```
/*#include<stdio.h>
#define max 100
int stack[max];
int top=-1;

void push()
{
   int val;
   if(top==max-1) printf("stack overflow");
   else
   {
   printf("Enter the value to be inserted:");
   scanf("%d",&val);
   top++;
   stack[top]=val;
   }
}

void pop()
{
   if(top==-1) printf("stack underflow");
   else
   {
      printf("the popeed element is:%d",stack[top]);
      top--;
   }
}

void display()
{
   int i;
   if(top==-1) printf("stack underflow");
   else
   {
      for(i=top;i>=0;i--)
      {
         printf("%d     ",stack[i]);
```

```
        }
    }
}

int main()
{
    while(1)
    {
        int c;
        printf("\n1.push\n2.pop\n3.display\n4.exit\nEnter your choice:");
        scanf("%d",&c);
        switch(c)
        {
            case 1:push();
                break;
            case 2:pop();
                break;
            case 3:display();
                break;
            case 4:exit(0);
                break;
            default:printf("Invalid choice try again!!");
                break;
        }
    }
}*/



/*
#include<stdio.h>
#define max 100
int rear=-1;
int front=-1;
int queue[max];

void insert()
```

```
{
   int val;
   if(rear==max-1)  printf("queue overflow");
   else
   {
      printf("enter the value to be inserted:");
      scanf("%d",&val);
      if(front==-1&&rear==-1)
      {
         front=0;
         rear=0;
         queue[rear]=val;
      }
      else
      {
         rear++;
         queue[rear]=val;
      }
   }
}

void delete()
{
   if(front==-1||front>rear) printf("queue underflow");
   else
   {
      printf("the deleted element is:%d",queue[front]);
      front++;
   }
}

void display()
{
   int i;
   if(front==-1||front>rear) printf("queue underflow");
   else
   {
```

```
        for(i=front;i<=rear;i++)
        {
            printf("%d     ",queue[i]);
        }
    }
}
int main()
{
    while(1)
    {
        int c;
        printf("\n1.insert\n2.delete\n3.display\n4.exit\nEnter your choice:");
        scanf("%d",&c);
        switch(c)
        {
            case 1:insert();
                break;
            case 2:delete();
                break;
            case 3:display();
                break;
            case 4:exit(0);
                break;
            default:printf("Invalid choice try again!!");
                break;
        }
    }
}

*/

/*

#include<stdio.h>
#define max 100
int queue[max];
int rear=-1;
```

```
int front=-1;

void enqueue()
{
   int val;
   if((rear+1)%max==front) printf("queue overflow");

   else
   {
      printf("enter the value to be inserted:");
      scanf("%d",&val);
      if(front==-1&&rear==-1)
      {
      front=0;
      rear=0;
      queue[rear]=val;
      }
      else
      {
         rear=(rear+1)%max;
         queue[rear]=val;
      }
   }
}

void dequeue()
{
   if(front==-1&&rear==-1) printf("queue underflow");
   else if(front==rear)
   {
      printf("the dequeued element is:%d",queue[front]);
      front=-1;
      rear=-1;
   }
   else
   {
      printf("the dequeued element is:%d",queue[front]);
```

```
        front=(front+1)%max;
    }
}

void display()
{
        int i;
    if(front==-1&&rear==-1) printf("queue underflow");
    else
    {
        for(i=front;i<=rear;i++)
        {
            printf("%d    ",queue[i]);
        }
    }
}
int main()
{
    while(1)
    {
        int c;
        printf("\n1.enqueue\n2.dequeue\n3.display\n4.exit\nEnter your choice:");
        scanf("%d",&c);
        switch(c)
        {
            case 1:enqueue();
                break;
            case 2:dequeue();
                break;
            case 3:display();
