List of Practical's

1. Write a program to reverse a number (user input).
2. Write a program to reverse a number using array.
3. Write a program to perform bubble sort.
4. Write a program to perform selection sort.
5. Write a program to perform insertion sort.
6. Write a program to perform insertion, deletion, merge and display the array.
7. Write a program to perform addition, subtraction, multiplication of a matrix.
8. Write a program to perform implementation of sparse matrix.
9. Write a program to perform insertion, insertion at position, deletion, deletion from a position and displaying a linked list.
10. Write a program to perform implementation of stack.
11. Write a program to perform implementation of stack using linked list.
12. Write a program to perform implementation of queue.
13. Write a program to perform implementation of queue using linked list.
14. Write a program to perform linear and binary search.
15. Write a program to implement doubly linked list.
16. Write a program to perform implementation of polynomial expression and perform addition of two polynomial expressions.

1. **Write a program to reverse a number (user input).**

   Code:

```
#include<stdio.h>
#include<conio.h>
int revint(int);
void main()
{
        int n,result;
        printf("Eneter the no");
        scanf("%d",&n);
        result=revint(n);
        printf("The reverse of number %d is %d",n,result);
        getch();
}
int revint(int p)
{
        int d;
        static int r=0;
        if(p==0)
        {
                return(p);
        }
        else
        {
                d=p%10;
                r=r*10+d;
                revint(p/10);
        }
        return(r);
}
```

2. Write a program to reverse a number using array

Code :

```
#include <stdio.h>
int main()
{
        int array[100], n, c, t, end;
        scanf("%d", &n);
         end = n – 1;
        for (c = 0; c < n; c++) {
        scanf("%d", &array[c]);
        for (c = 0; c < n/2; c++)
        {
                t = array[c];
                array[c] = array[end];
                array[end] = t;
                end--;
        }
        printf("Reversed array elements are:\n");
        for (c = 0; c < n; c++)
        {
                 Printf("%d\n", array[c]);
        }
        return 0;
}
```

3. Write a program to perform bubble sort.

Code:

```
#include<stdio.h>
void main()
{
        int a[100],n,i,j,temp;
        printf("How many elements");
        scanf("%d",&n);
        printf("Enter the element of array");
        for(i=0;i<=n-1;i++)
        {
                scanf("%d",&a[i]);
        }
        for(i=0;i<=n-1;i++)
        {
                for(j=0;j<=n-1-i;j++)
                {
                        if(a[j]>a[j+1])
                        {
                                temp=a[j];
                                a[j]=a[j+1];
                                a[j+1]=temp;
                        }
                }
        }
        printf("Element of array after the sorting are:\n");
        for(i=0;i<=n-1;i++)
        {
                printf("%d\n",a[i]);
        }
}
```

4. Write a program to perform selection sort.

Code:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a[100],n,i,j,temp,loc,min;
        clrscr();
        printf("\How many elements:\n");
        scanf("%d",&n);
        printf("Enter the element of array\n");
        for(i=0;i<=n-1;i++)
        {
                scanf("%d",&a[i]);
        }
        min=a[0];
        for(i=0;i<=n-1;i++)
        {
                min=a[i];
                loc=i;
                for(j=i+1;j<=n-1;j++)
                {
                        if(a[j]<min)
                        {
                                min=a[j];
                                loc=j;
                        }
                }
```

```
            if(loc!=1)
            {
                temp=a[i];
                a[i]=a[loc];
                a[loc]=temp;
            }
        }

        printf("The number after selection sorting are:\n");
        for(i=0;i<=n-1;i++)
        {
            printf("%d\n",a[i]);
        }
        getch();
}
```

5. Write a program to perform insertion sort

Code :

```
#include<stdio.h>
void main()
{
        int a[100],n,k,i,j,temp;
        printf("How many elements\n");
        scanf("%d",&n);
        printf("Enter the element of array");
        for(i=0;i<=n-1;i++)
        {
                scanf("%d",&a[i]);
        }
        for(k=1;k<=n-1;k++)
        {
                temp=a[k];
                j=k-1;
                while((temp<a[j])&&(j>=0))
                {
                        a[j+1]=a[j];
                        j=j-1;
                }
                a[j+1]=temp;
        }
        printf("Element of array after sorting\n");
        for(i=0;i<=n-1;i++)
        {
                printf("%d\n",a[i]);
        }
}
```

6. Write a program to perform insertion, deletion , merge and display the array.

Code:

