[*] update.cpp

```c
//updating na array
#include<stdio.h>
int main()
 {
   int arr[30], element, num, i, location;

   printf("\nEnter no of elements :");
   scanf("%d", &num);
   for (i = 0; i < num; i++) {
      scanf("%d", &arr[i]);
   }
   printf("\nEnter the element to be inserted :");
   scanf("%d", &element);

   printf("\nEnter the location");
   scanf("%d", &location);
   //Create space at the specified location
   for (i = num; i >= location; i--) {
      arr[i] = arr[i - 1];
   }
   num++;
   arr[location - 1] = element;
   //Print out the result of insertion
   for (i = 0; i < num; i++)
      printf("%d", arr[i]);

   return (0);
 }
```

] lift shifting array in dt.c

```c
//left shift na AARAY
#include<stdio.h>
int main()
{
    int A[5],i;
    printf("ente the elemtas in an array");
    for(i=0;i<5;i++)
    {
        scanf("%d",&A[i]);
    }

    int temp=A[5];
    for(i=5;i<0;i--)
    {
        A[i]=A[i-1];
        A[0]=temp;
    }
    for(i=0;i<5;i++)
    {

    printf("%d",A[i]);
}
}
```

search elem. in list.cpp

```
//create display nad search element in linked list
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
struct node
{
    int data;
    node *next;
};
struct node *head=NULL;
void create()
{
    char o;
struct node * tail=NULL,*ptr;
        do{
        ptr=(node*)malloc(sizeof(node));
        if(tail==NULL)
        {
            head=ptr;tail=ptr;
        }
        else
        {
            tail->next=ptr;
            tail=ptr;
            tail->next=NULL;
        }
        printf("enter no\n");
        scanf("%d",&ptr->data);
```

Compile Log | Debug | Find Results

search elem. in list.cpp

```
        scanf("%d",&ptr->data);
        printf("press y for next node\n");
        o=getch();
        }
        while(o=='y');
}

void display()
{node * temp;
temp=head;
    printf("the list is\n");
    while(temp!=NULL)
{
    printf("%d",temp->data);
    temp=temp->next;
    }
}

void search()
{node *temp;
temp=head;
    int tem,count=0;
    printf("\nEnter the no. to find=");
    scanf("%d",&tem);


    while(temp!=NULL)
    {
```

search elem. in list.cpp

```
    int tem,count=0;
    printf("\nEnter the no. to find=");
    scanf("%d",&tem);


    while(temp!=NULL)
    {
    if(temp->data==tem)
    {
        temp=temp->next;
    ++count;
        printf("\nelement found in node %d",count);

        }
    else
    {   temp=temp->next;
        ++count;
        printf("\nelement not found in node %d",count);
        }
    }
}

main()
{
    create();
    display();
    search();
}
```

```c
#include<stdio.h>
int main()
{
    int A[5],i;
    printf("enter aray");
    for(i=0;i<5;i++)
    {
        scanf("%d",&A[i]);
    }
    int temp=A[4];
    for(i=4;i>=0;i--)
    {
        A[i]=A[i-1];
    }
    A[0]=temp;
    for(i=0;i<5;i++)
    {
        printf("%d",A[i]);
    }
}
```

revrse an aaray.cpp

```
//reverse an array
#include<stdio.h>
int main()
{
    int A[5],AR[5],ARR[5],i;
    printf("enter the elements in 1 array");
    for(i=0;i<5;i++)
    {
        scanf("%d",&A[i]);
    }
    for(i=4;i>=0;i--)
    {
        printf("%d ",A[i]);
    }
}
```

[*] queue1.cpp

```
//progamr for queue
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
int q[5],front=-1,rare=-1;
void enqueue()
{
    if(front==-1)
    {
        front++;rare++;
    }
    else
    {
        printf("enter value ");
        scanf("%d",&q[rare]);
        rare++;
    }
    if(rare>4)
    {printf("queue is full");
    return;
    }
}
void dequeue()
{
    if(rare==-1)
     printf("queue is empty");
     else
        printf("element removed is%d/n",q[front]);
```

```
27          else
28              printf("element removed is%d/n",q[front]);
29              front++;
30
31              }
32
33  void view()
34  {
35      int i;
36      if(rare==-1)
37      printf("queue is empty ");
38      else
39      {
40      {
41      for(i=front;i<rare;i++)
42          printf("%d ",q[i]);
43      }
44  }
45  }
46  main()
47  {
48      int no;
49      while(1)
50      {
51          printf("enter 1 to insert\nenter 2 to delete\nenter 3 to view \n");
52          scanf("%d",&no);
53          switch(no)
54          {
55              case 1:enqueue();
```

Compile Log | Debug | Find Results

```
    else
    {
    {
    for(i=front;i<rare;i++)
        printf("%d ",q[i]);
    }
}
}
main()
{
    int no;
    while(1)
    {
        printf("enter 1 to insert\nenter 2 to delete\nenter 3 to view \n");
        scanf("%d",&no);
        switch(no)
        {
            case 1:enqueue();
            break;
            case 2: dequeue();
            break;
            case 3:view();
        }

    }


}
```

```c
//DOUBLE LINKED LIOST
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
struct node
{
    int data;
    struct node*next,*prev;
};
struct node *head=NULL;
create()
{char opt;struct node *ptr,*temp;
    do
    {
        ptr=(struct node*)malloc(sizeof(struct node));
        if(head==NULL)
        {
            ptr->next=NULL;
            ptr->prev=NULL;
            printf("enter the data");
            scanf("%d",&ptr->data);
            printf("press y to continue");
        }
        else
        {
            temp=(struct node*)malloc(sizeof(struct node));
            printf("enter the data");
            scanf("%d",&temp->data);
            temp->next=NULL;
```

Compile Log | Debug | Find Results

double linkes.cpp

```
        else
        {
            temp=(struct node*)malloc(sizeof(struct node));
            printf("enter the data");
            scanf("%d",&temp->data);
            temp->next=NULL;
            ptr->next=temp;
            temp->prev=ptr;
            ptr=temp;
            printf("press y to cin");
            opt=getchar();
        }
    }
    while(opt=='y');
}
void insert()
{
    struct node *temp,*ptr;
    temp->next=ptr->next;
    ptr->next->prev=temp;
    ptr->next=temp;
    temp->prev=ptr;
}
int main()
{
    create();
    insert();
}
```

[*] stack.cpp

```
//STACK
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
int top=-1,stack[5],i;
void push()
{ if(top<4)
    {
    top++;
    printf("enter value ");
    scanf("%d",&stack[top]) ;
 }
 }
void pop()
{

    printf("element deleted is%d",stack[top]);
    top--;
 }
void view()
{
    if(top==-1)
    printf("stack is empty");
    else

{
    printf("stack is ");
    for(i=0;i<=top;i++)
```

Compile Log Debug Find Results

[*] stack.cpp

```
23      printf("stack is empty");
24      else
25
26  {
27      printf("stack is ");
28      for(i=0;i<=top;i++)
29      printf("%d",stack[i]);
30  }
31
32  }
33  main()
34  {
35      int n;
36      while(1)
37      {
38          printf("\nEnter 1 to push\nEnter 2 to pop\nEnter 3 to vieew\n");
39          scanf("%d",&n);
40          switch(n)
41          {
42              case 1:push();
43              break;
44              case 2:pop();
45              break;
46              case 3:view();
47          }
48      }
49
50  }
```