                break;
            case 4:exit(0);
                break;
            default:printf("Invalid choice try again!!");
                break;
        }
    }
```

```
}
*/

/*
#include<stdio.h>
int main()
{
    int a,b,temp,*x,*y;
    printf("Enter a:");
    scanf("%d",&a);
    printf("Enter b:");
    scanf("%d",&b);
    printf("Before swapping\na=%d\nb=%d\n",a,b);
    x=&a;
    y=&b;
    temp=*x;
    *x=*y;
    *y=temp;
    printf("After swapping\na=%d\nb=%d\n",a,b);
}
*/

/*
#include<stdio.h>
struct student
{
    int usn;
    char name[40];
    char dept[40];
    char gender[40];
};

int main()
{
    int a,i,b;
    struct student st[100];
    printf("Enter how many records to be entered:");
```

```
    scanf("%d",&a);
    printf("enter the records of the students:\n");
    for(i=0;i<a;i++)
    {
        printf("\nrecord:%d\n",i+1);
        printf("Name:");
        scanf("%s",st[i].name);
        printf("USN:");
        scanf("%d",&st[i].usn);
        printf("Gender:");
        scanf("%s",st[i].gender);
        printf("Dept:");
        scanf("%s",st[i].dept);
    }
    printf("The details are:\n");
    for(i=0;i<a;i++)
    {
        printf("\nrecord:%d\n",i+1);
        printf("Name:%s\n",st[i].name);
        printf("USN:%d\n",st[i].usn);
        printf("Gender:%s\n",st[i].gender);
        printf("Dept:%s\n",st[i].dept);
    }
}
*/

/*
#include<stdio.h>
struct slist
{
    int no;
    struct slist *ptr;
};
typedef struct slist node;
node *head,*cur,*new1;

void create()
```

```
{
   new1=(node*)malloc(sizeof(node));
   printf("Enter the data to be inserted:");
   scanf("%d",&new1->no);
   head=new1;
   new1->ptr=NULL;
}

void insert()
{
   new1=(node*)malloc(sizeof(node));
   printf("Enter the data to be inserted:");
   scanf("%d",&new1->no);
   cur=head;
   new1->ptr=cur;
   head=new1;
}

void delete()
{
   cur=head;
   head=cur->ptr;
   free(cur);
}

void display()
{
   cur=head;
   while(cur!=NULL)
   {
      printf("%d->",cur->no);
      cur=cur->ptr;
   }
}

int main()
{
```

```
   int c;
   while(1)
   {
      printf("\n1.create\n2.insert\n3.delete\n4.display\n5.exit\nEnter your choice:");
      scanf("%d",&c);
      switch(c)
      {
         case 1:create();
            break;
         case 2:insert();
            break;
         case 3:delete();
            break;
         case 4:display();
            break;
         case 5:exit(0);
            break;
         default:printf("invaild choice!!try again!!");
            break;
      }
   }
}

*/

/*
#include<stdio.h>
#include<malloc.h>
struct dlist
{
   int no;
   struct dlist *lptr;
   struct dlist *rptr;
};
typedef struct dlist node;
node *head,*cur,*new1,*prev;
```

```
void create()
{
   new1=(node*)malloc(sizeof(node));
   printf("Enter the data to be inserted:");
   scanf("%d",&new1->no);
   head=new1;
   new1->lptr=new1->rptr=NULL;
}

void insert()
{
   new1=(node*)malloc(sizeof(node));
   printf("Enter the data to be inserted:");
   scanf("%d",&new1->no);
   cur=head;
   new1->rptr=cur;
   new1->rptr->lptr=new1;
   head=new1;
}

void delete()
{
   cur=head;
   cur->rptr->lptr=NULL;
   head=cur->rptr;
   free(cur);
}

void display()
{
   cur=head;
   while(cur!=NULL)
   {
      printf("<-%d->",cur->no);
      cur=cur->rptr;
   }
}
```

```
int main()
{
   int c;
   while(1)
   {
      printf("\n1.create\n2.insert\n3.delete\n4.display\n5.exit\nEnter your choice:");
      scanf("%d",&c);
      switch(c)
      {
         case 1:create();
            break;
         case 2:insert();
            break;
         case 3:delete();
            break;
         case 4:display();
