Experiment No 1.1

Date:

Sum of two polynomials

AIM:Program to read two polynomials and store them in an array. Calculate the sum of the two polynomials and display the first polynomial, second polynomial and the resultant polynomial.

ALGORITHM

PROGRAM

```
#include<stdio.h>
struct Polynomial
{
   int coeff;
   int exp;
};

struct Polynomial first[15], second[15], result[15];

void display(struct Polynomial poly[], int terms)
{
   int i;
   printf("\n");
   printf("%dX^%d ", poly[0].coeff, poly[0].exp);
   for(i = 1; i < terms ; i++)
   {
      printf("+%dX^%d ", poly[i].coeff, poly[i].exp);
   }
}

int readExpression(struct Polynomial poly[])
{
   int terms, i;
   printf("\nNumber of terms: ");
   scanf("%d", &terms);
   printf("\nEnter the coeffecients and exponents in DESCENDING order");
   for(i = 0 ; i<terms; i++)
   {
      printf("\nCoeffecient :");
      scanf("%d", &poly[i].coeff);
      printf("Exponent :");
      scanf("%d", &poly[i].exp);
   }
   return terms;
}
int addExpressions(int firstCount, int secondCount)
{
   int i, j, k;
   i = 0;
   j = 0;
   k = 0;
   while(i < firstCount || j < secondCount)
   {
      if(first[i].exp == second[j].exp)
      {
         result[k].coeff = first[i].coeff + second[j].coeff;
         result[k].exp = first[i].exp;
```

```
            i++;
            j++;
            k++;
        }
        else if(first[i].exp > second[j].exp)
        {
            result[k].coeff = first[i].coeff;
            result[k].exp = first[i].exp;
            i++;
            k++;
        }
        else
        {
            result[k].coeff = second[j].coeff;
            result[k].exp = second[j].exp;
            j++;
            k++;
        }
    }

    return k;
}
int main()
{
    int firstCount, secondCount, resultCount;
    printf("\nFirst Expression:\n");
    firstCount = readExpression(first);
    printf("\nSecond Expression:\n");
    secondCount = readExpression(second);
    printf("\nFirst Expression");
    display(first, firstCount);
    printf("\nSecond Expression:");
    display(second, secondCount);
    resultCount = addExpressions(firstCount, secondCount);
    printf("\nResultant Expression:\n");
    display(result, resultCount);
    return 0;
}
```

OUTPUT

First Expression:

Number of terms: 3

Enter the coeffecients and exponents in DESCENDING order

Coeffecient :2
Exponent :100

Coeffecient :3
Exponent :8

Coeffecient :2
Exponent :0

Second Expression:

Number of terms: 4

Enter the coeffecients and exponents in DESCENDING order
Coeffecient :2
Exponent :100

Coeffecient :5
Exponent :99

Coeffecient :4
Exponent :1

Coeffecient :3
Exponent :0

First Expression
2X^100 +3X^8 +2X^0
Second Expression:
2X^100 +5X^99 +4X^1 +3X^0
Resultant Expression:

4X^100 +5X^99 +3X^8 +4X^1 +5X^0 %

Experiment No 1.2

Date:

Read and polynomial of degree n and store in an array. Evaluate this polynomial for a given value of x.

AIM: Program to Read and polynomial of degree n and store in an array. Evaluate this polynomial for a given value of x.

ALGORITHM

PROGRAM

```
#include<stdio.h>
#include<math.h>
typedef struct Polynomial
{
   int coeff;
   int exp;
}Polynomial;

Polynomial first[15];
void display(Polynomial poly[], int terms)
{
   int i;
   printf("\n");
   printf("%dX^%d ", poly[0].coeff, poly[0].exp);
   for(i = 1; i < terms ; i++)
   {
      printf("+%dX^%d ", poly[i].coeff, poly[i].exp);
   }
}

int readExpression(Polynomial poly[])
{   int terms, i;
   printf("\nNumber of terms: ");
   scanf("%d", &terms);
   printf("\nEnter the coeffecients and exponents in DESCENDING order");
   for(i = 0 ; i<terms; i++)
   {
      printf("\nCoeffecient :");
      scanf("%d", &poly[i].coeff);
      printf("Exponent :");
      scanf("%d", &poly[i].exp);
   }
   return terms;
}
int evalExpressions(Polynomial poly[], int terms,int x)
{
   int i,sum=0;
   for(i = 0;i < terms; i++)
   {

      sum+=(poly[i].coeff)*pow(x,poly[i].exp);
   }
   return sum;

}
int main()
{
   int firstCount, resultSum,x;
   printf("\nFirst Expression:\n");
   firstCount = readExpression(first);
   printf("\nFirst Expression");
   display(first, firstCount);
   printf("\nEnter the value for x\n");
   scanf("%d",&x);
   resultSum=evalExpressions(first,firstCount,x);
   printf("\nThe sum is %d\n",resultSum);


}
```

OUTPUT

First Expression:

Number of terms: 3

Enter the coeffecients and exponents in DESCENDING order
Coeffecient :3
Exponent :2

