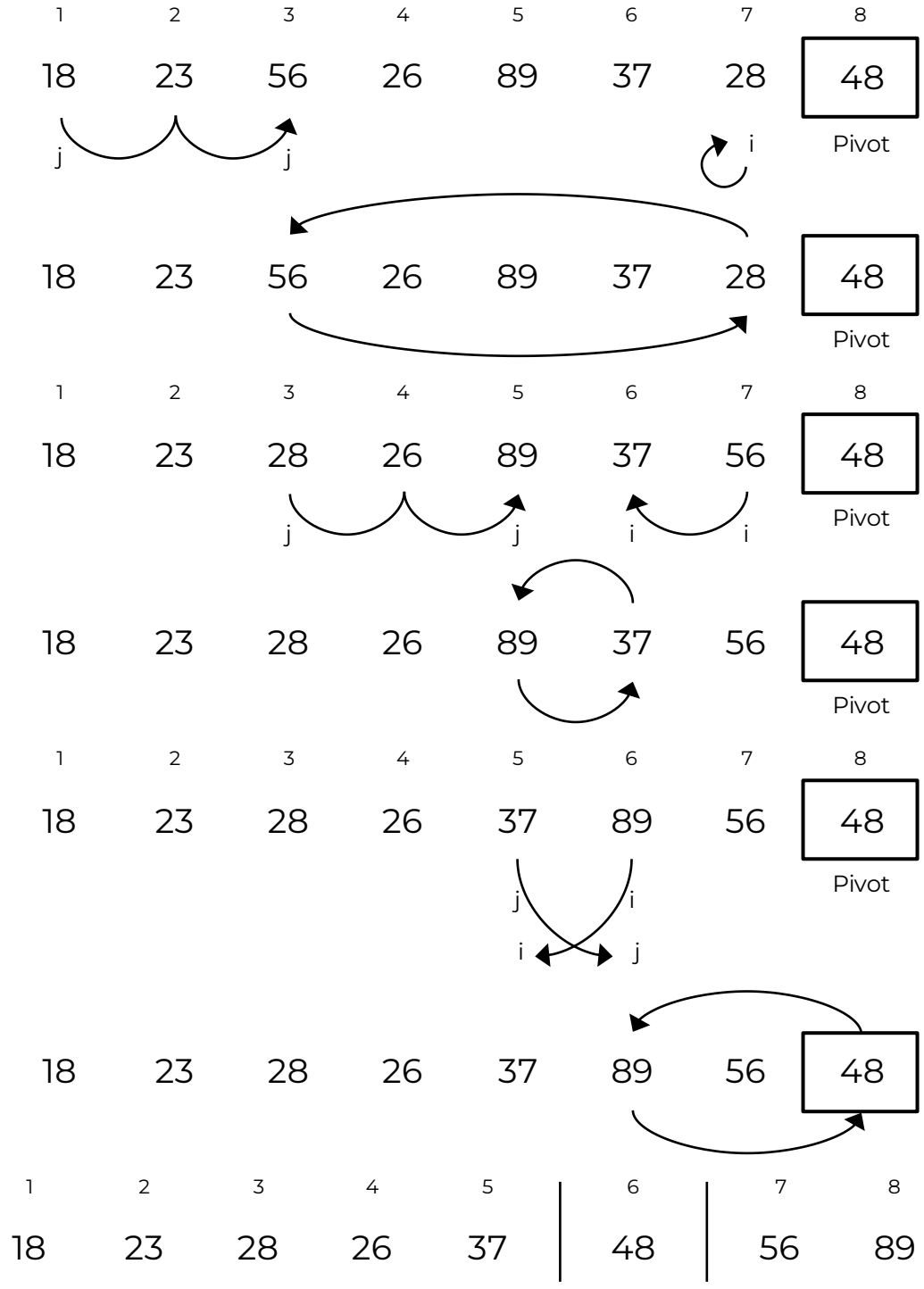
Also, show the condition of the given array once the partition function is executed

Partition Function:

```
PARTITION(A,p,r)
   1 x ← A[r]
   2 i ← p-1
   3 for j ← p to r-1
   4     do if A[j]<=x
   5           then i ← i+1
   6 exchange A[i] ↔ A[j]
   7 exchange A[i+1] ↔ A[r]
   8 return i+1
```

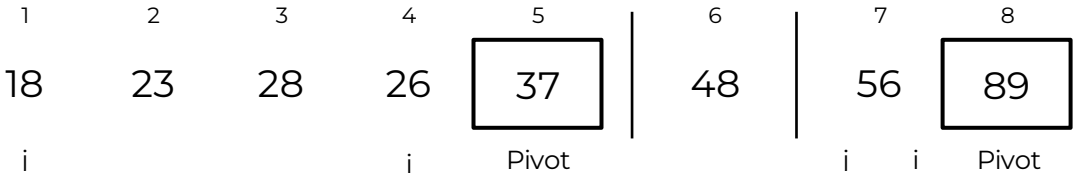
Solution:

According to given code, right-most element is pivot.



∴ PARTITION (A,1,8) function will be return index 6.

Condition of array after partition function executed:



2. b) Consider the following array of 5 elements

Array: 30, 10, 40, 20, 15

I. How many times will the condition of the while loop in the Descending Order Insertion Sort Algorithm be executed for the following data?
II. How many times will the while loop be executed if the above array was already sorted in ascending order?
III. How many times will the while loop be executed if the above array was already sorted in Descending order?

Solution:

I. For this case, condition of while loop will execute 4 times for true condition and 5-1 = 4 conditions for false condition.
∴ Total 4+4 = 8 times condition of while will be loop executed.

II. For this case, condition of while loop will execute 10 times for true condition and 5-1 = 4 conditions for false condition.
∴ Total 10+4 = 14 times condition of while will be loop executed.

III. For this case, condition of while loop will execute 0 times for true condition and 5-1 = 4 conditions for false condition.
∴ Total 0+4 = 4 times condition of while will be loop executed.

2. c) Consider the following array declaration in C.

```
double A[80][90];
```

Find the memory location of `A[60][70]` if `loc(A[15][20])=x+1200`, where x=last four digits of your student ID.
Assume column-wise memory is allocated for the array, where each data of type double is 8 bytes.

Solution:

Here, $l_1 = 0$ $\quad l_2 = 0$
$u_1 = 80 - 1 = 79$ $\quad u_2 = 90 - 1 = 89$
$L = 8$ $\quad x = 0170 = 170$
$b = ?$

We know,
For column-wise memory allocated 2D array,

$$loc(A[i][j]) = b + (j - l_2) \times (u_1 - l_1 + 1) \times L + (i - l_1) \times L$$

Given,

$$loc(A[15][20]) = x + 1200$$

or, $b + (20 - 0) \times (79 - 0 + 1) \times 8 + (15 - 0) \times 8 = 170 + 1200$

or, $b + 12800 + 120 = 1370$

or, $b = 1370 - 12920$

∴ $b = -11550$

Now,

$$loc(A[60][70]) = -11550 + (70 - 0) \times (79 - 0 + 1) \times 8 + (60 - 0) \times 8$$

or, $loc(A[60][70]) = -11550 + 44800 + 480$

∴ $loc(A[60][70]) = 33730$

∴ Memory location of A[60][70] is 33730.

3. a) Show the step-by-step simulation for searching character 'w' using a **binary search algorithm** in the following sorted array of characters. For each step, you must mention the high, low and mid values indices.

b, d, f, g, h, k, l, q, r, s, w, y

Solution:

Simulation for Binary Search Algorithm are showed below:

b	d	f	g	h	k	l	q	r	s	w	y
0	1	2	3	4	5	6	7	8	9	10	11

Search Key = w

Comparison 1:
Lower Limit, $l = 0$
Higher Limit, $h = 11$
Mid Value, $m = \frac{l+h}{2} = \frac{0+11}{2} = 5.5 \approx 5$
Here, $Data[5] = k$, k < w *[ASCII Order]*
∴ New Lower Limit, $l = m + 1 = 5 + 1 = 6$

Comparison 2:
Lower Limit, $l = 6$
Higher Limit, $h = 11$
Mid Value, $m = \frac{l+h}{2} = \frac{6+11}{2} = 8.5 \approx 8$
Here, $Data[8] = r$, r < w *[ASCII Order]*
∴ New Lower Limit, $l = m + 1 = 8 + 1 = 9$

Comparison 3:
Lower Limit, $l = 9$
Higher Limit, $h = 11$
Mid Value, $m = \frac{l+h}{2} = \frac{9+11}{2} = 10$
Here, $Data[10] = w$, w == w
∴ 'w' founded in Data

∴ 'w' has been founded in given data and total 3 element comparison is needed.

3. b) Prove that running time of $f(n) = 3n^3 + 2n^2 + 5n + 1$ is O(n^3)

Solution:

Here, $f(n) = 3n^3 + 2n^2 + 5n + 1$
$g(n) = n^3$

Proving $f(n) \leq C \cdot g(n)$ is enough to prove running time of $f(n)$ is O(n^3).