            break;
         case 5:exit(0);
            break;
         default:printf("invaild choice!!try again!!");
            break;
      }
   }
}
*/

/*
#include<stdio.h>
#include<malloc.h>
struct slist
{
   int co;
   int po;
   struct slist *ptr;
};
```

```
typedef struct slist node;
node *head,*head1,*head2,*new1,*cur,*start1,*start2;

void create(node *head)
{
   cur=head;
   int i;
   do{
      new1=(node*)malloc(sizeof(node));
      printf("Enter the Coeff:");
      scanf("%d",&new1->co);
      printf("Enter the power:");
      scanf("%d",&new1->po);
      new1->ptr=NULL;
      cur->ptr=new1;
      cur=new1;
      printf("do you want to continue?? if yes 1:");
      scanf("%d",&i);
   }while(i==1);
}


void display(node *head)
{
   cur=head->ptr;
   while(cur!=NULL)
   {
      printf("%d^%d+",cur->co,cur->po);
      cur=cur->ptr;
   }
}

void polyadd(node *head1,node *head2)
{
   start1=head1->ptr;
   start2=head2->ptr;
```

```
    printf("\n\nThe added polynomial is:\n");
    while(start1!=NULL&&start2!=NULL)
    {
        if(start1->po==start2->po)
        {
            printf("%d^%d+",start1->co+start2->co,start1->po);
            start1=start1->ptr;
            start2=start2->ptr;
        }
        else if(start1->po>start2->po)
        {
            printf("%d^%d+",start1->co,start1->po);
            start1=start1->ptr;
        }
        else
        {
            printf("%d^%d+",start2->co,start2->po);
            start2=start2->ptr;
        }
    }
    while(start1!=NULL)
    {
        printf("%d^%d+",start1->co,start1->po);
        start1=start1->ptr;
    }
    while(start2!=NULL)
    {
        printf("%d^%d+",start2->co,start2->po);
        start2=start2->ptr;
    }
}

int main()
{
    head1=(node*)malloc(sizeof(node));
    printf("Enter the first polynomial:\n");
    create(head1);
```

```
    printf("The first polynomial is:\n");
    display(head1);
    head2=(node*)malloc(sizeof(node));
    printf("\nEnter the second polynomial:\n");
    create(head2);
    printf("The second polynomial is:\n");
    display(head2);
    polyadd(head1,head2);
}

*/
/*

#include<stdio.h>
#define max 100
#include<ctype.h>
#include<string.h>
int stack[max];
int top=-1;

void push(int item)
{
    if(top==max-1) printf("stack overflow");
    else{
        top++;
        stack[top]=item;
    }
}

int pop()
{
    int i;
    if(top==-1) printf("stack underflow");
    else
    {
        i=stack[top];
        top--;
```

```
        return i;
    }
    return 0;
}


void eval(char post[])
{
    char ch;
    int i,A,B,val;
    for(i=0;post[i]!=')';i++)
    {
        ch=post[i];
        if(isdigit(ch))
            push(ch-'0');
        else if(ch=='+'||ch=='-'||ch=='*'||ch=='/')
        {
            A=pop();
            B=pop();
            switch(ch)
            {
                case '+':val=B+A;
                    break;
                case '-':val=B-A;
                    break;
                case '*':val=B*A;
                    break;
                case '/':val=B/A;
                    break;
            }
            push(val);
        }
    }
    printf("The result is:%d",pop());
}

int main()
```

```
{
   int i;
   char ch,post[max];
   printf("Enter the postfix exp:");
   for(i=0;i<max-1;i++)
   {
      scanf("%s",&post[i]);
      if(post[i]==')') break;
   }
   eval(post);
}



*/

/*

#include<stdio.h>
#define max 100
#include<ctype.h>
#include<string.h>
char stack[max];
int top=-1;

void push(char val)
{
   if(top==max-1) printf("stack oberflow");
   else{
      top++;
      stack[top]=val;
   }
}

char pop()
{
   char i;
