Question 1

Write a function “insert\_any ()” for inserting a node at any given position of the linked list. Assume

position starts at 0.

Ans).

Node insert\_any (int data, int pos)

{

struct node \*temp;

struct node \*temp1;

temp1=create(data);//new entry

temp=head;

if(temp==Null)

{

head=temp1;

temp1->link=Null

}

for(int i=1;i<pos-1&&temp!=Null;i++)

temp=temp->link;//will move to pos – 1 position

temp1->link=temp->link;

temp->link=temp1;

}

Question 2

Write a function “delete\_beg()” for deleting a node from the beginning of the linked list.

Ans).

Node delete\_beg ()

{

Struct Node \*temp;

temp=head;

if(head->link==0)

{

head=Null;

free(temp);

}

else

{

head=head->link;

free(temp);

}

}

Question 3

Write a function “delete\_end()” for deleting a node from the end of the linked list.

Ans).

Struct delete\_end()

{

Struct node \*temp=head;

Struct node\*prev;

while(temp->link!=Null)

{

prev=temp;

temp=temp->link;

}

If(temp==head)

{

head =Null;

free(temp);

}

else

{

prev->link=Null;

free(temp);

}

Question 4

In the Binary Search algorithm, it is suggested to calculate the mid as beg + (end - beg) / 2

instead of (beg + end) / 2. Why is it so?

Ans).

In the first case, (end-beg)ensures we will never go beyond end, so there is no chance of overflow, therefore, we prefer this method rather than (beg+end)/2.

Question 5

Write the algorithm/function for Ternary Search.

Ans).

int ternary(int arr[],int n,int key)

{

l=0;

r=n-1;

while(l<=r)

{

mid1=l+(r-l)/3;

mid2=r-(r-l)/3;

if(key==arr[mid1])

return mid1;

if(key==arr[mid2])

return mid2;

if(key<arr[mid1])

r=mid1-1;

else if(key>mid2)

l=mid2+1;

else

{

l=mid1+1;

r=mid2-1;

}

}