Let, $C = 1000$

Now, $f(n) \leq C \cdot g(n)$

or, $3n^3 + 2n^2 + 5n + 1 \leq 1000n^3$

Here, $f(n) \leq C \cdot g(n)$ is true for all value of $n_0 \geq 1$.

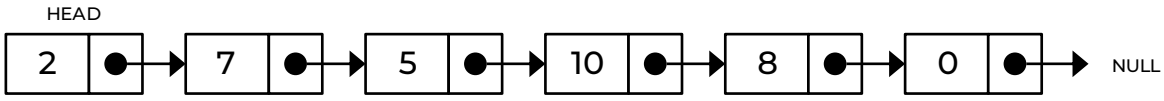
$\therefore f(n) \leq C_1 \cdot g(n)$ is true for all value of $n \geq 1$ over $C = 1000$.

$\therefore$ Running time of $f(n) = 3n^3 + 2n^2 + 5n + 1$ is O(n^3). (Proved)

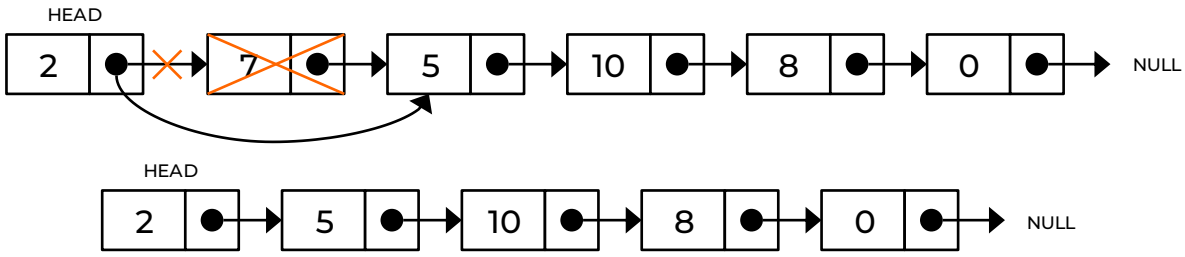
3. c) Given a linear/single linked list containing six nodes whose data values are 2, 7, 5, 10, 8, and 0 respectively. Show the step-by-step simulation and pseudocode for the following operations.

I. Delete the element at the second position of the linked list.
II. Find the minimum of the linked list.
III. Insert the square of the first element into the end of the linked list.

Solution:



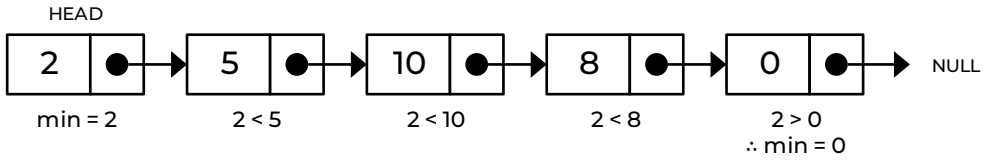
I. Simulation:



Pseudocode:

```
Node* temp = head->next;
head->next = temp->next
delete temp;
```

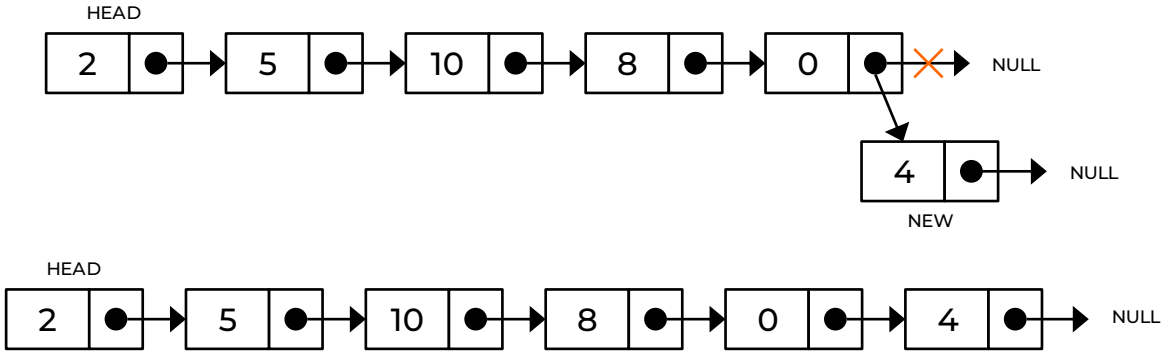
II. Simulation:



Pseudocode:

```
int min = head->value;
Node* temp = head;
while(temp != NULL) {
    if(min > temp->value)
        min = temp->value;
    temp = temp->next;
}
```

III. Simulation:



Pseudocode:

```
Node* new = new Node();
new->value = Math.pow(head->value, 2);
Node* temp = head;
while(temp->next != NULL) {
    temp = temp->next;
}
temp->next = new;
```

4. a) Consider the following code snippet. You will have to modify the code so that you can perform the "deleteLast" operation in O(1) time. After the necessary modifications, complete the implementation of both the" deleteLast" and "insertAtMiddle" functions.

Note: The "deleteLast" function will delete the last element of the linked list whilst the "insertAtMiddle" function will insert an element in the middle of the linked list.

```
struct node {
    int value;
    struct node *next;
    struct node *prev;
};
struct node *head;
void deleteLast(){
}
void insertAtMiddle(int value){
}
int main() {
    return 0;
}
```

Solution:

The modified code has been written below:

```
struct node {
    int value;
    struct node *next;
    struct node *prev;
};
struct node *head;
struct node *last;
```

```
void deleteLast(){
    struct node *temp = last;
    last = last->prev;
    last->next = NULL;
    delete temp;
}
void insertAtMiddle(int value){
    struct node *start = head;
    struct node *end = last;

    while(start != end) {
        start = start->next;
        if(start == end) break;
        end = end->next;
    }

    struct node *new = new node();
    start->value = value;
    new->prev = start;
    new->next = start->next;
    start->next->prev = new;
}
int main() {
    return 0;
}
```

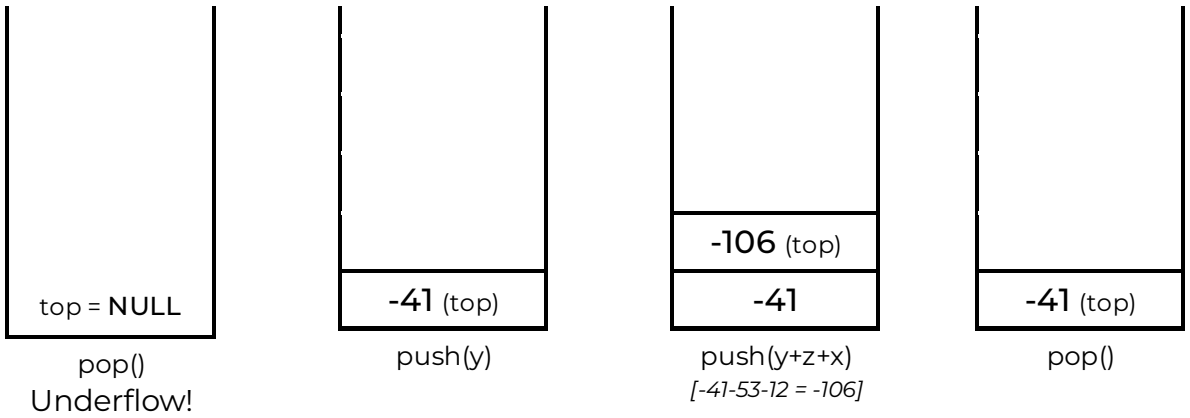
4. b) Show the status of a **STACK** implemented by a linear linked list for the operations given below. Here, x = -12 , y = x - 5 + 2x , and z = y + x.

Pop(), Push(y), Push(y+z+x), Pop(), Pop(), Push(z), Push(z), Push(x*y), Push(x+y), Push(y - z), Pop(), Pop().

