ASSIGNMENT-4

AP19110010101
CSE-H.

1) Write a c Programme to insert and delete an element at the nth and kth Position in a linked list where n and k is taken from user.

```
# include <stdio.h>
# include <stdbli.h>
struct Node
{
  int data;
  struct Node* next;
};
struct Node* delete (struct Node * head, int n);
struct Node* insert (struct Node * head, int n);
 struct Node * Create-list();
 void display (struct Node*);
 void
  main()
 { int k;
  struct Node * head;
   head = Create-list();
   display (head);
    Printf ("Enter the Index where you want to be inser
                enter:");
    scanf ("%d", & k);
     head = insert (head, k);
     display (head);
     head = delete (head, 3);
     display (head);
 }
 void display (struct Node * head)
 {
```

```
struct Node *c;
for (c=head; c!=NULL; c=c->next)
 {
  Printf ("\n Node data %d", c->data);
 }
 Printf("\n");
}

struct Node* create_list()
{ int k,n;
  struct Node *c *Head;
  Printf("\n enter the no:of elements to be enterd");
  scanf("%d", &n);
  for (k=0; k<n; k++)
   { if (k==0)
   {
    Head = (struct Node*) malloc (size of(struct Node));
    c= Head;
  }
   else
  {
   c->next = (struct Node*) malloc (sizeof (struct Node));
   c= c->nelt;
 } Printf("\n enter an %d th element", k);
   scanf ("%d", & ->data);
 }
  ->next = NULL;
  return (Head);
 }
```

```
struct Node* insert (struct Node *head int A)
{
  int H =0;
  struct Node*  , * temp;
   C=head;
   temp= (struct Node*)malloc (Size of (struct Node));
    while (H! =n)
     {
      c= C->next
      H++;
     if (H ==n)
  {  Printf(" Enter the elements you want to enter");
    scanf (" %d", &temp->data);
     temp ->next = c->next;
      c->next =temp;
     }
      }
       return (head);
  }
   struct Node * delete (struct Node *head, int n)
  {
    int H=0;
    struct Node* c, * temp;
      c = head;
     while (H! = n-1)
   {
     c= C->next;
     H++
     if (H ==n-1)
{
    e->next = (e->next)->next;
    }
  }
   return chead);
}
```

→ Construct a new linked list by merging a

[illegible] in list 1 [illegible]

Output:

```
Enter the no: of elements to be entered: 4
Enter an 0th element 1
Enter an 1th element 2
Enter an 2th Element 3
Enter an 3th Element 4
Node data 1
Node data 2
Node data 3
Node data 4
Enter the index ~~where~~ where you want to Enter 1
Enter the Element you want to Enter 44
Node data 1
Node data 2
Node data 44
Node data 3
Node data 4.
Node data 1
Node data 2
Node data 44
Node data 4.
```

```
return (head);
}
```

2) Construct a new linked list by merging alternate nodes of two lists for example in list 1 we have {1,2,3} and in list 2 we have {4,5,6} in the new list we should have {1,4,2,5,3,6}

```
# 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 Print list (struct Node * head)
  {      struct Node* temp = head
         While (temp != NULL)
         {
```

```
struct Node* insert (struct Node *head int n)
{
  int H = 0;
  struct Node*  , * temp;
   C = head;
   temp = (struct Node*) malloc (Size of (struct Node));
    while (H! = n)
     {
       C = C->next
       H++;
      if (H = = n)
   {  Printf ("Enter the elements you want to Enter");
     scanf ("%d", &temp->data);
      temp ->next = c->next;
       c ->next = temp;
     }
      }
       return (head);
  }
    struct Node * delete (struct Node * head, int n)
  {
        int H=0;
        struct Node* c, * temp;
          c = head;
        while (H! = n-1)
   {
        C = C->next;
        H++
        if (H = = n-1)
        Printf ("%d", temp->data);
         temp = temp->next;
      }
      Printf ("\n");
   }

Void merge (struct Node *B, struct Node **c)
{
      struct Node *B_curr = B, *c-curr = *c;
      struct Node * B-next , *c-next;
   while (B-curr != NULL && c-curr != NULL)

         B-next  = B-curr->next;
         c-next  = c-curr->next;

         c-curr->next  = B-next;
         B-curr->next = c-curr;

          B-curr = B-next;
          c-curr = c-next;

   }
      *c = c-curr

int main ()
      struct Node * B = NULL, *c = NULL;
        Push (&B, 3);
        Push (&B, 2);
         Push (&B, 1);
  cout<<printf (" First linked list:\n");
        Print list (B);
```

```
Push (& C, 8);
Push (& C, 7);
Push (& C, 6);
Push (& C, 5);
 Push (& C, 4);
cout<<
~~Printf~~ (" second linked list :\n");
Print list (q);
merge (B, & C);
Cout << " Modified first linked list : \n"
 Print list (B);
 Cout<< " Modified second linked list :\n"
  Print list (C);
  return 0;
}
```

