1) write a program to insert and delete an element at the n^{th} and k^{th} pointer in a linked list where n and K are taken from the users?

Siva Sai. Tujjuru
AP21100105417
CSE-H

A)
```
#include <stdio.h>
#include <stdlib.h>
struct Node {
int data;
struct Node * Next ;
};
struct Node * head ;
Void Insert (int data, int n) {
Node *temp = Newnode ( ) ;
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, x, K;
   head = Null;
   printf[" Enter the Positia for inserting");
   Scanf["%d", &n):
   scanf ["%d", &K);
  Insert (x, n);
  printf ("Enter the position to delete");
     scanf ("%d", & K);
     Delete(K);
     Print(x);
   return;
}
```

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

A)
```
#Include<stdio.h>
#Include<stdlib.h>
Struct node{
int data;
Struct node * link;
}*head1=Null,*temp,*tem1,*head2=Null,*head3=Null;
struct node *insert(struct node *head,int x)
{
 temp=(struct node*)malloc(sizeof(structnode));
 temp→data=x;
 temp→link=Null
 if(head==Null)
 {
   head=temp;
 }
 Else
  {
     temp1=head
     while(temp1→link!=NULL)
     {
       temp1=temp1→link;
     }
     temp1→link=temp;
  }
   return head;
}
int main()
{
 int $,@,yx,i;
 Printf("Enter the no.of Elements of first linked list");
```

```
scanf("%d", &s);
for(i=0; i<s; i++)
{
   printf("Enter the element");
   scanf("%d", &y)
   head1 = insert(head1, y);
}
 printf("Enter the no.of elements of second list");
 scanf("%d", &n);
 for(i=0; i<n; i++)
 {
  printf("Enter the Element");
  scanf("%d", &y);
  head2 = insert(head2, y);
 }
  temp = head1;
  temp1 = head2;
  while(temp != Null && temp1 != Null)
  {
   printf("%d", temp->data);
   printf("%d", temp1->data);
   temp = temp->link;
   temp1 = temp1->link;
  }
  while(temp != Null)
  {
   printf("%d", temp->data);
    temp = temp->link;
  }
  while(temp1 != Null)
  {
   printf("%d", temp1->data);
   temp1 = temp1->link;
  }
}
```

Input:-

In list 1 -

1
2
3

In list 2 - 3

4
5
6

Output:-

New list is

{1,4,2,5,3,6}

3) Find all the elements in the stack whose sum is equal to k, where k is given from user?

A)

```
# include<stdio.h>
int S1[10], top1 = -1, S2[10], top2 = -1
int s1Empty()
{
   if (top1 == -1)
       return 1;
   else
       return 0;
}
int stop()
{
   return stop[1];
}
int s1push(int x)
{
   S1[++top1] = x;
}
int S2Empty()
{
   if (top2 == -1)
       return 1;
   else
       return 0;
}
int S2top()
{
   return S2[top2];
}
int S2pop()
{
   top2--;
}
int S2push(int x)
{
   S2[++top2] = x;
}
```

```
int sum (int K)
{
  int n;
  while (S1.Empty()!= 1)
  {
    x = S1.top()
    S1.pop()
    while (S1.Empty()!=1)
    {
      if (n+S1.top()== K)
      {
        printf("%d,%d", n, S1.top());
      }
      S2.push(S1.top());
      S1.Pop();
    }
    while(S2.Empty()!=1)
    {
      S1.push(S2.top());
    }
  }
}
int main()
{
  int n,i,e k;
  printf("Enter the no.of Elements of stack:");
  Scanf("%d",&n);
  for (i=0; i<n;i++){
  scanf("%d',&e);
  S1.Push(e);
  }
  printf("Enter the value of constant sum \n");
  scanf("%d",&K);
  printf("The combinations whose sum is Equal to K");
  Sum(K);
}
```

Input

Enter the no.of Ele is stack-4

Enter values - 1
2
3
4

Enter the value of constant sum
K-5

Outputs :-

The combinations sum is equal to K is

(2, 3)

(4, 1)

4) Write a program to print a Elements in a Queue in reverse order?