Solution:

Here, x = -12
y = -12-5+2(-12) = -41
z = -12-41 = -53

The status of the Stack for each operation showed below:

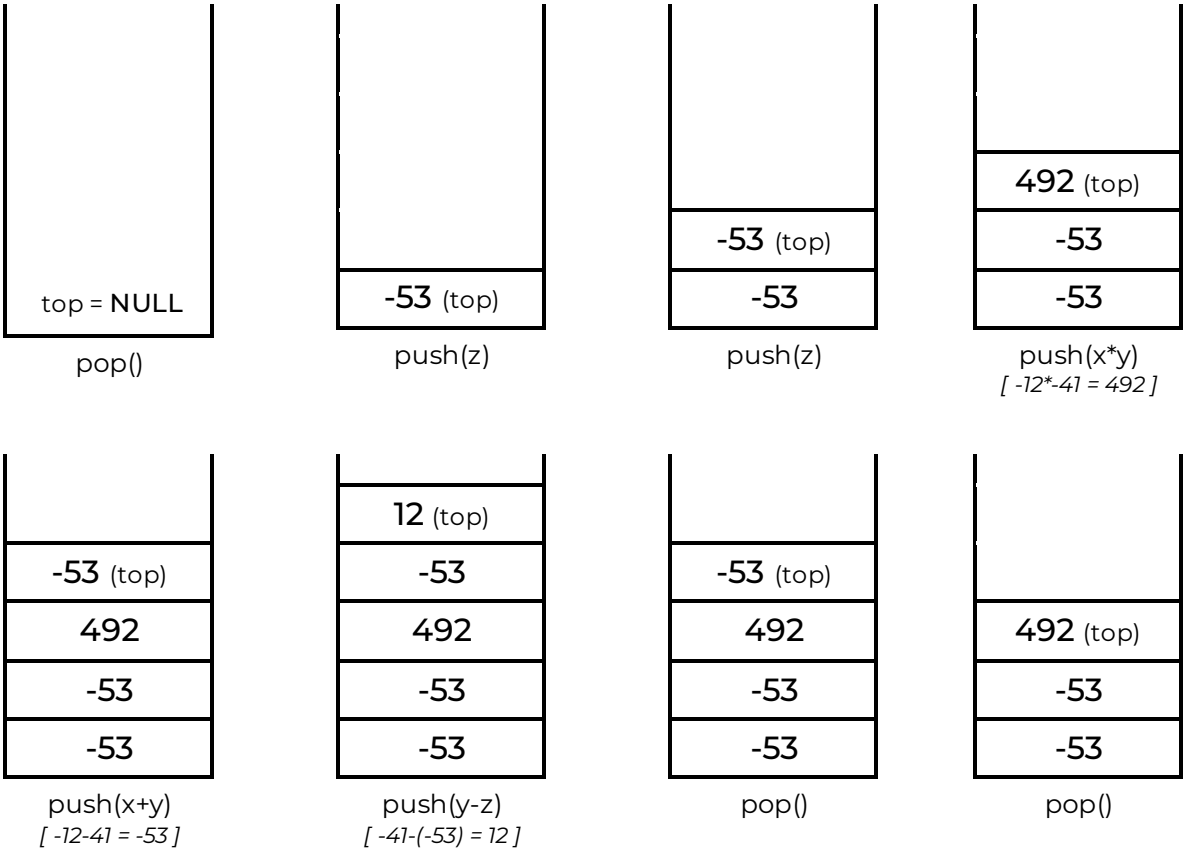


pop()
Underflow!

push(y)

push(y+z+x)
[-41-53-12 = -106]

pop()



4. c) Show the status of a **QUEUE** of size 5 implemented by an array for the operations given below. Here, x = 43 , y = x + 13, and z = 2y.

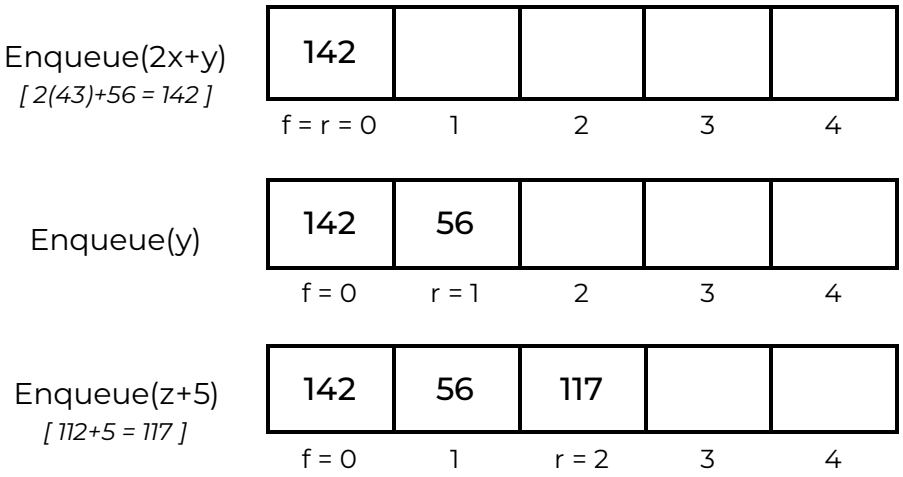
Here, Enqueue and Dequeue are meant by insertion and deletion, respectively.

Enqueue(2x+y), Enqueue(y), Enqueue(z+5), Dequeue(), Enqueue(y*x), Enqueue(y+z), Enqueue(z-y), Dequeue(), Dequeue().

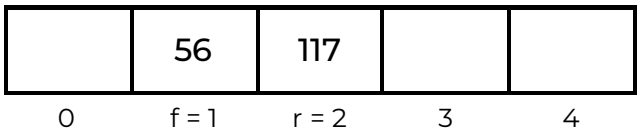
Solution:

Here, x = 43
y = 43+13 = 56
z = 2(56) = 112

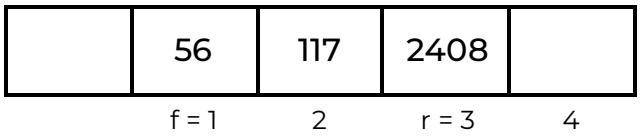
The status of the Stack for each operation showed below:



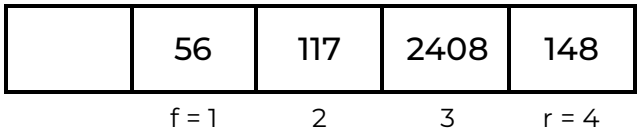
Dequeue()



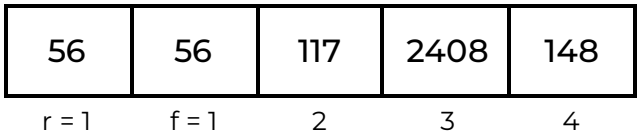
Enqueue(y*x)
*[56*43 = 2408]*



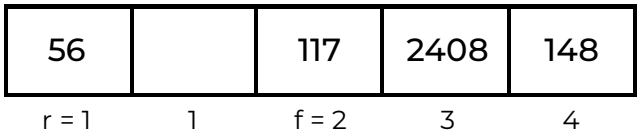
Enqueue(y+z)
[56+112 = 168]



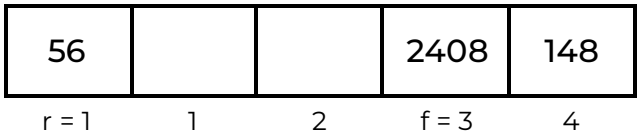
Enqueue(z-y)
[112-56 = 56]



Dequeue()



Dequeue()



1. a) How does the **Ascending Order Merge Sort** work on the following data?

y p z x r s

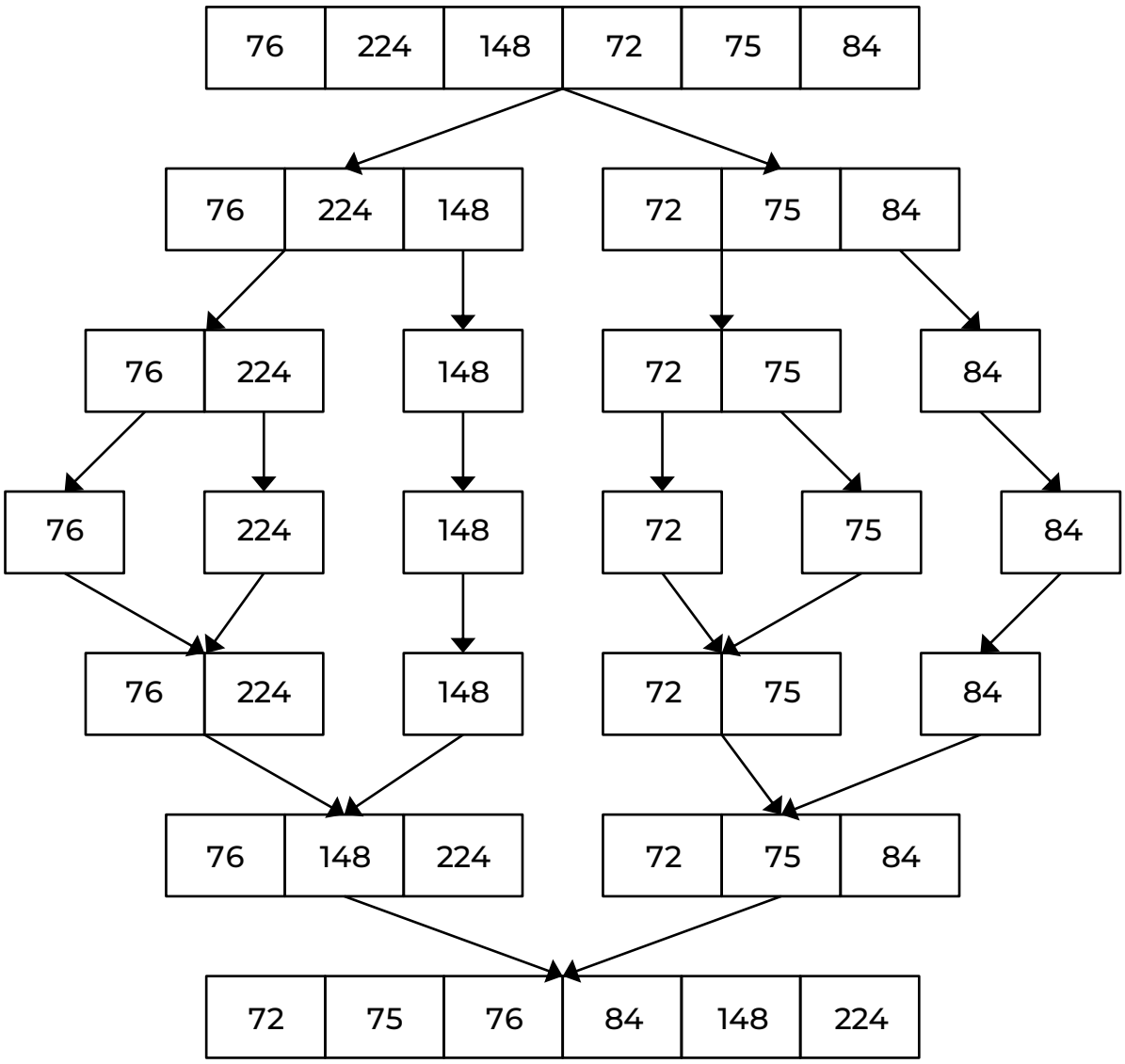
Here, x=last two digits of your student id+2, y=x+4, z=x+y, p=y+z, r=x+3, s=y+8