```
#include<stdio.h>
#include<conio.h>
void insert();
void del();
void traversa();
int a[10],i=-1;
void main()
{
        int n;
        clrscr();
        do
        {
                printf("****************************** YOUR CHOICES
                ARE ***************************\n");
                printf("1. Insert\n");
                printf("2. Delete\n");
                printf("3. Traverse\n");
                printf("4. Exit\n");
                printf("NOW ENTER YOUR CHOICE :\n");
                scanf("%d",&n);
                switch(n)
                {
                        case 1:
                                insert();
                                break;
                        case 2:
                                del();
                                break;
```

```
                        case 3:
                                traversa();
                                break;
                        case 4:
                                break;
                        default:
                                printf("SORRY!YOUR CHOICE IS INVALID!!");
                }
        }while(n!=5);
        getch();
}
void insert()   /Body of insert function
{
        if(i<=8)
        {
                printf("Enter the No.(10 elements)\n");
                while(i<=8)
                {
                        i++;
                        scanf("%d",&a[i]);
                }
        }
        else
        {
                printf("The Array is Full");
        }
}
void del()  //Body of del function
{
        int item,j,flag=0,pos;
        printf("Enter the no to delete : ");
```

```
        scanf("%d",&item);
        for(j=0;j<=i;j++)
        {
                if(a[j]==item)
                {
                        flag=1;
                        pos=j;
                }
        }
        if(flag)
        {
                for(j=pos;j<=i;j++)
                {
                        a[j]=a[j+1];
                }
                i--;
        }
        else
        {
                printf("The no is not in list, Sorry\n");
        }
}
void traversa()  //Body of traversa function
{
        int j;
        for(j=0;j<=i;j++)
        {
                printf("%d\n",a[j]);
        }
}
```

7. Write a program to perform addition ,subtraction,multiplication of a matrix

Code:

```
#include<stdio.h>
void main( )
{
        int a[2][2], b[2][2],s[2][2];
        int i,j,k;
        printf("Enter first matrix:\n" );
        for( i=1;i<=2;i++)
        {
                for( j=1;j<=2;j++)
                {
                        printf("Enter%d%d\n",i,j);
                        scanf("%d",&a[i][j]);
                }
        }
        printf("Enter second matrix:\n");
        for(i=1;i<=2;i++)
        {
                for(j=1;j<=2;j++)
                {
                        printf("Enter %d%d\n",i,j);
                        scanf("%d",&b[i][j]);
                }
        }
        for (i=1;i<=2;i++)
        {
                for (j=1;j<=2;j++)
                {
                         s[i][j]= a[i][j]+b[i][j];
```

```
            }
        }
        printf("The addition matrix is:\n");
        for (i=1;i<=2;i++)
        {
            for (j=1;j<=2;j++)
            {
                printf("%d\n",s[i][j] );
            }
        }
        for(i=1;i<=2;i++)
        {
            for(j=1;j<=2;j++)
            {
                s[i][j]=0;
                for(k=1;k<=2;k++)
                {
                    s[i][j] =s[i][j]+a[i][k]*b[k][j];
                }
            }
        }
        printf("Matrix Multiplication Is: \n");
        for(i=1;i<=2;i++)
        {
            for (j=1;j<=2;j++)
            {
                printf("%d\n",s[i][j]);
            }
        }
}
```

8. Write a program to perform implementation of sparse matrix.

Code:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a[10][10],b[10][3],r,c,s=0,i,j;
        clrscr();
        printf("\nenter the order of the sparse matrix");
        scanf("%d%d",&r,&c);
        printf("\nenter the elements in the sparse matrix(mostly zeroes)");

        for(i=0;i<r;i++)
        {
                for(j=0;j<c;j++)
                {
                        printf("\n%d row and %d column ",i,j);
                        scanf("%d",&a[i][j]);
                        if(a[i][j]!=0)
                        {
                                b[s][0]=a[i][j];
                                b[s][1]=i;
                                b[s][2]=j;
                                s++;
                        }
                }

                printf("\nthe sparse matrix is given by");
                printf("\n");
```

```
        for(i=0;i<s;i++)
        {
                for(j=0;j<3;j++)
                {
                `       printf("%d ",b[i][j]);
                }
                printf("\n");
        }
        getch();
}
```

9. Write a program to perform insertion,insertion at position ,deletion ,deletion from a position and displaying a linked list.

