ASSIGNMENT-4

P Naveen
AP19110010465
CSE-H

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

```
#include <stdio.h>
#include <stdlib.h>
int size = 0;
struct node {
int data;
struct node * next;
};
node * get node (int data)
{
node * new node = (struct node*) malloc (new node);
new node → data = data;
new node → next = null;
return new node;
} 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 delete to be insertion ");
 Scanf ("%d", &n);
 Sfanf ("%d", &x);
 Snsert (x,n);
 printf ("Enter to delete the position");
 Scanf ("%d", &k);
  delete (k);
   printf (x);
   return 0;
 }
```

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

```
#include <stdio.h>
#include <stdlib.h>
struct node {
int data;
struct node * next ;
}
void print list (struct node * head)
{
printf ("%d→", (ptr → data) ;
ptr = ptr → next ; }
printf (" Null \n") ;
}
void push (struct node * head, int data)
{
struct node * new node = (struct node * ) malloc (sizeof
                         (struct node)) ;
new → data = data ;
new → next * head ;
   * head = new node ;
}
struct node * merge (struct node * a) ; (struct node * b)
{
  struct node fake ;
struct node * fail = fake ;
  fake = next = null ;
```

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

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

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

```
#include <stdio.h>
int top = -1;
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 in stack");
Scanf("%d",&n);
for (i=0, i<=n;i++) {
printf("Enter next elements");
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(f!=1)
printf("Enter the elements in the stack dont add
upto the sum");
}
 void push(int x)
 {
 if (top == 99)
 {
  printf ("\n stack is full !!!\n");
  return;
 }
 top = top+1;
 stack [top] = x;
 }
 char pop()
 {
  if (stack [top] == -1)
 {printf("\n stack is EMPTY !!!\n");
  return 0;
 }
  x = stack [top];
  top = top - 1
  return x;
```

4) Write a program to print the elements in a queue

1) In reverse order

2) In alternate 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("\n\n*** MENU ***\n");
    printf("1. Insertion\n2. Deletion\n3
    print Reverse\n4. print alternate\n5. Exit");
    printf("Enter your choice.");
    scanf("%d", &choice);
    switch(choice){
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 to 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("\n Wrong selection !!! Try again!!!");
}
}}
void Insert (int value) {
if ((f==0 && r == size-1) || f==r+1)
printf("\n queue is full !!!! Insertion is not
possible !!!");
else {
if (f==-1)
f=0;
r=(r+1) % size;
```

```
queue[r] = value
printf("\n Insertion success!!!");
}

void delete() {
  if (f == -1)
  printf("\n queue is Empty!!! deletion is not
  possible!!!");
   else {
    printf("\n deleted : %d", queue[f]);
  f = (f+1) % size
  if (f == r)
  f = r = -1;
  }}
```

5) 1) How array is different from the linked list

A) The major difference b/w array and linked list regards to their structure. Arrays are index based data structure. where each element associated with an index, on the other hand linked list relies on references to their previous and next element.

5) 2) write a program 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 list 2 we have to get {4,1,2,3} as output for list 1 and {5,6} for list 2.

```
#include <stdio.h>
#include <stdlib.h>
 int len (int a())
 {
  int i=0, x, y=0;
  while (1)
  {
  if (x[i])
  {
   xy++, i++;
 }
  else
  {
   break;
  }
 }
  return xy;
}
 void change list (int x[], int a[])
{
```

```
for(int i = len(x) - 1 ; i>0; i--)
{
  x[i+1] = x[i] ;
}
x[0] = a[0];
printf (" \n elements of old array : \n");
for (int i=0; i<len(x); i++)
{
  printf (" %d", x[i]);
}
for (int i=0; i<len(y) , i++)
{
  y[i] = y[i+1] ; }
printf(" \n elements of new array : \n");
for (int i=0; i<len() ; i++)
{
  printf (" %d", a[i]);
}
int main()
{
  int x[10] = {1,2,3}, a[10] = {4,5,6};
  change list = (a,b);
}
```