Solution:

Here, x = 70+2 = 72 p = 76+148 = 224
y = 72+4 = 76 r = 72+3 = 74
z = 72+76 = 148 s = 76+8 = 84

76	224	148	72	75	84
y	p	z	x	r	s

Ascending Merge Sort:



∴ Final sorted data is: 72, 75, 76, 84, 148, 224.

1. b) Discuss the time complexity of the following algorithm.

```
sum=0;
for(i=1; i<=n; i++){
    for(j=1; j<=n; j++){
        sum=sum+i+j;
    }
}
printf("%d", sum);
```

Solution:

Statements	Times
`sum=0;`	1
`for(i=1; i<=n; i++){`	n+1
`for(j=1; j<=n; j++){`	n(n+1)
`sum=sum+i+j;`	n*n
`}`	0
`}`	0
`printf("%d", sum);`	1

∴ Time complexity, $T(n) = 1 + n + 1 + n(n+1) + n \times n + 1$
$= 3 + n + n^2 + n + n^2$
$= O(n^2)$

∴ Time complexity of this algorithm is $O(n^2)$.

2. a) How many times the condition of while loop in the **Ascending Order Insertion Sort Algorithm** will be executed for the following data?

Insertion Sort Algorithm:

```
for j=2 to n do
   t=A[j]
   i=j-1
   while ((i>=1) AND (A[i]>t))
      A[i+1]=A[i]
      i=i-1
   end while
   A[i+1]=t
end for
```

Data Set-I: 40, 30, 20, 10
Data Set-II: 10, 20, 30, 40
Data Set-III: 30, 10, 20, 40

Solution:

Data Set-I: For this data set, condition of while loop will execute 6 times for true condition and 4-1 = 3 conditions for false condition.
∴ Total 6+3 = 9 times condition of while will be loop executed.

Data Set-II: For this data set, condition of while loop will execute 0 times for true condition and 4-1 = 3 conditions for false condition.

∴ Total 0+3 = 3 times condition of while will be loop executed.

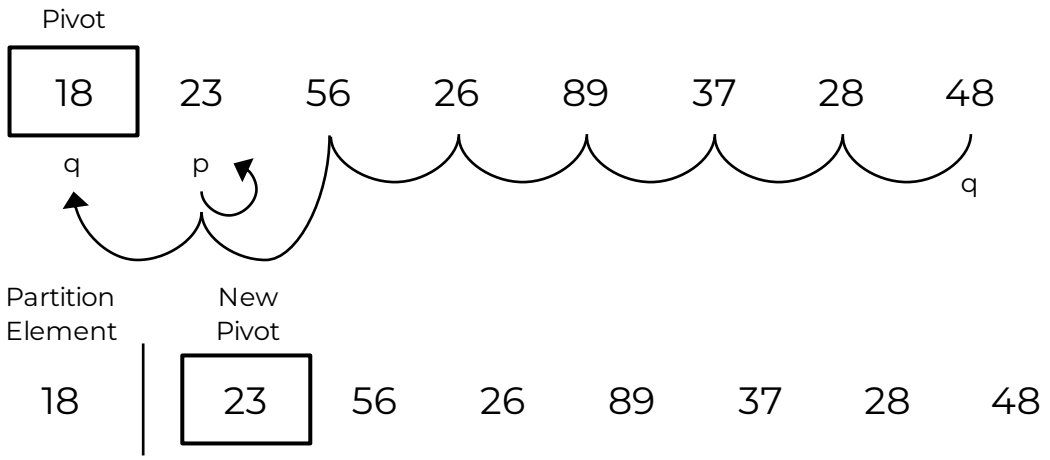
Data Set-III: For this data set, condition of while loop will execute 2 times for true condition and 4-1 = 3 conditions for false condition.
∴ Total 2+3 = 5 times condition of while will be loop executed.

2. b) Apply the Ascending Order Quick Sort Algorithm for the following instance to find the first partitioning element?

18 23 56 26 89 37 28 48

Solution:

Applying Ascending Order Quick Sort Algorithm,



∴ First portioning element is 18.

2. c) Find the memory location of A[70][60] if loc(A[20][15])=x+1300, where x=last four digits of your student ID. Assume row-wise memory is allocated in the floating point type array A[80][100], where each float data is 4 bytes.

Solution:

Here, $l_1 = 0$ $l_2 = 0$
$u_1 = 80 - 1 = 79$ $u_2 = 100 - 1 = 99$
$L = 4$ $x = 0170 = 170$
$b = ?$

We know,
For row-wise memory allocated 2D array,

$$loc(A[i][j]) = b + (i - l_1) \times (u_2 - l_2 + 1) \times L + (j - l_2) \times L$$

Given,

$$loc(A[20][15]) = x + 1300$$

or, $b + (20 - 0) \times (99 - 0 + 1) \times 4 + (15 - 0) \times 4 = 170 + 1300$

or, $b + 8000 + 60 = 1470$

or, $b = 1470 - 8060$

∴ $b = -6590$

Now,

$$loc(A[70][60]) = -6590 + (70 - 0) \times (99 - 0 + 1) \times 4 + (60 - 0) \times 4$$

or, $loc(A[70][60]) = -6590 + 28000 + 240$

$\therefore$ $loc(A[70][60]) = 21650$

$\therefore$ Memory location of A[70][60] is 21650.

3. a) How does the **Binary Search Algorithm** work on the following data?

Input Data: t r p z y x
Search Key = r

Here, x=last two digits of your student ID, y=x+4, z=x+y, p=y+z, r=z+p, and t=p+r

Solution:

Here, x = 70
y = 70+4 = 74
z = 70+74 = 144
p = 74+144 = 218
r = 144+218 = 362
t = 218+362 = 580

t	r	p	z	y	X
580	362	218	144	74	70
0	1	2	3	4	5

Search Key = r = 362

Comparison 1:

Lower Limit, $l = 0$
Higher Limit, $h = 5$
Mid Value, $m = \frac{l+h}{2} = \frac{0+5}{2} = 2.5 \approx 2$

Here, $Data[2] = 218, \quad 218 < 362$

$\therefore$ New Higher Limit, $h = m - 1 = 2 - 1 = 1$

Comparison 2:

Lower Limit, $l = 0$
Higher Limit, $h = 1$
Mid Value, $m = \frac{l+h}{2} = \frac{0+1}{2} = 0.5 \approx 0$

Here, $Data[0] = 580, \quad 580 > 362$

$\therefore$ New Lower Limit, $l = m + 1 = 0 + 1 = 1$

Comparison 3:

Lower Limit, $l = 1$
Higher Limit, $h = 1$
Mid Value, $m = \frac{l+h}{2} = \frac{1+1}{2} = 1$

Here, $Data[1] = 362, \quad 362 == 362$

$\therefore$ 'r' founded in Data

$\therefore$ 'r' founded in given data using Binary Search Algorithm.

