LAB-1

Write a program to input an element to an array and display the biggest and smallest element using function.

```
#include <stdio.h>
void search (int * arr, int n)
{
    int i, small = arr[0], lar = arr[0];
   for (i= 1; i<n; i++)
    { if (smallest > arr[i])
        small = arr[i];
      if (lar < arr[i])
        larges  lar = arr[i];
    }
   printf ("smallest element : %d", small);
   printf ("\n largest element : %d, lar);
}
int main ()
{ int arr[20], n, i;
  printf ("Enter no. of digits :");
  scanf ("%d", &n);
  printf ("enter the number: \n");
```

```
for (i=0; i<n ; i++)
  { scanf (" %d", &arr[i]);
    }
  Search (arr, n);
    return 0;
}
```

Write a program to dynamically allocate memory for an array of n elements and display the second biggest and second smallest element using function.

```
#include<stdio.h>
#include<stdlib.h>
void func(int* arr, int n)
{
    int i, j;
for (i=0; i<n-i; i++)
  {
    for(j=0; j<n-i-1; j++)
    {
      if (arr[j] > arr[j+1])
    { int temp = arr[j];
      arr[j] = arr[j+1];
      arr[j+1] = temp;
      }
    }
}
printf("\n 2nd Smallest = %d", arr[1]);
printf("\n 2nd largest = %d", arr[n-2]);
}
```

```
int main ( )
{ ent n,i;
  printf ("enter the size:");
  scanf ("%d", &n);
  int * arr = (int*) malloc(n * sizeof (int));
  printf ("Enter array : \n");
  for (i=0; i<n ; i++)
    {  scanf ("%d",&arr[i]);
    }
    func (arr, n);
    return 0;
}
```

Write a program to populate an array with n number of random numbers and arrange them in decreasing order using random().

```
#include <stdio.h>
#include <stdlib.h>
void func (int * arr, int n)
{ int i, j;
    for (i=0; i<n-1 ; i++)
    {
      for (j=0; j<n-1-i ; j++)
      {
        if (arr[j] > arr[j+1])
        { int temp = arr[j];
          arr[j] = arr[j+1];
          arr[j+1] = temp;
        }
      }
    }

for (i=0 ; i<n ; i++)
  printf ("%d \t", arr[i]);
}
```

4) Write a program to insert an element at a particular position and delete an element from a particular position using functions.

```
#include <stdlib.h>
#include <stdio.h>
void delete (int *arr, int n, int pos)
{ int i;
  for (i=pos-1; i<n-1; i++)
  { arr[i] = arr[i+1]; }
    n--;
  for (i=0; i<n; i++)
      printf ("%d \t", arr[i]);
}
void insert (int * arr, int n, int pos)
{ int val;
 printf ("\n enter new value: ");
   scanf ("%d", &val);
   int i;
 for (i=n-1; i>=pos-1; i--)
 { arr [i+1] = arr[i]; }
    arr [pos] = val;
  for (i=0; i<=n; i++)
  {
```

```
printf ("%d \t", arr[i]);
    }
int main ()
{   int arr[20], n, i;
    printf ("\n enter no. of digits: ");
    printf ("%d", &n);
     printf ("Enter the digits: ");
     for (i=0; i<n; i++)
      { scanf ("%d", &arr[i]);
       } int del; int insert;
    printf ("Enter the element to be deleted: ");
     scanf ("%d", &del);
     printf ("Enter the element to be inserted: ");
    scanf ("%d", &insert);
     delete (arr, n, del);
     insert (arr, n, insert);
printf ("The array after insertion and deletion: ");
   for (i=0; i<n; i++)
     { printf ("%d", arr[i]);
       } return 0;
      }
```

5) Write a menu driven program to perform the following operation on an array &c.
1. generate the random numbers. 2.) Display the elements 3.) Insert a new element at the begenning. 4) Insert a new element at the end. 5.) insert a new element at the middle. 6.) exit

```
#include <stdio.h>
#include <stdlib.h>
  void ran (int a[], int n)
  { int i;
    for(i=0; i<n; i++)
     { a[i] = rand() % n;
     }
  }
  void display (int a[], int n)
  { int i;
    for(i=0; i<n; i++)
     { printf("%d \n", a[i]);
     }
  }
```

```
void insert_beg(int a[], int *n)
{ int i,p;
for (i=n; i>=1; i--)
  { a[i] = a[i-1];
  }
printf ("Enter no. to be inserted");
scanf ("%d", &p);
    a[0] = p;
   *m = *m + 1;
}
void insert_end (int a[], int *n)
{ int p;
printf("Enter no. to be inserted ...");
scanf ("%d", &p);
a[*m] = (int*) malloc (sizeof (int));
    a[*m] = p;
   *m = *m+1;
}
```

```
void insert_middle (int a[], int * n)
{ int p, num;
 printf ("Enter position to insert element...");
  scanf ("%d", &p);
a[*m] = (int*) malloc (sizeof (int));
    for (i=*n; i>p ; i--)
       a[i] = a[i-1];
       printf ("Enter no. of insert...");
        scanf ("%d", &num);
           a[p] = num;
          *n = *n + 1 ;
   }
  int main ()
{ int n, i, s=1, ch;
  printf ("Enter 1. generate random nos \n
          2. display \n3. insert at begin \n
          4. insert at end \n5. insert at middle \n
          6. exit");
  scanf ("%d", & ch);
     switch (ch)
{     case 1: ran (a,m);
            break;
      case 2: display (a, n); break;
```

```
case 3: insert_beg (a, &n);
        break;
case 4: insert_end (a, &n);
        break;
case 5: insert_middle (a, &n);
        break;
case 6: exit(1);
default: printf ("wrong choice");
}
printf ("Want to continue !!! press 1n");
scanf ("%d", &n);
}
return 0;
}
```