Code:

```
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
struct node
{
        int info;
        struct node *next;
};
typedef struct node NODE;
NODE *start;
void createmptylist(NODE **start)
{
        *start=(NODE *)NULL;
}
void traversinorder(NODE *start)
{
        while(start != (NODE *) NULL)
        {
                printf("%d\n",start->info);
                start=start->next;
        }
}
void insertatbegin(int item)
{
        NODE *ptr;
        ptr=(NODE *)malloc(sizeof(NODE));
        ptr->info=item;
```

```
        if(start==(NODE *)NULL)
                ptr->next=(NODE *)NULL;
        else
                ptr->next=start;
        start=ptr;
}
void insert_at_end(int item)
{
        NODE *ptr,*loc;
        ptr=(NODE *)malloc(sizeof(NODE));
        ptr->info=item;
        ptr->next=(NODE *)NULL;
        if(start==(NODE*)NULL)
                start=ptr;
        else
        {
                loc=start;
                while(loc->next!=(NODE *)NULL)
                        loc=loc->next;
                loc->next=ptr;
        }
}

void dele_beg(void)
{
        NODE *ptr;
        if(start==(NODE *)NULL)
                return;
        else
        {
                ptr=start;
```

```
                start=(start)->next;
                free(ptr);
        }
}

void dele_end(NODE *start)
{
        NODE *ptr,*loc;
        if(start==(NODE *)NULL)
                return;
        else if((start)->next==(NODE *)NULL)
        {
                ptr=start;
                start=(NODE *)NULL;
                free(ptr);
        }
        else
        {
                loc=start;
                ptr=(start)->next;
                while(ptr->next!=(NODE *)NULL)
                {
                        loc=ptr;
                        ptr=ptr->next;
                }
                loc->next=(NODE *)NULL;
                        free(ptr);
        }
}
```

```
void insert_spe(NODE *start,int item)
{
        NODE *ptr,*loc;
        int temp,k;
        for(k=0,loc=start;k<temp;k++)
        {
                loc=loc->next;
                if(loc==NULL)
                {
                        printf("node in the list at less than one\n");
                        return;
                }
        }
        ptr=(NODE *)malloc(sizeof(NODE));
        ptr->info=item;
        ptr->next=loc->next;;
        loc->next=ptr;
}

dele_spe(NODE *start)
{
        NODE *ptr,*loc;
        int temp;
        printf("\nEnter the which element do you want to delete\n");
        scanf("%d",&temp);
        ptr=start;
        if(start==NULL)
        {
                printf("Empty list\n");
                return;
        }
```

```
        else
        {
                loc=ptr;
                while(ptr!=NULL)
                {
                        if(ptr->info==temp)
                        {
                                loc->next=ptr->next;
                                free(ptr);
                                return;
                        }
                        loc=ptr;
                        ptr=ptr->next;
                }
        }
}

void main()
{
        int choice,item,after;
        char ch;
        clrscr();
        createmptylist(start);
        do
        {
                printf("1.Insert element at begin \n");
                printf("2. insert element at end positon\n");
                printf("3. insert specific the position\n");
                printf("4.travers the list in order\n");
                printf("5. delete from the begin\n");
                printf("6. delete from the last\n");
```

```
printf("7. delete the element from the specific position\n");
printf("8. exit\n");
printf("enter your choice\n");
scanf("%d",&choice);
switch(choice)
{
        case 1:
                printf("Enter the item\n");
                scanf("%d",&item);
                insertatbegin(item);
                break;
        case 2:
                printf("Enter the item\n");
                scanf("%d",&item);
                insert_at_end(item);
                break;
        case 3:
                printf("Enter the item\n");
                scanf("%d",&item);
                insert_spe(start,item);
                break;
        case 4:
                printf("\ntravers the list\n");
                traversinorder(start);
                break;
        case 5:
                printf("Delete the item\n");
                dele_beg();
                break;
```

```
                    case 6:
                            printf("Delete the element\n");
                            dele_end(start);
                            break;
                    case 7:
                            printf("Delete the element");
                            dele_spe(start);
                            break;
                    case 8:
                             return;
                }
                fflush(stdin);
                printf("do your want continous\n");
                scanf("%c",&ch);
        }while((ch='y')||(ch='y'));
        getch();
}
```

