AP19110010406 Assignment-4

CSE-H K. Janardhan

① write a 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.

Sol:

```
#include <stdio.h>
#include <stdlib.h>
void ans(node *, int, int)
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;
}
void ins( node * current, int pos, int data)
{
  if (pos<1 // pos> size+1)
  printf ("Invalid");
  else
 {  while (pos--)
    {
     if (pos==0)
     {
     node * temp= get node (data);
     temp-> next= *current;
```

```
        *current = temp;
        }
 else
 {
 current = &(*current)->next;
 }
size++;
}
}
void printf(struct node * head)
 {
   while (head != null)
   {
     printf ("%d", head -> data);
      head = head -> next;
      }
      printf("\n");
      }
void del(struct node * head, int pos){
if (head -uef == null)
 return;
 temp=head-uef
if (pos=0)
{
 *head uef= temp->next;
 free(temp);
 return;
for( int i=0; temp!= null && r< pos-1; i++)
   temp = temp->next;
   free (temp->next);
   temp -> next = next;
 }
 int main()
 {
struct node * head = Null;
 push (&head,7);
push(&head,8);
```

```
push(&head, 6);
ins(&head, 7, 1);
del(&head, 4);
print list(head);
return(0);
}
```

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 we should 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)
{
  struct node* ptr = head;
  while(ptr)
  {
    printf("%d->", ptr->data);
    ptr = ptr->next;}
    printf("Null\n");
  }
void push(struct node** head, int data)
{
struct node * new=(struct node*) malloc(size of(struct node));
new->data = data;
new->next = *head;
*head = new;
}
struct node * merge(struct node* a, struct node* b)
{
```

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

(3) find all the elements in the stack whose sum is equal to k (where k is given by the user).

```
#include <stdio.h>
void find(int arr[], int n, int s){
int sum=0;
int l=0, h=0;
for(l=0; l<n; l++){
  while(sum<s && h<n)
    sum += arr[h];
    h++;
if(sum==s)
{
  printf("found");
  return; }
  sum -= arr[l];
}}
int main(void){
  int arr[]={2,6,0,9,7,3}
  int s=15
  int n= size of(arr)/size of(arr[0]);
  find(arr,n,s);
  return 0;
}
```

(4) write a program to print the elements in a queue.
(i) reverse order (ii) in alternate order.

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
  int data;
  struct node * next;
}
```

```
void print rev(struct node * head)
{
    if (head = Null)
        return;
    print rev(head->next);
    Printf("%d", head->data);
void push(struct node * head rev, chan new)
{
 struct node* node_new = (struct node*) malloc (size of(struct node));
 node new->data = new;
 node_new -> next = (head*_ref);
 (* head_ref) = node_new;
}
 int main()
   struct node * head = Null;
   Push(& head, 4);
   Push(& head, 3);
   push(& head, 2);
   print new(head); print alternate (head);
   return 0;
 }
 void print alternate (struct node * head)
 {
    int count = 0;
    while (head != Null)
    {
        if (count % 2 == 0)
            count << head->data << " ".
        count++;
        head = head -> next;
    }
```

⑤ How array is different from the linked list.

① ① An array is a data structure that contains a collection of similar type data elements where as the linked list is considered as non primitive data structure contains a collection of unordered linked elements known as nodes.

② In the array the elements belong to indexes, if you want to get into the fourth element you have to write the variable name with its index of location within the square bracket.

③ In a linked list through, you have start from the head and work your way through until you get to the fourth element.

④ According an element in an array is fast access while in linked list take linear time.

⑤ operations like insertion and deletion in array consume a lot of time. on the other hand the performance of these operations in linked list is fast

⑤ ⑪
```
#include <stdio.h>
#include <stdlib.h>
int fn(int a[])
{
 int i=0,a,n=0;
  while(i)
 {
 if(a[i])
 {
   a n++ ,i++;
 }
  else
 {
   break;
 }
 }
 return an;
```

```
}
void changing list( int a[], int b[ ])
{
 for(int i= len(a)-1; i>0, i--)
 {
  a[i+1]= a[i];
 }
 a[0]= b[0];
 printf("\n the elements of first array: \n");
for( int i=0 ; i< len(a); i++)
 {
  printf("%d", a[i]);
 }
 for(int i=0, i< len(b); i++)
 {
  b[i]= b[i+1];}
  printf(" the elements of second array:");
  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);
 }
```