   if (top==-1) printf("stack  underflow");
```

```
    else{
        i=stack[top];
        top--;
        return i;
    }
    return 0;
}

int isop(char sy)
{
    if(sy=='*'||sy=='/'||sy=='+'||sy=='-'||sy=='^')
        return 1;
    else
        return 0;
}

int pre(char sy)
{
    if(sy=='^') return 3;
    else if(sy=='*'||sy=='/') return 2;
    else if(sy=='+'||sy=='-') return 1;
    else return 0;
}

void intopo(char in[],char po[])
{
    int i,j;
    char x,item;
    push('(');
    strcat(in,")");
    i=0;
    j=0;
    item=in[i];
    while(item!='\0')
    {
        if(item=='('
            push(item);
```

```
    else if(isdigit(item)||isalpha(item))
    {
        po[j]=item;
        j++;
    }
    else if(isop(item)==1)
    {
        x=pop();
        while(isop(x)==1&&pre(x)>=pre(item))
        {
            po[j]=x;
            j++;
            x=pop();
        }
        push(x);
        push(item);
    }
    else if(item==')')
    {
        x=pop();
        while(x!='(')
        {
            po[j]=x;
            j++;
            x=pop();
        }
    }
    else
    {
        printf("invalid infix expression!!!");
    }
    i++;
    item=in[i];
}
if(top>-1)
{
    printf("invalid infix expression!!!");
```

```
    }
    po[j]='\0';
}

int main()
{
    char po[max],in[max];
    printf("Enter the infix expression:");
    gets(in);
    intopo(in,po);
    printf("The postfix expression is:%s",po);
    return 0;
}
*/

/*
#include<stdio.h>
int main()
{
    int a[6]={4,1,8,6,2,7};
    int n,i,j,temp;
    for(i=0;i<6;i++)
    {
        for(j=0;j<6;j++)
        {
            if(a[j]>a[j+1])
            {
                temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
            }
        }
    }

    for(i=0;i<6;i++)
    {
        printf("%d,",a[i]);
```

```
    }

}
*/
/*

#include<stdio.h>
int main()
{
    int a[6]={4,1,8,6,2,7};
    int i,j,small,temp;
    for(i=0;i<6;i++)
    {
        small=i;
        for(j=i+1;j<6;j++)
        {
            if(a[j]<a[small])
                small=j;
        }
        temp=a[small];
        a[small]=a[i];
        a[i]=temp;
    }

    for(i=0;i<6;i++)
    {
        printf("%d,",a[i]);
    }
}

*/
/*

#include<stdio.h>
int main()
{
    int a[6]={4,1,8,6,2,7};
```

```
    int i,j,key;
    for(i=1;i<6;i++)
    {
        key=a[i];
        j=i-1;
        while(j>=0&&a[j]>key)
        {
            a[j+1]=a[j];
            j--;
        }
        a[j+1]=key;
    }
    for(i=0;i<6;i++)
    {
        printf("%d,",a[i]);
    }
}
*/

/*
#include<stdio.h>
void merge(int a[],int lo,int mid,int hi)
{
    int n1,n2,i,j,k;
    int L[20];
    int R[20];
    n1=mid-lo+1;
    n2=hi-mid;
    for(i=0;i<n1;i++)
    {
        L[i]=a[lo+i];
    }
    for(j=0;j<n2;j++)
    {
        R[j]=a[mid+1+j];
    }
    i=0;
```

```
    j=0;
    k=lo;
    while(i<n1&&j<n2)
    {
        if(L[i]<=R[j])
        {
            a[k]=L[i];
            i++;
        }
        else
        {
            a[k]=R[j];
            j++;
        }
        k++;
    }
    while(i<n1)
    {
        a[k]=L[i];
        i++;
        k++;
    }
    while(j<n2)
    {
        a[k]=R[j];
        j++;
        k++;
    }
}
void mergesort(int a[],int lo,int hi)
{
    int mid;
    if(lo<hi)
    {
        mid=(lo+hi)/2;
        mergesort(a,lo,mid);
        mergesort(a,mid+1,hi);
```

```
        merge(a,lo,mid,hi);
    }
}
int main()
{
    int i;
    int a[6]={4,1,8,6,2,7};
    mergesort(a,0,6);
    for(i=0;i<6;i++)
    {
        printf("%d,",a[i]);
    }
}
*/

/*

#include<stdio.h>
bin(int a[],int lo,int hi,int key)
{
    int mid =(lo+hi)/2;
    if(lo>hi)
    {