3. b) If $f(n) = kn - 4$, prove that $f(n) = \theta(n)$. Here, k=last digit of your student id+5

Solution:

Here, $k = 0 + 5 = 5$

$f(n) = 5n - 4$

$g(n) = n$

Proving $f(n) \leq C_1 \cdot g(n)$ and $C_2 \cdot g(n) \leq f(n)$ is enough to prove $f(n) = \theta(n)$.

Let, $C_1 = 100$

Now, $f(n) \leq C_1 \cdot g(n)$

or, $5n - 4 \leq 100n$

Here, $f(n) \leq C_1 \cdot g(n)$ is true for all value of $n_0 \geq 1$.

Let, $C_2 = 1$

Now, $C_2 \cdot g(n) \leq f(n)$

or, $1n \leq 5n - 4$

Here, $C_2 \cdot g(n) \leq f(n)$ is true for all value of $n_0 \geq 1$.

$\therefore f(n) \leq C_1 \cdot g(n)$ and $C_2 \cdot g(n) \leq f(n)$ is true for all value of $n \geq 1$ over $C_1 = 100, C_2 = 1$.

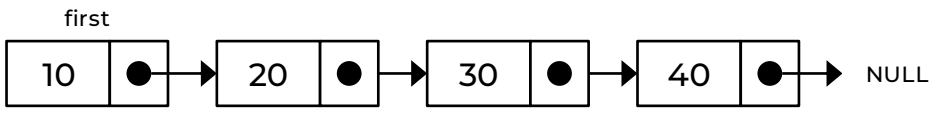
$\therefore f(n) = \theta(n)$. (Proved)

3. c) Suppose a linear linked list headed with "first" contains four nodes whose data values are 10, 20, 30, 40, respectively, where each node has two fields' **data** and **next**, where **data** is of integer type and **next** will contain the address of the next node. Show the following operations.

i) Draw a diagram for the linear linked list.
ii) Find a name for each of the nodes with respect to "first" that contain 10, 20, 30, 40, respectively?
iii) Write statements to represent 10, 20, 30, 40, respectively.
iv) How can you set NULL at the end of the linked list?
v) Design a code segment to insert 35 in-between 30 and 40.
vi) Convert your linear linked list to linear circular linked list by a code segment.

Solution:

i) The diagram has been drawn below:



ii) Name for each of the nodes with respect to "first" has been written below:

```
Node first = first;
Node second = first->next;
Node third = first->next->next;
Node fourth = first->next->next->next;
```

iii) Statements to represent 10, 20, 30, 40, respectively are written below.

10:
```
first->value;
```

20:
```
first->next->value;
```

30:
```
first->next->next->value;
```

40:
```
first->next->next->next->value;
```

iv) We can set NULL at the end of the linked list by this statement:

```
first->next->next->next->next = NULL;
```

v) Code segment to insert 35 in-between 30 and 40 are written below:

```
Node* new = new Node();
new->value = 35;
new->next = first->next->next->next;
first->next->next->next = new;
```

vi) We can delete a node containing 30 from the list by these statements:

```
Node* temp = first->next->next;
temp->next = first->next->next->next;
first->next->next = temp->next;
delete temp;
```

vii) We can convert this linked list to a circular linked list by these statements:

```
Node* temp = first;
while(temp->next != NULL) {
    temp = temp->next;
}
temp->next = first;
```

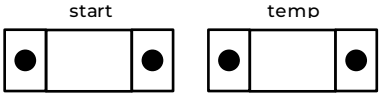
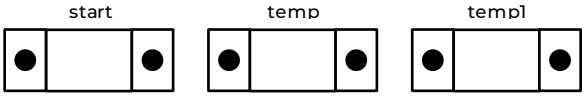
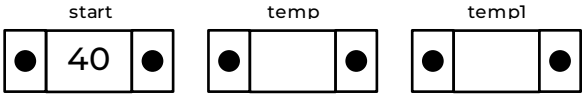
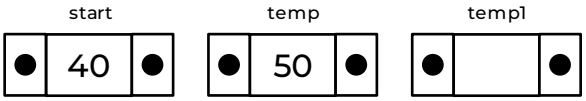
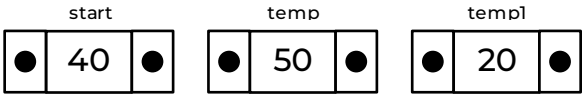
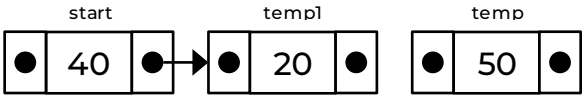
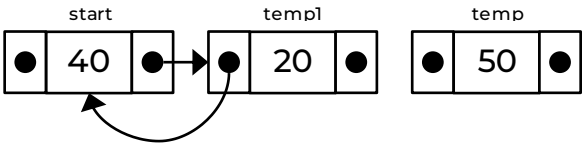
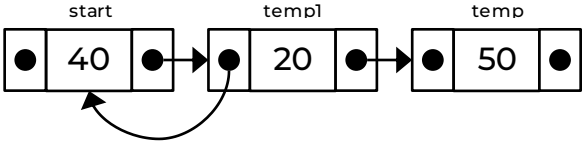
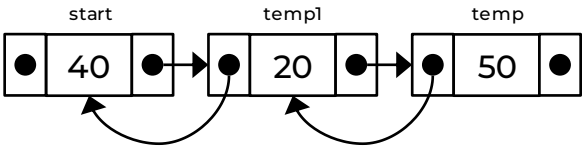
4. a) Show the effect of each of the statements given in the following code segment. Assume, each of the nodes in the doubly linked list has three fields' **data**, **next** and **prev**, where data is of integer type, **next** and **prev** will contain the address of the next and previous nodes, respectively.

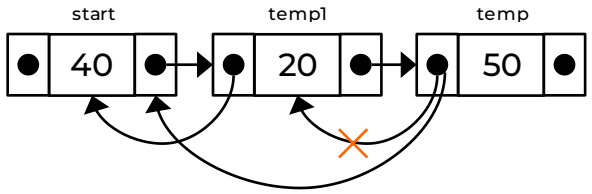
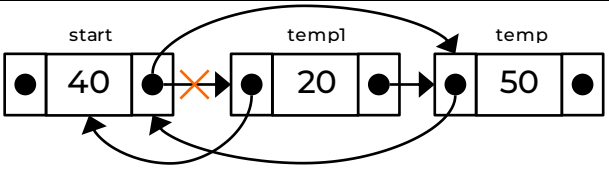
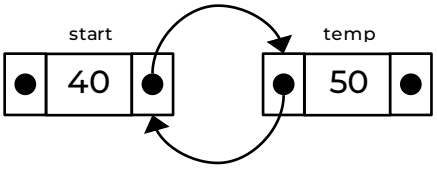
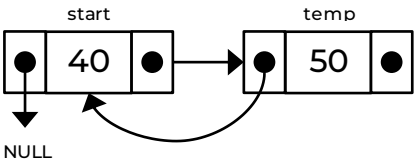
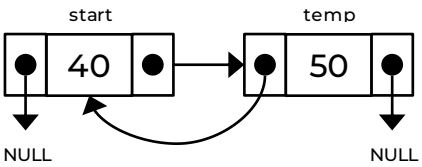
```
start=(node*)malloc(sizeof(node));
temp=(node*)malloc(sizeof(node));
temp1=(node*)malloc(sizeof(node));
start->data =40;
temp->data=50;
temp1->data=20;
start->next=temp1;
```

```
temp1->prev=start;
start->next->next=temp;
temp->prev=temp1;
temp1->next->prev=temp1->prev;
temp1->prev->next=temp1->next;
free(temp1);
start->prev=NULL;
temp->next=NULL;
```

Solution:

Effect of each of the given statements are showed below:

Statements	Effect
`start=(node*)malloc(sizeof(node));`	
`temp=(node*)malloc(sizeof(node));`	
`temp1=(node*)malloc(sizeof(node));`	
`start->data=40;`	
`temp->data=50;`	
`temp1->data=20;`	
`start->next=temp1;`	
`temp1->prev=start;`	
`start->next->next=temp;`	
`temp->prev=temp1;`	

`temp1->next->prev=temp1->prev;`	
`temp1->prev->next=temp1->next;`	
`free(temp1);`	
`start->prev=NULL;`	
`temp->next=NULL;`	

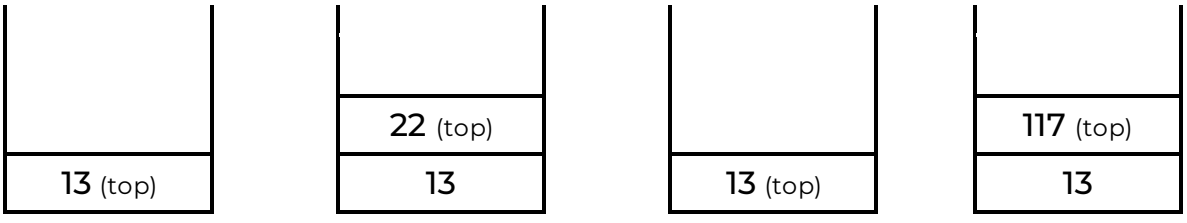
4. b) Show the status of a STACK implemented by a linear linked list for the operations given below. Here, x=last digit of your student id+4, y=x+5, and z=y+x.

