1) Write a program to insert and delete an element at the nth and kth position in a linked list where n and k are taken from user.

```
#include <stdio.h>
#include <stdlib.h>
struct node{
 int data;
 struct Node *next;
};
struct Node *head;
void insert(int data, int n){
 Node *temp = new node();
  temp -> data = data;
  temp -> next = null;
  if (n==1){
   temp -> next = head;
     head = temp;
     return;
  }
   void Delete (int k){
   struct node *temp = head;
    if (k==1){
     head = temp -> next;
     free (temp);
      return;
    }
```

```
Node *temp = head;
for (int i=0; i<n-2; i++){
 temp = temp -> next;
}
temp -> next = temp -> next ;
temp -> next = temp;
}
    void print ();
 for (int i=0; i<k-2; i++)
  temp = temp -> next;
  free (temp);
 }
int main(){
int n, a, k;
 head = null;
Printf ("enter the position for inserting: ");
Scanf ("%d", &n);
Scanf ("%d", &a);
 Insert (a, n);
  Printf ("enter the position to delete");
  Scanf ("%d", &k);
  Delete (k);
  Printf (n); → return; }
```

(2.) Construct a new linked list by merging alternate nodes of two lists ?

```
#include <stdio.h>
#include <stdlib.h>

struct Node {
    int data;
    struct Node * next;
};

void push ( struct node ** head_ref , int new_data)
{
    struct Node* new_node = (struct Node*) malloc (sizeof(struct node));
    new_node -> data = new data;
    new_node -> next = (*head_ref);
    (*head_ref) = new_node;
}

void printlist (struct Node *head)
{
    struct Node *temp = head;
    while (temp != NULL)
    {
        Printf("%d", temp -> data );
        temp = temp -> next;
    }
    Printf("\n");
}

void merge (struct node *p, struct node **q)
{
    Struct Node *p_curr = p, *q_curr = *q;
    Struct Node *p_next, = *q_next;
```

```
while (p_curr != Null && q_curr != Null)
{
    p_next = p_curr -> next;
    q_next = q_curr -> next;

    q_curr -> next = p_next;
    p_curr -> Next = q_curr;

    p_curr = p_next;
    q_curr = q_next;
  }
   *q = q_curr;
}
 int main()
 {
   Struct Node *p = Null, *q = Null;
    Push (&p, 3);
    Push (&p, 2);
    push (&p, 1);
    Printf (" First linked list \n");
    Printf(
     Printlist(p);

     Push (&q, 6);
     Push (&q, 5);
     Push (&q, 4);
     Printf ("Second. linked list \n");
     Printlist(q);
```

(3.) Find all the elements in the stack whose sum is equal to k.

```
#include<stdio.h>
int top=-1;
int n;
char stack[100];
void push(int x);
char pop();
int main()
{
int i, n, a, t, k, j, sum=0, count=1;
Printf("enter the number of elements in the stack");
Scanf("%d", &n);
for(i=0; i<n; i++){
Printf("enter the next element");
Scanf("%d", &a);
Push(a);
}
Printf("enter the sum to be Checked");
Scanf("%d", &k);
for(i=0; i<n; i++)
{
t=pop();
Sum+=t;
Count+=1;
if(Sum==k)
{
for(int j=0; j<count; j++)
Printf("%d", stack[j]);
```

```
f=1;
break;
}
push(t);
}
if(j!=1)
printf("The elements in the stack don't add upto
        sum");
}
void push(int n)
{ if(top==99)
  {
    printf("stack is Full");
     return;
  }
  top=top+1;
  stack[top]=n;
  }
   char pop()
   {
     if(stack[top]==-1)
     {
      printf("stack is empty");
      return 0;
     }
     n=stack[top];
      top=top-1;
   } return n;
```

(10) Write a program to print the elements in a queue

i) Reverse order.

```
#include <stdio.h>
# define SIZE 10
  void insert (int);
  void delete ();
  int queue [10], f = 1, r = -1;
  void main () {
   int value choice;
    while (1) {
     Printf (" MENU ");
     Printf (" 1) insertion \n
              2) Deletion \n
              3) Print Reverse \n
              4) Print Alternate in
              5) Eint ");
      Scanf ("
     Printf (" Enter your choic ");
     Scanf ("%d", &choice);
     Switch (chocie) {
  Case 1 ; Printf (" enter the value to be insert ");
    Scanf ("%d", & value);
 insert(value);
  break;
 Case 2: delete (); break;
```

```
case 3:
    printf("The reversed queue is ");
    for(int i=SIZE; i>=0; i--)
    {
        if(queue[i]==0){
            continue;
        }
        printf("%d", queue[i]);
    }
    break;
case 4:
    printf("Alternate elements of the queue are ");
    for(int i=0; i<SIZE; i+=2){
        if(queue[i]==0){
            continue;
        }
        printf("%d", queue[i]);
    }
    break;
case 5: exit(0);
default: printf("In wrong selection");
}
}
void insert(int value)
{
    if((f==0 && r==SIZE-1) || f==r+1)
        printf("queue is full");
```

```
else {
  if (f == r-1)

f = 0;
 r = (r+1) % size;
  queue[r] = value;
 printf ("\n insertion success");
 }
 {
  void delete() {
   if (f == r-1)
     printf ("\n queue is empty");
    else {
    printf ("\n Deleted %d", queue[f]);
      f = (f+1) % SIZE;
       if (f == r)
    f = r = -1;
  }
 }
```

(5)

i) Difference between array and linked list is that their structure are different. arrays are index based data structure where each element associated with an index and linked list refers having a node. Which node consists of the data and the references to the previous and next element.

ii) write a program to add the first element of one list to another list.

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
  int data;
  struct node *next;
};
void printlist ( struct node* head)
{
  struct node * ptr = head;
  while (ptr)
  {
    printf ("%d -> ", ptr -> data);
    ptr = ptr -> next;
  }
  printf ("Null");
}
void push ( struct Node ** head, int data)
```

```
{
struct Node* newNode = (struct Node*) malloc(sizeof(struct
                                                Node));
  newNode -> data = data;
  newNode -> next = *head;
  *head = newNode;
}

void moveNode(struct Node** destRef, struct Node**
     sourceRef)
{
  if (*sourceRef == NULL)
  return;
  struct Node* newNode = *sourceRef;
 *sourceRef = (*sourceRef) -> next;
  newNode -> next = *destRef;
  *destRef = newNode;
}
  int main(void)
{
   int keys[] = {1, 2, 3};
   int n = sizeof(keys)/sizeof(keys[0]);
  struct Node * a = NULL;
  for(int i=n-1; i>=0; i--)
   push(&b, 2*keys[i]);

    moveNode(&a, &b);
   printf("first list: ");
   printList(a);
```

```
printf("Second list");
printList(b);

return 0;
}
```