10. Write a program to perform implementation of stack

Code:

```
#include<stdio.h>
#include<conio.h>
#define MAXSIZE 10
void push();
int pop();
void traverse();
int stack[MAXSIZE];
int Top=-1;
void main()
{
        int choice;
        char ch;
        do
        {
                clrscr();
                printf("\n1. PUSH ");
                printf("\n2. POP ");
                printf("\n3. TRAVERSE ");
                printf("\nEnter your choice");
                scanf("%d",&choice);
                switch(choice)
                {
                        case 1: push();
                                break;
                        case 2:printf("\nThe deleted element is %d",pop());
                                break;
                        case 3:  traverse();
                                 break;
```

```
                default: printf("\nYou Entered Wrong Choice");
            }
            printf("\nDo You Wish To Continue (Y/N)");
            fflush(stdin);
            scanf("%c",&ch);
        }while(ch=='Y' || ch=='y');
}

void push()
{
        int item;
        if(Top == MAXSIZE - 1)
        {
            printf("\nThe Stack Is Full");
            getch();
            exit(0);
        }
        else
        {
            printf("Enter the element to be inserted");
            scanf("%d",&item);
            Top= Top+1;
            stack[Top] = item;
        }
}

int pop()
{
        int item;
        if(Top == -1)
        {
```

```
                printf("The stack is Empty");
                getch();
                exit(0);
        }
        else
        {
                item = stack[Top];
                Top = Top-1;
        }
        return(item);
}

void traverse()
{
        int i;
        if(Top == -1)
        {
                printf("The Stack is Empty");
                getch();
                exit(0);
        }
        else
        {
                for(i=Top;i>=0;i--)
                {
                        printf("Traverse the element");
                        printf("\n%d",stack[i]);
                }
        }
}
```

11. Write a program to perform implementation of stack using linked list.

Code:

```
#include<stdio.h>
#include<conio.h>
struct stack
{
        int no;
        struct stack *next;
}
*start=NULL;
typedef struct stack st;
void push();
int pop();
void display();

void main()
{
        char ch;
        int choice,item;
        do
        {
                clrscr();
                printf("\n 1: push");
                printf("\n 2: pop");
                printf("\n 3: display");
                printf("\n Enter your choice");
                scanf("%d",&choice);
                switch (choice)
                {
                        case 1: push();
```

```
                        break;
                case 2: item=pop();
                        printf("The delete element in %d",item);
                        break;
                case 3: display();
                        break;
                default : printf("\n Wrong choice");
            }
            printf("\n do you want to continue(Y/N)");
            fflush(stdin);
            scanf("%c",&ch);
        }while (ch=='Y'||ch=='y');
}

void push()
{
        st *node;
        node=(st *)malloc(sizeof(st));
        printf("\n Enter the number to be insert");
        scanf("%d",&node->no);
        node->next=start;
        start=node;
}

int pop()
{
        st *temp;
        temp=start;
        if(start==NULL)
        {
                printf("stack is already empty");
```

```
                getch();
                exit();
        }
        else
        {
                start=start->next;
                free(temp);
        }
        return(temp->no);
}

void display()
{
        st *temp;
        temp=start;
        while(temp->next!=NULL)
        {
                printf("\nno=%d",temp->no);
                temp=temp->next;
        }
        printf("\nno=%d",temp->no);
}
```

12. Write a program to perform implementation of queue.

Code:

```
#include<stdio.h>
#include<conio.h>
int queue[5];
long front,rear;
void main()
{
        int choice,info;
        clrscr();
        //Initialising queue
        initqueue();
        while(1)
        {
                clrscr();
                 //Displaying menu
                printf("            MENU            \n");
                printf("1. Insert  an element in queue\n");
                printf("2.delete  an element from queue\n");
                printf("3.Display the queue\n");
                printf("4.Exit!\n");
                printf("Your choice: ");
                scanf("%i",&choice);
                switch(choice)
                {
                        case 1:
                                if(rear<4)
                                {
                                        printf("enter the number");
                                        scanf("%d",&info);
```

```
                if (front==-1)
                {
                        front=0;
                        rear=0;
                }
                else
                        rear=rear+1;
                        queue[rear]=info;
         }
        else
                printf("queue is full");
        getch();
        break;
case 2:
        int info;
        if(front!=-1)
        {
                info=queue[front];
                 if(front==rear)
                {
                        front=-1;
                         rear=-1;
                }
                else
                        front=front+1;
                        printf("no deleted is = %d",info);
        }
        else
                printf("queue is empty");
        getch();
        break;
```

```
                        case 3: display();
                                getch();
                                 break;
                        case 4: exit();
                                break;
                        default:
                                printf("You entered wrong choice!");
                                getch();
                                break;
                }
        }
}

void initqueue()
{
         //Initialising front & rear to -1
        front=rear=-1;
}

/*displays the current position of the queue*/
void display()
{
        int i; //For loop driver
        //Displaying elements in queue
        for(i=front;i<=rear;i++)
        printf("%i\n",queue[i]);
}
```