Push(x+y), Push(y+z), Pop(), Push(y*z), Push(x*y), Pop(), Pop(), Push(x+z)

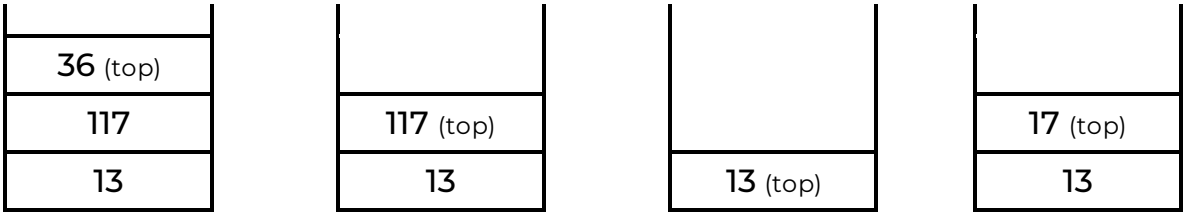
Solution:

Here, x = 0+4 = 4
y = 4+5 = 9
z = 9+4 = 13

The status of the Stack for each operation showed below:



[P.T.O]



push(x*y)
*[4*9 = 36]*

pop()

pop()

push(x+z)
[4+13 = 17]

4. c) Show the status of a QUEUE of size 3 implemented by an array for the operations given below. Here, x=last digit of your student id+4, y=x+5, and z=y+x. Here, Enqueue and Dequeue are meant by insertion and deletion, respectively.

Enqueue(x+y), Dequeue(), Enqueue(y*z), Enqueue(x*y), Dequeue(), Enqueue(y+z)

Solution:

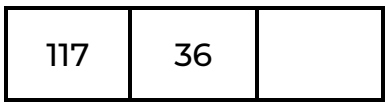
Here, x = 0+4 = 4
y = 4+5 = 9
z = 9+4 = 13

The status of the Stack for each operation showed below:

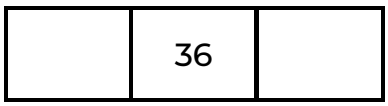
Enqueue(x+y)
[4+9 = 13]

Dequeue()

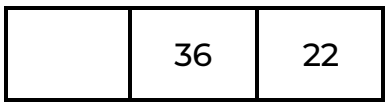
Enqueue(y*z)
*[9*13 = 117]*



Enqueue(x*y)
*[4*9 = 36]*



Dequeue()



Enqueue(y+z)
[9+13 = 22]

Fall 2022

1. a) Demonstrate how **Descending Order Merge Sort** will work on the following data?

y p z x r s

Here, x=last two digits of your student id+1, y=x+3, z=x+y, p=y+z, r=x+2, s=y+9

Solution:

Repeat of Summer 2023 Question 1(a)

1. b) Discuss the time complexity of the following algorithm.

```
sum=0;
for(i=1; i<=n; i++){
    for(j=1; j<=i; j++){
        sum=sum+i+j;
    }
}
printf("%d", sum);
```

Solution:

Statements	Times
`sum=0;`	1
`for(i=1; i<=n; i++){`	n+1
`for(j=1; j<=i; j++){`	T
`sum=sum+i+j;`	T-1
`}`	0
`}`	0
`printf("%d", sum);`	1

Here, $T = 1 + 2 + 3 + \cdots + n = \frac{n^2 + n}{2}$

∴ Time complexity, $T(n) = 1 + n + 1 + \frac{n^2 + n}{2} + \frac{n^2 + n}{2} - 1 + 1$

$= 2 + n + n^2 + n$

$= O(n^2)$

∴ Time complexity of this algorithm is $O(n^2)$.

2. a) How many times the condition of while loop in the **Ascending Order Insertion Sort Algorithm** will be executed for the following data?

Data Set-I: 10, 20, 30, 40
Data Set-II: 40, 30, 20, 10
Data Set-III: 30, 10, 40, 20

Solution:

Data Set-I: For this data set, condition of while loop will execute 0 times for true condition and 4-1 = 3 conditions for false condition.
∴ Total 0+3 = 3 times condition of while will be loop executed.

Data Set-II: For this data set, condition of while loop will execute 6 times for true condition and 4-1 = 3 conditions for false condition.
∴ Total 6+3 = 9 times condition of while will be loop executed.

Data Set-III: For this data set, condition of while loop will execute 3 times for true condition and 4-1 = 3 conditions for false condition.
∴ Total 3+3 = 6 times condition of while will be loop executed.

2. b) How many element comparisons are needed for the following instance of the **Ascending Order Quick Sort** to find the first portioning element?

18 23 56 26 89 37 28 48

Solution:

*Repeat of **Fall 2023 Question 1(c)***

2. c) Find the memory location of A[60][70] if loc(A[15][20])=x+1200, where x=last four digits of your student ID. Assume column-wise memory is allocated in the floating point type array A[80][100], where each float data is 4 bytes.

Solution:

Here, $l_1 = 0$ $\quad l_2 = 0$
$u_1 = 80 - 1 = 79$ $\quad u_2 = 100 - 1 = 99$
$L = 4$ $\quad x = 0170 = 170$
$b = ?$

We know,
For column-wise memory allocated 2D array,
$$loc(A[i][j]) = b + (j - l_2) \times (u_1 - l_1 + 1) \times L + (i - l_1) \times L$$

Given,
$$loc(A[15][20]) = x + 1200$$
or, $b + (20 - 0) \times (79 - 0 + 1) \times 4 + (15 - 0) \times 4 = 170 + 1200$
or, $b + 6400 + 60 = 1370$
or, $b = 1370 - 6460$
∴ $b = -5090$

Now,
$$loc(A[60][70]) = -5090 + (70 - 0) \times (79 - 0 + 1) \times 4 + (60 - 0) \times 4$$
$$loc(A[60][70]) = -5090 + 22400 + 240$$
∴ $loc(A[60][70]) = 17550$

∴ Memory location of A[60][70] is 33730.

3. a) How does the **Binary Search Algorithm** work on the following data?

Input Data: t r p z y x
Search Key = y

Here, x=last two digits of your student ID, y=x+3, z=x+y, p=y+z, r=z+p, and t=p+r

Solution:

*Repeat of **Fall 2023 Question 2(c)***

3. b) If $f(n) = kn^2 - 5$, prove that $f(n) = \Theta(n^2)$. Here, k=last digit of your student id+2.

Solution:

Here, $k = 0 + 2 = 2$
$f(n) = 2n^2 - 5$
$g(n) = n^2$

Proving $f(n) \leq C_1 \cdot g(n)$ and $C_2 \cdot g(n) \leq f(n)$ is enough to prove $f(n) = \Theta(n^2)$.

Let, $C_1 = 100$

Now, $f(n) \leq C_1 \cdot g(n)$
or, $2n^2 - 5 \leq 100n$

Here, $f(n) \leq C_1 \cdot g(n)$ is true for all value of $n_0 \geq 1$.

Let, $C_2 = 1$

Now, $C_2 \cdot g(n) \leq f(n)$
or, $1n \leq 2n^2 - 5$

Here, $C_2 \cdot g(n) \leq f(n)$ is true for all value of $n_0 \geq 2$.

∴ $f(n) \leq C_1 \cdot g(n)$ and $C_2 \cdot g(n) \leq f(n)$ is true for all value of $n \geq 2$ over $C_1 = 100, C_2 = 1$.

∴ $f(n) = \Theta(n^2)$. (Proved)

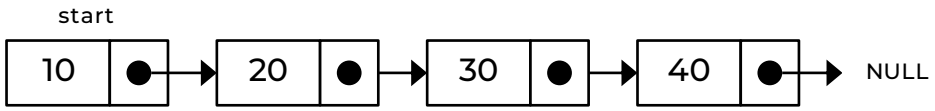
3. c) Suppose a linear linked list headed with "start" contains four nodes whose data values are 10, 20, 30, 40, respectively. Show the following operations.

i) Draw a diagram for the linear linked list.
ii) Find a name for each of the nodes with respect to "start" that contain 10, 20, 30, 40, respectively?
iii) Write statements to represent 40, 50, 30, 20, respectively.
iv) Write a statement to set NULL at the end of the linked list.

Solution:

i) The diagram has been drawn below:

[P.T.O]



ii) Name for each of the nodes with respect to "start" has been written below:

```
Node first = start;
Node second = start->next;
Node third = start->next->next;
Node fourth = start->next->next->next;
```

iii) Statements to represent 10, 20, 30, 40, respectively are written below.

