D.S.A ASSIGNMENT

K.Vivek

AP19110010413

CSE-F

1] write a program to insert and delete at the nth and kth element at the nth and kth are taken from the users

A)
```
# include <studio.h>
# include <studio.h>
   struct Node {
      int data {
  struct Node* next;
  };
  struct node * head;
   Void insert(int data, int r){
   Node * temp = new node ();
   temp → data = data;
   temp → next = null;
     if(n = 1){
   Temp → next = null;
   If (n == 1){
   Temp → next = head;
     head = temp;
       return;
     }
     Void delet(int k){
```

```
struct Node *temp= headj
  if (k==1){
 head =temp→next
   fro (temp);
    return;
 }
   Node *temp= headj
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;
   ~~temp→next=temp~~
    free (temp);
  }
    int main () {
    int n,x,k;
 head =null;
printf ("Enter the position for inserting");
   scanf("%d", &n);
   scanf("%d", &x);
     Insert (x,n);
```

```
Prinf("enter the Prition todelete");
   Scanf ("%d", &k);
   delete(k);
    Print (x)
    ~~return~~ return;
    }
2) # include<stdio.h>
   # includ <stio.h>
  struct node{
    int data;
   struct node next;
  }
   void Print list(struct node*head)
  {
   Print f ("%d→", (Ptr→data));
      Ptr=Ptr→next
       Print f ("Null\n");
     }
  void Push(struct node*head; int data)
   {
   struct node*new=(struct node) malloe
                 (size of struct node);
    new→data=data;
    new→newt=*head;
     *head=new;
     }
```

```
struct node*merge(struct node* a,struct
node b);
{
  struct node fake;
  struct node *tail=fake;
   fake.next=Null;
    while(1){
   if(a==null)
   {
   tail->next=a;
    break;
  }
    else;
  {
    tail->next =a;
   tail= a;
   a=a->next;
   tail->next=b;
  }
  }
   return fakenext;
  }
    void main()
  {
    int keys[]={1,2,3,4,5,6,7}
 int n=size of(keys)/size of key[a]
  struct node *a=Null; *b=Null;
   for int(i=n-1,i>0;i=i-a)
       push(&a,keys[i]);
    for(int i=x-2;i>=0,i=i-2)
       push(&b;key[i]);
```

```
struct node * head= merge (a,b)
 print(head);
 }

3] #include<stdio.h>
 int top=-1j
 int x
 char stack [100];
 void push (int x);
  char pop()
  int main ()
 {
  int i, n, a,t, k,f, sum=0, count =1;
  printf ("Enter the number of elements
             with in the stack")j
   scanf ("%d", &a);
    push (a);
  }
  Print f ("Enter the sum to be checked");
  scanf ("%d", &a);
    ~~Push (a)~~) for (i=0; i<=n; i++)
  {
  ~~printf ("Enter~~ the sum
    t = pop(i);
    sumt = t;
    count t= 1;
   If (sum == t){
   for(int j =0;j<count; j++)
    Printf("%d", stack[i]);
```

```
F=1
break
}
push(t);
}
if(f!=1))
printf("The elements in the stack don't add
        up to the sum");
}
    void push(intx)
{
   if(top==99)
 {
 printf("Instack is full\n");
  return;
  }
   top=top+1;
   stack[top]=x;
 }
   char pop()
  {
   If(stack[top]=x;
  }
    printf("In stack empty")
   return 0;
   }
   x=stack[top];
    top=top-1
     }
```

```
4] #include <stdio.h>
   #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 ("\n\n**menu**\n");
   Printf ("1.Insertion\n2.deletion3.reverse4.Alternate");
   Printf (" \n Enter your choice");
   scanf(" %d &choice);
  switch (choice){
 case1: Printf("enter the value to be insert");
  scanf("%d", &value);
  insert (value);
  break;
 case2: delete();
      break;
 case3 :
  Printf (" The reversed queue is");
  for list 1=size; i>=0; i--)
 {
   if queue [i] == 0]
   continue;
   Printf ("%d", queue(i]);
  }
```

```
break;
Case 4:
Printf("Alternate elements of queue)
  for(int i=0; i< SIZE; i+=2)
 {
   if (queue[i]==0)
   continue;
   printf("%d",queue[i]);
 }
    break;
Case 5: exit(0)
default: Printf("wrong selection");
  }
  }}
 void insert (int value){
 if((f==0&&r==(SIZE-1))||f==r+1)
   printf("\n Queue is full")
   else{
    if (f==-1)
   f=0;
      r= (r+i)% SIZE;
  queue[r] = value;
  Print f("\n insertion success");
 }}
  void delet() {
   if(f==-1)
```

```
else{
printf("\n Deleted: %d", queue[f]);
    f = (f+1)% SIZE;
    if(f==r)
   f = r = -1;
   }}
```

5) How array is different from the linked list?

sol The major difference b/w Array and the linked list regards to their structure. Array are index based data structures where elements associated with an index on the other hand, linked list relies on reference to previous and next element.

```
6) # include<stdio.h>
   # include <stdlist.h>
   struct node
 {
   int data;
   struct node *next
 };
 void push(struct node** header, int, new
            -data);
 {
 struct node* new-node =(struct node*)
                       malloc (sizeofstructure)
```

```
new_node → data = new_data;
new_node → next = (*head_ref);
 (*hed_ref) - new_node;
}
void printlist(struct node *head)
{
  struct node * node * temp = head;
   while (temp != Null)
  {
    Printf("%d", temp → data;
    temp = temp → next;
  }
   print ("\n");
  }
```