13. Write a program to perform implementation of queue using linked list.

Code :

```
#include<stdio.h>
struct queue
{
        int no;
        struct queue *next;
}
*start=NULL;
void add();
int del();
void traverse();
void main()
{
        int ch;
        char choice;
        do
        {
                printf("----1. add\n");
                printf("----2. delete\n");
                printf("----3. traverse\n");
                printf("----4. exit\n");
                printf("Enter your choice\n");
                scanf("%d",&ch);
                switch(ch)
                {
                        case 1: add();
                                break;
                        case 2: printf("the delete element is\n%d",del());
                                break;
```

```
                        case 3: traverse();
                                break;
                        case 4: return;
                        default : printf("wrong choice\n");
                }
                fflush(stdin);
                scanf("%c",&choice);
        }
        while(choice!=4);
}
void add()
{
        struct queue *p,*temp;
        temp=start;
        p=(struct queue*)malloc(sizeof(struct queue));
        printf("Enter the data");
        scanf("%d",&p->no);
        p->next=NULL;
        if(start==NULL)
        {
                start=p;
        }
        else
        {
                while(temp->next!=NULL)
                {
                        temp=temp->next;
                }
                temp->next=p;
        }
}
```

```
int del()
{
        struct queue *temp;
        int value;
        if(start==NULL)
        {
                printf("queue is empty");
                getch();
                return(0);
        }
`       else
        {
                temp=start;
                value=temp->no;
                start=start->next;
                free(temp);
        }
        return(value);
}
void traverse()
{
        struct queue *temp;
        temp=start;
        while(temp->next!=NULL)
        {
                printf("no=%d",temp->no);
                temp=temp->next;
        }
        printf("no=%d",temp->no);
}
```

14. Write a program to perform binary search.

Code:

```
#include<stdio.h>
 main()
{
        int c, first, last, middle, n, search, array[100];
         printf("Enter number of elements\n");
        scanf("%d",&n);
         printf("Enter %d integers\n", n);
         for ( c = 0 ; c < n ; c++ )
        {
                 scanf("%d",&array[c]);
                printf("Enter value to find\n");
                 scanf("%d",&search);
                first = 0;
                 last = n - 1;
                 middle = (first+last)/2;
                 while( first <= last )
                 {
                        if ( array[middle] < search )
                                first = middle + 1;
                         else if ( array[middle] == search )
                         {
                                printf("%d found at location %d.\n", search,
                                middle+1);
                                break;
                        }
                        else
                        {
                                 last = middle - 1;
```

```
                middle = (first + last)/2;
            }
            if ( first > last )
                printf("Not found! %d is not present in the list.\n",
                search);
            return 0;
        }
    }
```

15. Write a program to implement doubly linked list.

Code:

```
# include <stdio.h>
# include <malloc.h>
struct Dlink_list
{
        struct Dlink_list *previous;
        int info;
        struct Dlink_list *next;
};
typedef struct Dlink_list Dnode;
/* Global declaration */
int num ;
Dnode *lp=NULL,*rp=NULL;
/* Prototypes of various functions */
void Create(Dnode *);
void display (Dnode *);

/* Function main */
void main()
{
        lp = (Dnode *) malloc(sizeof(Dnode));
        Create(lp);
        display(lp);
        getch();
}

/* Function create */
void Create(Dnode *ptr)
{
```

```
        char ch;
        clrscr();
        num = 0;
        lp->previous = NULL;
        printf("\n Enter E for Exit any other key for continue: ");
        ch = getchar();
        if(ch=='e')
        {
                free(lp);
                exit(0);
        }
        do
        {
                printf("\n Input the values of the node : %d: ", (num+1));
                scanf("%d", &ptr->info);
                fflush(stdin);
                printf("\n want to continue(y/n)->: ");
                ch = getchar();
                if(ch=='y')
                {
                        ptr->next = (Dnode *) malloc(sizeof(Dnode));
                        ptr->next->previous = ptr;
                        ptr = ptr->next;
                }
                num ++;
        }while(ch=='y');
        ptr->next=NULL;
        rp=ptr; /* Assign the adress of rightmost node to rp */
        printf("\n Total nodes = %d", num);
}
```

```
/* DELETE LAST NODE FROM A SIMPLE DOUBLY LINKED LIST*/
/* Function delete */
void Delete (Dnode *ptr)
{
      if ( lp == NULL)   /* for empty list */
      {
            printf("\n List is empty\n");
            exit(0);
      }
      if(lp==rp)    /* for list containing single node */
      {
            lp=rp=NULL;
            free(ptr);
      }
      else            /* for list containing more than one node */
      {
            ptr=rp;
            rp=ptr->previous;
            rp->next=NULL;
            free(ptr);
      }
}

/* DELETE FIRST NODE FROM A SIMPLE DOUBLY LINKED LIST */
void Delete (Dnode *ptr)
{
      if ( ptr == NULL)
      {
            printf("\n List is empty\n");
            exit(0);
      }
```

```
        if(lp==rp)
        {
                lp=rp=NULL;
                free(ptr);
        }
        else
        {
                lp=ptr->next;
                lp->previous=NULL;
                free(ptr);
        }
}

/* Functio Display  */
void display (Dnode *ptr)
{
        while(ptr!=NULL)  /* Traverse up to end of the list */
        {
                printf("\n 0x%x", ptr);
                printf(" %d", ptr->info);
                ptr = ptr->next;
        }
}
```