        printf("Key not found");
        return;
    }
    if(key==a[mid]) printf("key found");
    else if(key<a[mid])
    {
        bin(a,lo,mid-1,key);
    }
    else if(key>a[mid])
    {
        bin(a,mid+1,hi,key);
    }
}
int main()
```

```
{
   int i,j,temp;
   int a[6]={4,1,8,6,2,7};
   for(i=0;i<6;i++)
   {
      for(j=0;j<6;j++)
      {
         if(a[j]>a[j+1])
         {
            temp=a[j];
            a[j]=a[j+1];
            a[j+1]=temp;
         }
      }
   }
bin(a,0,6,10);
}

*/
/*




#include<stdio.h>
#include<malloc.h>
struct tree
{
   struct tree *lch;
   int no;
   struct tree *rch;
};

typedef struct tree node;
node *root=NULL,*cur,*cur1,*bak,*par,*new1;

void insert(node *cur,node *new1)
```

```
{
   if(new1->no>cur->no)
   {
      if(cur->rch!=NULL) insert(cur->rch,new1);
      else cur->rch=new1;
   }
   else if(new1->no<cur->no)
   {
      if(cur->lch!=NULL) insert(cur->lch,new1);
      else cur->lch=new1;
   }
}

void traverse(node *cur)
{
   if(cur==NULL) return;
   else
   {
      traverse(cur->lch);
      printf("<-%d->",cur->no);
      traverse(cur->rch);
   }
}

void delete(node *cur,int key)
{
   par=cur;
   while(cur!=NULL)
   {
      if(key>cur->no)
      {
         par=cur;
         cur=cur->rch;
      }
      else if(key<cur->no)
      {
         par=cur;
```

```
            cur=cur->lch;
        }
        else if(key==cur->no)
        {
            break;
        }
    }

    if(cur->lch==NULL&&cur->rch==NULL)
    {
        if(par->rch==cur) par->rch=NULL;
        else par->lch=NULL;
    }
    else if(cur->lch!=NULL&&cur->rch==NULL)
    {
        if(par->rch==cur) par->rch=cur->lch;
        else par->lch=cur->lch;
    }
    else if(cur->lch==NULL&&cur->rch!=NULL)
    {
        if(par->rch==cur) par->rch=cur->lch;
        else par->lch=cur->rch;
    }
    else if(cur->rch!=NULL&&cur->lch!=NULL)
    {
        cur1=cur->lch;
        bak=cur1;
        while(cur1!=NULL)
        {
            bak=cur1;
            cur1=cur1->rch;
        }
        cur->no=cur1->no;
        if(cur1->lch!=NULL) bak->rch=cur1->lch;
        else bak->rch=NULL;
    }
}
```

```
int main()
{
   int ch,key;
   while(1)
   {
      printf("\n1.insert\n2.delete\n3.traverse\n4.exit\nenter your choice:");
      scanf("%d",&ch);
      switch(ch)
      {
         case 1:new1=(node*)malloc(sizeof(node));
                printf("Enter the data:");
                scanf("%d",&new1->no);
                new1->lch=new1->rch=NULL;
                if(root==NULL) root=new1;
                else
                {
                   insert(root,new1);
                }
                break;
         case 2:printf("Enter the value to be deleted:");
                scanf("%d",&key);
                delete(root,key);
                break;
         case 3:traverse(root);
                break;
         case 4:exit(0);
                break;
         default:printf("Wrong choice!!!try again!!");
                break;
      }
   }
}
```

```
*/
/*

#include<stdio.h>

int partition(int a[],int start,int end)
{
   int pivot=a[end];
   int temp,j;
   int i=start-1;
   for(j=start;j<end;j++)
   {
      if(a[j]<pivot)
      {
         i++;
         temp=a[i];
         a[i]=a[j];
         a[j]=temp;
      }
   }
   temp=a[i+1];
   a[i+1]=a[end];
   a[end]=temp;
   return i+1;
}

void quick(int a[],int start,int end)
{
   if(start<end)
   {
      int p=partition(a,start,end);
      quick(a,start,p-1);
      quick(a,p+1,end);
   }
}
int main()
{
```

```
    int i;
    int a[6]={6,3,7,1,9,4};
    quick(a,0,6);
    for(i=0;i<6;i++)
    {
        printf("%d    ",a[i]);
    }
}
*/
```