10: `start->value;`

20: `start->next->value;`

30: `start->next->next->value;`

40: `start->next->next->next->value;`

iv) Statement to set NULL at the end of the linked list are written below:

```
start->next->next->next = NULL;
```

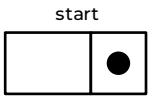
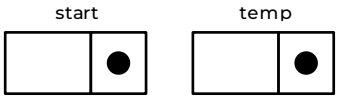
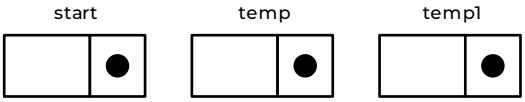
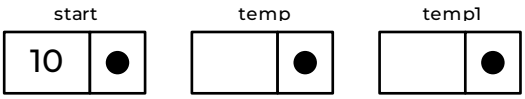
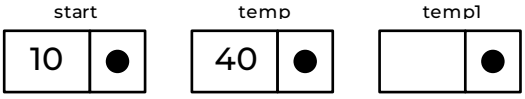
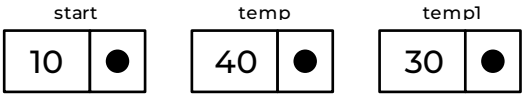
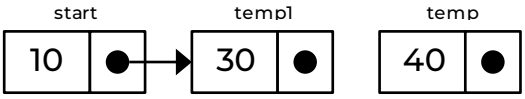
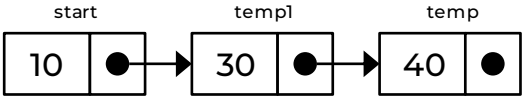
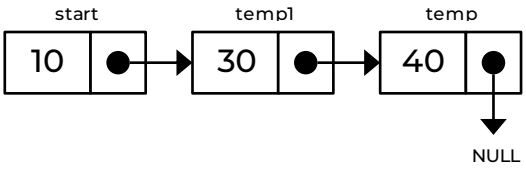
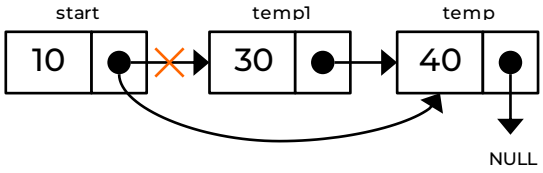
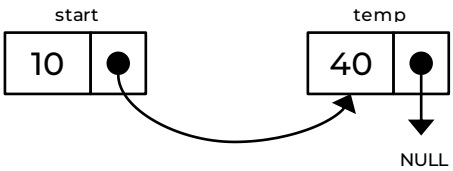
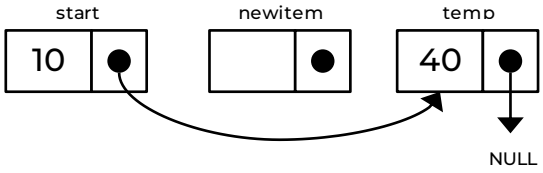
4. a) Show the effect of each of the statements given in the following code segment. Assume, each of the nodes in the linear linked list has two fields' **data** and **next**, where **data** is of integer type, and **next** will contain the addresses of the next nodes.

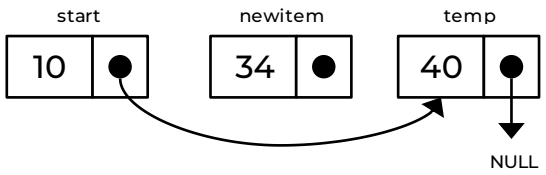
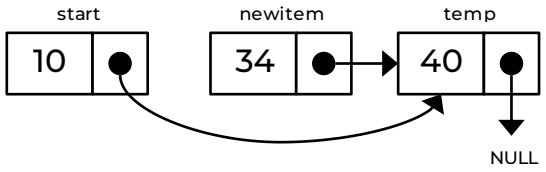
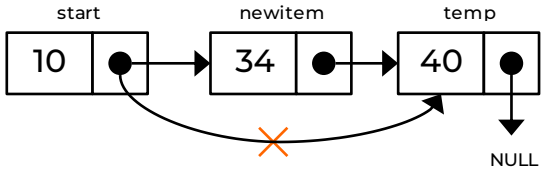
```
start=(node*)malloc(sizeof(node));
temp=(node*)malloc(sizeof(node));
temp1=(node*)malloc(sizeof(node));
start->data=10;
temp->data=40;
temp1->data=30;
start->next=temp1;
start->next->next=temp;
temp->next=NULL;
start->next=temp1->next;
free(temp1);
newitem=(node*)malloc(sizeof(node));
newitem->data=34;
newitem->next=start->next;
start->next=newitem;
```

Solution:

Effect of each of the given statements are showed below:

[P.T.O]

Statements	Effect
`start=(node*)malloc(sizeof(node));`	
`temp=(node*)malloc(sizeof(node));`	
`temp1=(node*)malloc(sizeof(node));`	
`start->data=10;`	
`temp->data=40;`	
`temp1->data=30;`	
`start->next=temp1;`	
`start->next->next=temp;`	
`temp->next=NULL;`	
`start->next=temp1->next;`	
`free(temp1);`	
`newitem=(node*)malloc(sizeof(node));`	

`newitem->data=34;`	
`newitem->next=start->next;`	
`start->next=newitem;`	

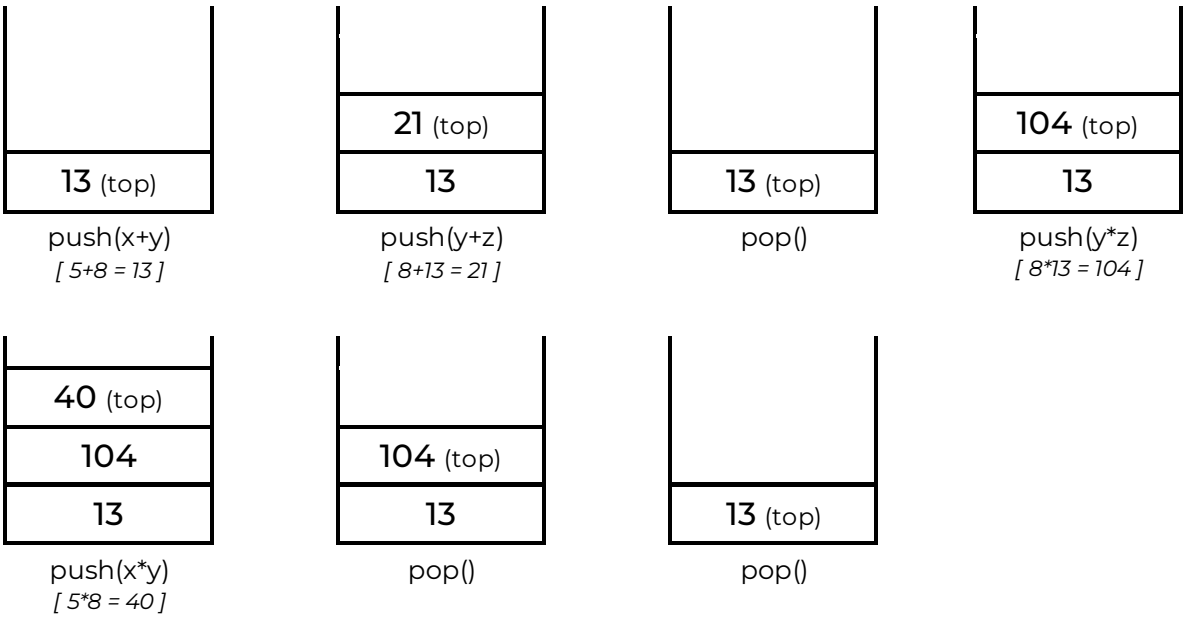
4. b) Show the status of a STACK implemented by a linear linked list for the operations given below. Here, x=last digit of your student id+5, y=x+3, and z=y+x.

Push(x+y), Push(y+z), Pop(), Push(y*z), Push(x*y), Pop(), Pop()

Solution:

Here, x = 0+5 = 5
y = 5+3 = 8
z = 8+5 = 13

The status of the Stack for each operation showed below:



4. c) Show the status of a QUEUE of size 3 implemented by an array for the operations given below. Here, x=last digit of your student id+5, y=x+3, and z=y+x. Here, Enqueue and Dequeue are meant by insertion and deletion, respectively.