6) WAP to arrange the first half of the array in ascending order and second half in descending order.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{ int i,n, j;
printf("Enter numbers of terms....");
  scanf ("%d", &n);
int * a = (int *) malloc (sizeof(int));
for (i=0 ; i < n/2 ; i++)
  { for (j=0 ; j < (n/2)-i-1; j++)
    { if (a[j] > a[j+1])
      { int b = a[j];
        a[j] = a[j+1];
          a[j+1] = b;
    } }
  }
for(i=n/2; i<n ; i++)
  { for (i=n/2; j<n-i-1 ; j++)
      {
```

```
if (a[j] < a[j+1])
{
    int t = a[j];
    a[j] = a[j+1];
    a[j+1] = t;
}
}
}
printf("The new array is... \n");
for(i=0; i<n; i++)
    printf("%d \n", a[i]);
    return 0;
}
```

7) Write a program to delete all the duplicate elements from an array.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{ int i, j, k, n;
printf ("Enter the number of elements...");
scanf ("%d", &n);
int *a = (int *) malloc (sizeof (int));
for (i=0; i<n; i++)
{ printf ("Enter a number");
  scanf ("%d", &a[i]);
}
printf ("Array with unique lists... \n");
for(i=0; i<n; i++)
{ for(j=i+1; j<n; j++)
  { if (a[j] == a[i])
    { for (k=j; j<m; k++)
      { a[k] = a[k+1]; }
      m--; }
```

```
        else
        j++;
        }
    }
    for (i=0; i<m; i++)
        printf ("%d", a[i]);
            return 0;
        }
```

9) Input the details of n no. of employees and display in tabular format -

```
#include <stdio.h>
struct emp
{ int empid;
  char pname[50];
};
int main ()
{ struct emp* p;
  int n;
printf ("Enter the no. of records ...");
scanf ("%d",&n);
p= (struct emp*) malloc (n*sizeof(struct emp));
for (i=0; i<n; i++)
  { printf ("Enter id and name respectively \n");
    scanf ("%d %d\n", p[i].name, p[i].empid);
  }
printf ("Displaying information ..... \n");
for (i=0; i<n; i++)
printf ("%s \t %d \n", p[i].name,
                       p[i].empid);
return 0;
}
```

Lab-2

1) Write a program to create a single linked list and display it.

```
#include <stdio.h>
#include <malloc.h>
void create (int a);
void display ();
struct node
{ int data;
  struct node* next;
};
struct node* start = NULL;
void create (int n)
{ struct node * temp;
  struct node * new = (struct node*) malloc (sizeof (struct node));

  new->data = n;
  if (start == NULL)
    { start = new;
    }
  else
{ temp = start;
```

```
while (temp->next != NULL)
{ temp = temp->next;
  }
 temp->next = new;
 }
 }
void display()
{ struct node * temp;
   temp = start;
   while (temp != NULL)
  { printf ("%d \n", temp->data);
   temp = temp->next;
    } }

int main()
{ create (5);
   create (6);
   create (8);
   create (10);
  display ();
  return 0;
    }
```

