# Assignment - 4

M. Mani Krishna
Karth Reddy.
AP19110010115
CSE - G

1.) write a programme 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 uses.

A.)
```
# include <stdio.h>
 # include <stdlib.h>
 Struct Node {
  int data;
 Struct Node *next;
   };
 Struct Node * head;
 Void Insert (int data, intn) {
  Node * temp = new node ();
   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 (inti=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 ;
Print f ("enter the position for
         and inserting;");
Scanf ("%d", &n);
Scant ("%d", &x);
Insert (x,n);
Print f ("enter the position to delete);
Scant ("%d", &K);
Delete (K);
Print (x°)
return ;
}
```

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

```
A.)   # include < stdio.h >
      # include < std lib.h >

      Structnode {
      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. dabri)
  {
  struct. node *new = (struct nodet) malla
               (size of (structnode));
   new -> data = data;
   new -> next = * head;
     * head = new;
     }
```

```
structnode * merge (struct node * a, struct
                        node * b)
{ struct node dalce ;
  structnode * fail = fake;
  fake.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 fake.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-a)
Push (&a, keys [i]);
for (int i = n-2; i>=0; i = i-2)
   Push (&b; key [j]);
struct node * head = merge (a,b);
Printlist (head);
}
```

3] find all the elements in the stack whoose sum is equal to k

A.)

```
#include < stdio.h>
void find (int arr[ ], int a, intk) {
int total =0
int x=0, y=0;
```

```
for (x=0; x<a; x++){
  while (Sum<K, & & y<a)
           = arr [y];
      y++;
for (x=0; x<a; x++){
  while (total<K; & & y<a)
       total= arr [y];
       y++;
     if (total = =0)
     {
       print f ("find");
        return; }
     total - = arr [x];
    }
   int main (void){
  int arr[ ] = {9,10,12,4,1,2,3}
   int k= 565;
   int a= Size of (arr)/Size of(arr(0));
   find (arr, a, k);
   return 0;
   }
```

4.) write a programme to print elements of Queue?

i.) Reverse order    ii.) Alternate order

```
# include < stdio.h>
# define size 20
void insert (int);
void delete ( ) ;
int queue [20], a=-1, b=-1;
void main ( ) {
int num; choice;
while(1) {
Print f ("n" new"\n");
print f ("1.insert|n2. Delete|n3. point
n4. Reverse|n4. Alternate|n5. Exit);
Print f ("n enter your choice");
Scanf (K%d", &choice ");
Switch (choice) {
Case 1 : print f ("enter the num' to insert");
scanf ("%d", & num);
insert (num);
break ;

Case 2:,  Print f ("Reverse queue");
     for(int i = size, i>0; i--)
       if (queue[i]=0]
    Contince;
   Print f ("%dn", queue[i]);'
  }
    break ;

Case 3:-
   Print f ("Alternate elementsn");
       for (int i=0, P< size, i>0, i++2)
  {
   if (queue[i] = =0)
    Continue;
    Print f ("%d", queue[i]);
  }
      break;
   return 0;

   }
```

5) i) How array is different from linked list

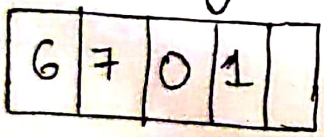
2) write a programme to add first element of one list to another list for example we have (1,2,3) in list 1 and (4,5,6) in list 2 we have to get (4,1,2,3) as output for list 1, and (5,6) list 2.

A) i) Arrays vs linked lists.

1. Both are the data structures Both are used to store the data.

2. Cost of accessing the elements

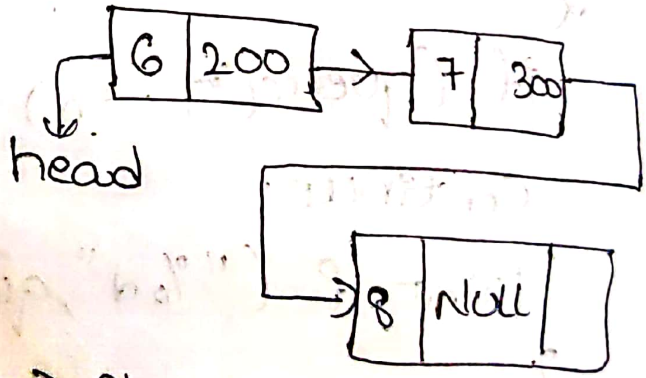
Arrays



=) it takes at constant time

$\boxed{O(1)}$

linked list



=) it depends on number of nodes in the linked list

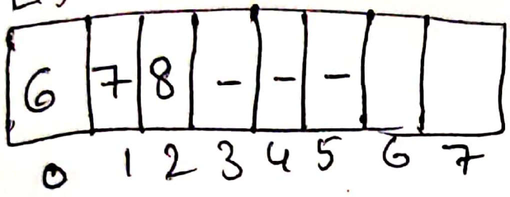
$\boxed{O(n)}$

## 3. memory Requirement and utilization

**Array**

=> Ineffective in memory utilization

Ex)



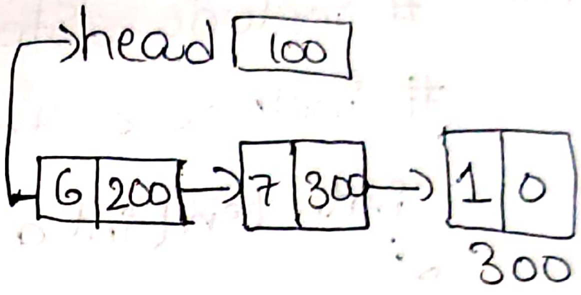
8×4=32 bytes

used=12

=> Require memory is less

**linked list**

=> it is in dynamic Size.



8×3=24 bytes

=> More requirement

## 4.) cost of Insertion and cost of deletion

| Array | | linked list |
|---|---|---|
| Begining - O(n) | — | O(1) |
| At end - O(1) | — | O(n) |
| ith position - O(n) | — | O(n) |

5. Easy Use and operations:-

| Array | linked lists |
|---|---|
| => easier to use | => less easier |
| => linear and Binary | => linear |

ii.)

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