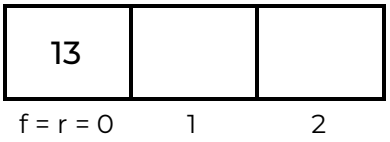
Enqueue(x+y), Enqueue(y+z), Dequeue(), Enqueue(y*z), Enqueue(x*y), Dequeue()

Solution:

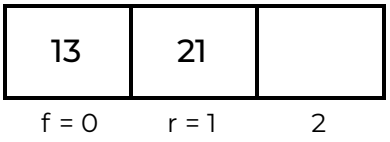
Here, x = 0+5 = 5
y = 5+3 = 8
z = 8+5 = 13

The status of the Stack for each operation showed below:

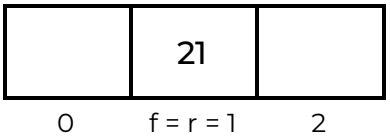
Enqueue(x+y)
[5+8 = 13]



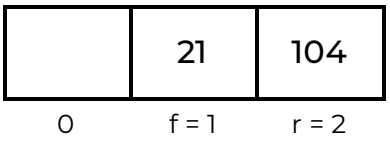
Enqueue(y+z)
[8+13 = 21]



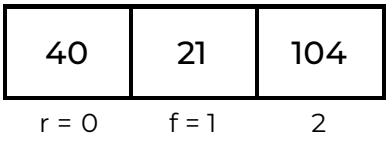
Dequeue()



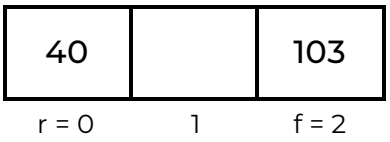
Enqueue(y*z)
*[8*13 = 104]*



Enqueue(x*y)
*[5*8 = 40]*



Dequeue()



Summer 2022

1. a) How does the ascending order Merge Sort algorithm work on the following data?

y p z x r s

Here, x=last two digits of your student id+1, y=x+3, z=x+y, p=y+z, r=x+3, s=y+8

Solution:

*Repeat of **Spring 2023 Question 1(a)***

1. b) Discuss the time complexity of the following algorithm.

```
sum=0;
for(i=2; i<=n; i++){
    for(j=2; j<=i; j++){
        sum=sum+i+j;
    }
}
printf("%d", sum);
```

Solution:

*Repeat of **Summer 2023 Question 1(a)***

2. c) Find the memory location of A[40][70] if loc(A[15][20])=8000+w, where x=last four digits of your student ID. Assume row-wise memory is allocated in the double array A[80][100], where each float data is 8 bytes.

Solution:

Here, $l_1 = 0$ $l_2 = 0$

$u_1 = 80 - 1 = 79$ $u_2 = 100 - 1 = 99$

$L = 8$ $w = 0170 = 170$

$b = ?$

We know,

For row-wise memory allocated 2D array,

$$loc(A[i][j]) = b + (i - l_1) \times (u_2 - l_2 + 1) \times L + (j - l_2) \times L$$

Given,

$$loc(A[15][20]) = 8000$$

or, $b + (15 - 0) \times (99 - 0 + 1) \times 8 + (20 - 0) \times 8 = 8000 + 170$

or, $b + 12000 + 160 = 8170$

or, $b = 8170 - 12160$

$\therefore$ $b = -3990$

Now,

$$loc(A[40][70]) = -3990 + 32000 + (70 - 0) \times 8$$

or, $loc(A[40][70]) = -3990 + 28000 + 560$

$\therefore \quad loc(A[40][70]) = 24570$

∴ Memory location of A[40][70] is 24570.

2. b) How does the Binary Search algorithm work on the following data?

Input Data: t r p z y x
Search Key = y

Here, x=last two digits of your student ID, y=x+4, z=x+y, p=y+z, r=z+p, and t=p+r
Also find the total element comparisons for the given instance of the Binary Search.

Solution:

*Repeat of **Fall 2023 Question 2(c)***

2. c) If $f(n) = kn - 5$, prove that $f(n) = \Theta(n)$. Here, k=last digit of your student id+4

Solution:

Here, $k = 0 + 4 = 4$
$f(n) = 4n - 5$
$g(n) = n$

Proving $f(n) \leq C_1 \cdot g(n)$ and $C_2 \cdot g(n) \leq f(n)$ is enough to prove $f(n) = \Theta(n)$.

Let, $C_1 = 100$

Now, $f(n) \leq C_1 \cdot g(n)$
or, $4n - 5 \leq 100n$

Here, $f(n) \leq C_1 \cdot g(n)$ is true for all value of $n_0 \geq 1$.

Let, $C_2 = 1$

Now, $C_2 \cdot g(n) \leq f(n)$
or, $1n \leq 4n - 5$

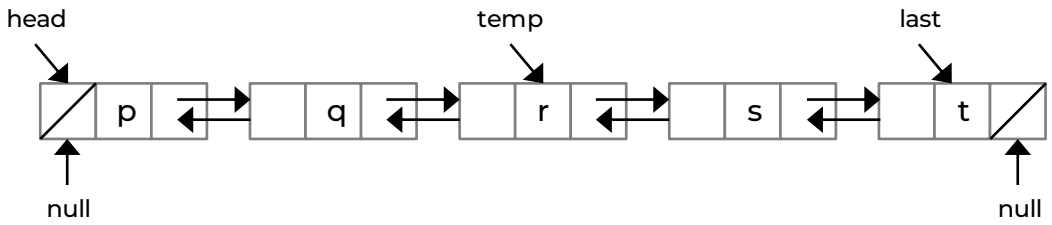
Here, $C_2 \cdot g(n) \leq f(n)$ is true for all value of $n_0 \geq 2$.

$\therefore f(n) \leq C_1 \cdot g(n)$ and $C_2 \cdot g(n) \leq f(n)$ is true for all value of $n \geq 2$ over $C_1 = 100, C_2 = 1$.

$\therefore f(n) = \Theta(n)$. (Proved)

3. a) Answer the following questions for the doubly linked list as shown below, where p = last two digits of your student id + 9, q = p+4, r = p+q, s = r-3, t = r+s.

a) head->next->next->value = ?
b) last->prev->next->value = ?
c) temp->prev->prev->prev = ?
d) temp->next->prev->prev->value = ?
e) last->prev->prev->next->value = ?



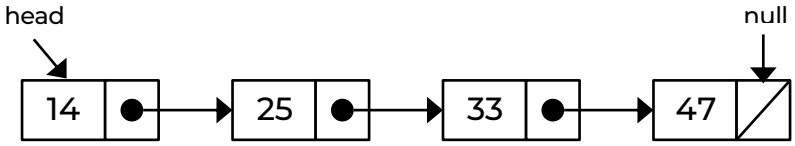
Solution:

Here, p = 70+9 = 79
q = 79+4 = 83
r = 78 + 83 = 162
s = 162-3 = 159
t = 162+159 = 321

a) head->next->next->value = r = 162
b) last->prev->next->value = t = 321
c) temp->prev->prev->prev = null
d) temp->next->prev->prev->value = q = 83

3. b) Assume that you are given a single linked list as shown below. Write the statements to perform the following:

i) To insert 40 in between 33 and 47.
ii) To delete 14 from the list
iii) To make a linear circular linked list from the current list.



Solution:

i)

```
Node *newitem = new Node();
newitem->value = 47;
newitem->next = head->next->next->next;
head->next->next->next = newitem;
```

ii)

```
Node *temp = head;
head = head->next;
delete temp;
```

iii)

```
Node *temp = head;
while(temp->next != NULL) {
    temp = temp->next;
}
temp->next = head;
```

4. b) Show the status of a STACK implemented by a linear linked list for the operations given below. Here, x=last digit of your student id+5, y=x+3, and z=y+x.

Push(x+y), Push(y+z), Pop(), Push(y*z), Push(x*y), Pop(), Pop()

Solution:

*Repeat of **Fall 2022 Question 4(b)***

4. b) Show the effect of each of the statements given in the following code segment. Assume, each of the nodes in the linear linked list has two fields' data and next, where data is of integer type, and next will contain the addresses of the next nodes.

```
start=(node*)malloc(sizeof(node));
temp=(node*)malloc(sizeof(node));
temp1=(node*)malloc(sizeof(node));
start->data=10;
temp->data=40;
temp1->data=30;
start->next=temp1;
start->next->next=temp;
temp->next=NULL;
start->next=temp1->next;
free(temp1);
newitem=(node*)malloc(sizeof(node));
newitem->data=34;
newitem->next=start->next;
start->next=newitem;
```

Solution:

*Repeat of **Fall 2022 Question 4(a)***

4. c) Write an algorithm to display the data stored in a double linked list in reverse order. Assume only head pointer is given for the linked list.

Solution:

The algorithm has been written below:

```
Node *temp head;
while(temp->next != NULL) {
    temp = temp->next;
}
while(temp != NULL) {
    printf("%d", temp->value);
    temp = temp->back;
}
```

4. d) Show the status of a QUEUE of size 3 implemented by an array for the operations given below. Here, x=last digit of your student id+5, y=x+3, and z=y+x. Here, Enqueue and Dequeue are meant by insertion and deletion, respectively.

Enqueue(x+y), Enqueue(y+z), Dequeue(), Enqueue(y*z), Enqueue(x*y), Dequeue()

Solution:

*Repeat of **Fall 2022 Question 4(d)***