2) output:

```
  first linked list:
   1 2 3
second linked list:
  4 5 6 7 8
Modified first linked list
 1 4 2 5 3 6
 Modified second linked list
  7 8
```

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

```
# include <stdio.h>
int a[10], top1 = -1, b[10], top2 = -1;
int a empty ()
{
    if (top1 == -1)
        return 1;
    else
        return 0;
} int s1 top()
{
    return s1 [top1];
}
 int s1 POP ()
 {     top1--;
 }
int a Push (int x)
 {     a [++ top1] = x
 } int s2 Empty ()
 }  if (top2 == -1)
        return 1;
 else
    return 0;
} int b top ()
 {  return b [top2];
 }
      int b POP ()
  {  top2--;
```

```
}
  int ~~a~~ bPush.(int x)
{
  b~~a~~ [++ top2]= x;
}
   int sum [int k]
{
   int x;
 while ( a Empty()!=1)
{
    x = a ~~s~~top();
   a POP();
 while (a Empty ()!=1)
 {
     if (x+a top()=k)
   {
     Printf (%d, %d)\n", x, a top();
   }
     b Push (a top());
       a POP ();
   }
   while( b empty()!=1
   {
       a Push (b top());
         b POP();
    }
 }
}
  int main ()
{
   int n, i, e, k;
   Printf (" Enter the no: of elements to stack:");
   Scanf ("%d", &n);
for (i=0; i<n; i++)
{
 scanf ("%d", &e);
   a Push(e);
}
Printf ("Enter the value of Constant sum:\n");
scanf ("%d", & k);
Printf (" The Combination whose sum equal to k is:
          \n");
   sum(k);
}
```

7) write a Program to print the element in a queue
 (i) in reverse order
 (ii). in Alterante order.

```
(i) # include <stdio.h>
    # include "stack.h"
    # include. "Q.h"
    int main ()
    {   int n, arr[20], m, k=0;
      struct stack s;
      int stack (& s);
      Printf ("Enter no");
      scanf ("%d", &n);
      for (m=0, m<n, m++)
      {
         Printf (" enter values:");
         scanf ("%d", & arr [m]);
      }
      for (m=0; m<n; m++)
```

Scanned with CamScanner

```
  {
   insert (arr[m]);
  }
   while (K!=n)
  {
     Push (&s, del(s));
     K++;
  }
     Printf ("Reverse is");
    while (stop!=-1)
   {
     Print f ("%d", PoP(&s));
   }
     Printf("\n");
   return0;
}
(ii)  # include <stdio.h>
  # include <stdblio.h>
  struct node {
    int data;
   struct Node * next;
   }
   Void  Printtnodes(struct Node*head)
   {  int Count=0;
     while (head! = NULL) {
       if (Count %2==0) {
         Printf("%d", head->data);
       }
        Count++;
        head = head -> next;
   }
  }
```

```
void Push (struct Node ** head - ref ; int new - data)
 {
  struct node* new_node = (struct node*) malloc.
                     (size of (struct node));
  new - node -> data = new-data ;
  new - node -> next = (*head - ref);
   (* head - ref) = new - node;
 }
 int main ()
{
 struct node * head = NULL;
       Push (& head, 6);
       Push (& head, 12);
       Push (& head, 22);
       Push (& head, 8);
       Push (&head , 9);
      Print node (head);
 return0;
}
```

5) (i) How many array is different from the linked list

(ii) write a Programme to add the first element of one list to another list for example we have {1,2,3} in list 1 and {4,5,6} in list2 we have to get {4,1,2,3} as output for list 1 & {5,6} for list 2.

(i) The difference between Array and linkedlist is structures. Arrays are index based data. structure where each element is Associated with an Index. on the other hand, linked list ~~to~~ lies on Previous and next elements.

(ii)
```
#include <stdio.h>
#include <stdlib.h>
int len (int a[])
{
   int i=0, an=0;
   while(1)
  {  if (a [i])
      {
        an++, i++
      }
     else
      { break;
      }
  }
  return an;
} void changing list (int a[] ~~b~~ int b[])
{ for (int i= len(a)-1; i>=0; i--)
    {
      a [i+1] = a[i];
    }
      a[0] = b[0];
  Print f ("\n the elements of first array:\n");
     for (int i=0; i < len(a); i++)
     { Print f ("%d", a[i]);
}
  for (int i=0;i<len(b); i++)
  {  b[i] = b[i+1];
  }
 Printf ("\n the elements of second array:\n");
  for (int i =0; i<len(b) ;i++)
   {  Printf ("%d", b[i]);
   }
}
int main()
{
  int a [10] = {1,2,3}, b[10] = {4,5,6};
 changing list (a,b);
}.
```

output:

```
The elements of first array;
4123
the elements of second array:
56.
```