Q2) Write a menu-driven program with following functions: ① insert at beginning of linked list.
② insert at end.
③ insert at any position.
④ delete from front
⑤ delete from last
⑥ delete from any position
⑦ display contents
⑧ exit.

```
#include <stdio.h>
#include <stdlib.h>
 Struct node
{   int data;
    Struct node * next;
};
struct node* insertHead (struct node * head)
{ int value;
  printf ("Enter the value to inserted at head:");
  scanf ("%d",&value);
  Struct node* newNode = (struct node*) malloc
                          (size of (struct node));
   newNode -> data = value;
   newNode -> next = head;
    head = newNode;
     return head;
}
```

```
struct node* insertEnd(struct node* head)
{ int value;
  printf("Enter the value to be inserted at
          end: ");
  scanf("%d", &value);
  struct node* newNode = (struct node*) malloc
                         (sizeof(struct node));
    newNode -> data = value;
    newNode -> next = head; NULL;
      head = newNode;
      return head;
  }
if (head == NULL)
{ head = newNode;
  return head;
 }
 struct node* temp = head;
 while (temp -> next != NULL)
  temp = temp -> next;
    temp -> next = newNode;
      return head;
 }
```

```
struct node * insertAtAny (struct node * head)
{ int value, pos;
  printf ("Enter the value to be inserted : ");
    scanf ("%d", &value);
    printf ("Enter the position at which to insert");
    scanf ("%d", &pos);
    struct node * newNode = (struct node *) malloc
                            (sizeof (struct node));

    newNode -> data = value;
    newNode -> next = NULL;
   struct node * temp = head;
   int i;
   if (pr == 0)
   {
      newNode -> next = head;
        head = newNode;
         return head;
   }
  for (i=0; i<pos; i++)
    temp = temp -> next;
    newNode -> next = temp -> next;
    temp -> next = newNode; return head;
  }
struct node * delfront (struct node * head)
{ struct node * temp = head;
    head = head -> next;
      free (temp);
      return head;
}
```

```
struct node* dellast(struct node * head)
{
  struct node* temp = head;
  while (temp -> next -> next != NULL)
   temp = temp -> next;
  struct node* temp2 = temp -> next;
    temp -> next = NULL;
    free (temp2);
    return head;
}
struct Node * deleteAtAny (struct node* head)
{ int pos;
  struct node * temp = head;
  printf("Enter the position to be deleted: ");
    scanf ("%d", &pos);
    if (pos == 0)
  { struct node * temp = head;
      head = head -> next;
      free (temp);
      return head;
  } int i;
  for (i = 0; i < pos-1; i++)
      temp = temp -> next;
   struct node* temp2 = temp -> next;
    temp -> next = temp2 -> next;
     free (temp2);
     return head;
  }
```

```
void display (struct node * head)
{
  while (head != NULL)
  { printf ("%d", head -> data);
    head = head -> next;
  }
  printf ("\n");
}
int main()
{ struct node *head = NULL; int ch;
  printf ("1. Insert at head \n 2. Insert at end \n 3.
  Insert at any pos \n 4. Delete from front \n 5.
  Delete from last \n 6. Delete from any position
  \n 7. Display \n 8. Exit \n");
  printf ("Enter your choice:");
     scanf ("%d", &ch);
     switch (ch)
  { case 1: head = insertHead (head);
            break;
    case 2: head = insertEnd (head);
            break;
    case 3: head = insertAtAny (head);
            break;
    case 4: head = delfront (head);
            break;
    case 5: head = delLast (head);
            break;
```

```
case 6: head = delAtAny (head);
        break;

case 7: display (head);
        break;
 case 8 : break;
}}
while(ch != 8)
  return 0;
  }
```

Assignment-3

1) Write a menu driven program to perform the following double linked list program.

a) create
b) display
c) Insert at any point
d) Delete at any point

```
#include<stdio.h>
void create(int n);
void display();
void Insert(int n, int x);
void Delete(int x);

void create(int n)
{   struct Node
    { int data;
      struct Node *next;
      struct Node *prev;
    }*start = NULL;
```

```
void create (int n)
{
  struct Node* new = (struct Node *) malloc (sizeof (struct Node));

      new -> data = n;
      new -> prev = NULL;
      new -> next = NULL;

   if (start == NULL)
     {  start = new;
      }
  else
{  struct Node* temp;
      temp = start;
    while (temp -> next != NULL)
     {  temp = temp -> next;
      }
    temp -> next = new;
     new -> prev = temp;
  }
void display ()
{  struct Node* ~~start~~ temp;
    temp = start;
   while (temp != NULL)
   {  printf ("%d\n", temp -> data);
       temp = temp -> next;  }
```

```
int main ()
{ int n ; int *x ;
printf ("Enter 1 for create \n 2 for display \n
        3 for Insert at any point \n 4 Delete at any
        pt.\n");
  c;
Switch (n)
{
Case 1:
{ printf("Enter element \n"); int k;
        scanf ("%d",&k);
            create (k);
      break;
}
Case 2:
{ display ();
    break;
}
Case 3:
{ int a,b;
    printf ("Enter element and position\n");
      scanf ("%d,%d",&a,&b);
       insert (a,b);
       break; }
```

```
Case 4
{
  int a;
  printf("Enter position \n");
   scanf ("%d", &a);
    delete (a);
 break;
}
default ,
{  printf ("Enter correct choice \n");
    }
}
goto c; printf ("Enter more choice? \n");
        scanf ("%d", &x);
        if (x == 1)
        { goto c;
          }
        else
      return 0;
      }
```