A)
```
#include<stdio.h>
#define SIZE 10
 Void insert(int);
 Void delete();
 int queue[10], p=-1, s=-1;
Void main(){
 Int value, choice;
 while(1){
  Printf("menu");
  Printf["1. Insertion\n 2. deletion\n 3. print Reverse\n 4. Print Alternate\n
                5. Exit");
  printf("\n Enter your choice:");
  Scanf("%d", &choice);
  switch(choice){
  case 1: printf("Enter the value to be inserted:");
   scanf["%d", &value);
   insert(value);
   break;
   case 2: delete();
   break;
  case 3: printf("print reverse");
   for(int i=SIZE; i>=0; i--)
  {
    if(queue[i]==0)
    continue;
     printf("%d", queue[i]);
   }
   break;
```

```
case4: 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:print("\n wrong selection !!! Try again !!!");
      }
}
Void insert(int value){
  if ( p==0 && S==SIZE-1) || p==S+1)
  Print f("Que is. full!!! Insertion not possible");
  else {
   if(p==-1)
  p=0;
   S=(S+1)%SIZE;
   queue[s]=value
   printf("\n Insertion sucess");
   }
}
 void delete(){
   if(p==-1)
   Printf("\n Queue is Empty!!! deletion is not possible!!!");
  else{
   printf("\n Deleted: %d", queue[P]);
   P=(P+1)% SIZE;
   if(P==S)
   P=S=-1;
  }
}
```

⑤

(i) Difference between Array and linked list is the major difference b/w Array and linked list is regars to their structure. Arrays are index based data structure where each element associated with an index. On the other hand, Linked list relies on reference where each node consists of the data and the reference to the previous and next element.

* The size is fixed in array but the varies in the linked list

* The memory allocation in the Array is continous and It is Random in the linked list

* The memory utilization in an ARRAY is inefficient and where as it is efficient in linked list.

* In Array Insertion and deletion is not easy, but it is easy in linked list.

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

Prog ra:-

```
#include <stdio.h>
#include<stdlib.h>
struct node {
    int data;
    structnode * link;
   } *head1=Null, *temp, *temp, *head2 = Null;
    struct node *insert (struct node *head, int n)
{
  temp = (struct node *) mall (size of (structnode));
  temp=>data = n;
  temp -> link = Null;
  if (head == Null)
  {  head = temp;
   }
   else
{
  temp1 = head;
  while (temp1 ->link != Null)
   {
     temp1 = temp1 -> link;
    }
    temp1 -> link = temp;
  } return head
}
int main(){
  int p, q, n, i;
  printf ("Enter the elements of first linked list");
  scanf ("%d", &p);
  for (i=0; i<p; i++);
   {
```

```
printf("Enter the no.of elements in second linked list");
scanf("%d",&q);
for(i=0;i<q;i++)
{
   printf("Enter the element in second list");
   scanf("%d",&q);
   for(i=0;i<q;i++){
    printf("Enter the element");
    scanf("%d",&n);
    head2 = insert(head2,n);
3
temp = (struct node *)malloc(sizeof(struct node));
temp->link = head1;
temp->data = head2->data;
head1 = temp;
head2 = head2->link;
temp1 = head1;
while(temp1 != Null)
{
  printf("%d", temp1->data);
   temp1 = temp1->link;
 }
  printf("\n linked list 2\n");
   temp1 = head2;
   while(temp1 != NULL)
   {
    printf("%d", temp1->data);
     temp1 = temp1->link;
   }
}
```

Input :-

Enter the no. of Elements of first linked list - 3

1
2
3

Enter the no. of elements of second linked list - 2

5
6

Output :-

Elements of linked list-1 ,

('4, 1, 2, 3)

Elements of linked list 2 ,

(5, 6)