16. Write a program to perform implementation of polynomial expression and perform addition of two polynomial expressions.

Code :

```
#include<stdio.h>
#include<conio.h>
#include<limits.h>
int select();
struct rec
`{
        float coef;
        int exp;
        struct rec *next;
};
struct rec *rear;
struct rec *create(struct rec *list);
void *add(struct rec *first,struct rec *second);
struct rec *insert(double coef,int exp,struct rec *rear);
void *display(struct rec *list);
int nodes;

void main()
{
        struct rec *first=NULL,*second=NULL;
        int choice;
        do
        {
                choice=select();
                switch(choice)
                {
                        case 1:
```

```
                        first=create(first);continue;
                case 2:
                        second=create(second);continue;
                case 3:
                        add(first,second);continue;
                case 4:
                        puts("END");exit(0);
                }
        }while(choice!=4);
}

int select()
{
        int selection;
        do
        {
                puts("Enter 1: create the first list");
                puts("Enter 2: create the second list");
                puts("Enter 3: add the two list");
                puts("Enter 4: END");
                puts("Entr your choice");
                scanf("%d",&selection);
        }while((selection<1)||(selection>4));
        return (selection);
}

struct rec *create(struct rec *x)
{
        float coef;
        int exp;
        int endexp=INT_MAX;
```

```
        struct rec *element;
        puts("Enter cofs &exp:exp in descending order:""to quit enter 0 for exp");
        x=(struct rec *)malloc(sizeof(struct rec));
        x->next=NULL;
        rear=x;
        for(;;)
        {
                puts("Enter coefficient");
                element=(struct rec*)malloc(sizeof(struct rec));
                scanf("%f",&coef);
                element->coef=coef;
                if(element->coef==0.0)break;
                puts("Enter exponent");
                scanf("%d",&exp);
                element->exp=exp;
                if((element->exp<=0)||(element->exp>=endexp))
                {
                        puts("Invalid exponent");
                        break;
                }
                element->next=NULL;
                rear->next=element;
                rear=element;
        }
        x=x->next;
        return(x);
}

void *add(struct rec *first,struct rec *second)
{
        float total;
```

```
struct rec *end,*rear,*result;
result=(struct rec *)malloc(sizeof(struct rec));
rear=end;
while((first!=NULL)&&(second!=NULL))
{
        if(first->exp==second->exp)
        {
                if((total=first->exp+second->exp)!=0.0)
                rear=insert(total,first->exp,rear);
                first=first->next;
                second=second->next;
        }
        else
                if(first->exp>second->exp)
                {
                        rear=insert(first->coef,first->exp,rear);
                        first=first->next;
                }
                else
                {
                        rear=insert(second->coef,second->exp,rear);
                        second=second->next;
                }
        }
        for(;first;first=first->next)
                rear=insert(first->coef,first->exp,rear);
        for(;second;second=second->next)
                rear=insert(second->coef,second->exp,rear);
        rear->next=NULL;
        display(end->next);
        free(end);
```

```
}

void *display(struct rec *head)
{
        while(head!=NULL)
        {
                printf("%2lf",head->coef);
                printf("%2d",head->exp);
                head=head->next;
        }
        printf("\n");
}

struct rec *insert(double coef,int exp,struct rec *rear)
{
        rear->next=(struct rec *)malloc(sizeof(struct rec));
        rear=rear->next;
        rear->coef=coef;
        rear->exp=exp;
        return(rear);
}
```