Coeffecient :2
Exponent :1

Coeffecient :1
Exponent :0

First Expression
3X^2 +2X^1 +1X^0
Enter the value for x
2

The sum is 17

Experiment No 1.3

Date:

Sparse Matrix representation.Also find the sparsity

AIM:Given a sparse matrix. Represent and store it using an efficient method. Also find the sparsity

ALGORITHM

PROGRAM

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


typedef struct
{
        int row;
        int col;
        int val;
}sparse;

float readsparse(sparse a[], int m, int n)
{
        int i, j, k, item, p,zero=0;
        float sparsity;
        a[0].row = m;
        a[0].col = n;
        k = 1;

       printf("\nEnter the elements:\n");
       for(i=0; i<m; i++)
      {
            for(j=0; j<n; j++)
           {
                scanf("%d", &item);
                if(item != 0){
                   zero++;

                a[k].row = i;
                a[k].col = j;
                a[k].val = item;
                 k++;
               }
            }
       }
       a[0].val = k-1;

       printf("\nThe entered sparse matrix is:\n");
       printf("\nRow\tColumn\tValue\n");
       for(p=0; p<=a[0].val; p++)
       {
               printf("%d\t", a[p].row);
               printf("%d\t", a[p].col);
               printf("%d\n", a[p].val);
        }
    sparsity = (float)(m*n-(zero))/(m*n);
   return sparsity;
```

```
}

void main()
{
        int m, n;
        float t;
        sparse a[100];
        printf("\nEnter  the no of rows and columns:\t");
        scanf("%d%d",&m, &n);
        t=readsparse(a, m, n);

        printf("the Sparsity of matrix is %f",t);
}
```

OUTPUT

```
Enter  the no of rows and columns:      3 3

Enter the elements:
1 0 2
0 0 9
0 4 3

The entered sparse matrix is:

Row     Column  Value
3       3       5
0       0       1
0       2       2
1       2       9
2       1       4
2       2       3
the Sparsity of matrix is 0.444444%
```

Experiment No 1.4

Date:

Sum of two sparse matrices

AIM: Input the representation of two sparse matrices. Obtain the representation of their sum.

ALGORITHM

PROGRAM

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

typedef struct
{
        int row;
        int col;
        int val;
}sparse;

void readsparse(sparse a[],int s)
{
        int i;

        a[0].val=s;

  printf("Enter the number of rows and columns: ");
  scanf("%d %d", &a[0].row, &a[0].col);

  printf("Enter the sparse matrix representation: ");
  for (int i = 1; i <= s; i++)
    scanf("%d %d %d", &a[i].row, &a[i].col, &a[i].val);

}

void sparseAdd(sparse a[],sparse b[],sparse sum[])
{
   int i=1,j=1,l=1;
   if(a[0].row!=b[0].row || a[0].col!=b[0].col)
   {
      printf("Cannot Add !");
      exit(0);
   }
   sum[0].row=a[0].row;
   sum[0].col=a[0].col;
   while(i<=a[0].val && j<=b[0].val)
   {
      if(a[i].row < b[j].row || (a[i].row == b[j].row && a[i].col < b[j].col))
      {
         sum[l].row=a[i].row;
         sum[l].col=a[i].col;
         sum[l].val=a[i].val;
         i++;
         l++;
      }
      else if(a[i].row > b[j].row || (a[i].row == b[j].row && a[i].col > b[j].col))
      {
```

```
            sum[l].row=b[j].row;
            sum[l].col=b[j].col;
            sum[l].val=b[j].val;
            j++;
            l++;
        }
        else if(a[i].row==b[j].row && a[i].col==b[j].col)
        {
            sum[l].row=a[i].row;
            sum[l].col=a[i].col;
            sum[l].val=a[i].val+b[j].val;
            j++;
            i++;
            l++;
        }
    }
    while(i<=a[0].val)
    {
        sum[l].row=a[i].row;
        sum[l].col=a[i].col;
        sum[l].val=a[i].val;
        i++;
        l++;
    }
    while(j<=b[0].val)
    {
        sum[l].row=b[j].row;
        sum[l].col=b[j].col;
        sum[l].val=b[j].val;
        j++;
        l++;
    }
    sum[0].val=l-1;

}
void printsparse(sparse a[])
{
        int p;
        printf("\nRow\tColumn\tValue\n");
        for(p=0; p<=a[0].val; p++)
        {
                printf("%d\t", a[p].row);
                printf("%d\t", a[p].col);
                printf("%d\n", a[p].val);
         }
}
```

```
void main()
{

        sparse a[100],b[100],sum[100];
        int s1,s2;
        printf("Enter the number of non zero elements in matrix1: ");
        scanf("%d",&s1);
        readsparse(a,s1);
        printf("\nThe entered sparse matrix is:\n");
        printsparse(a);
        printf("Enter the number of non zero elements in matrix 2: ");
        scanf("%d", &s2);
        readsparse(b,s2);
        printf("\nThe entered sparse matrix is:\n");
        printsparse(b);
        sparseAdd(a,b,sum);
        printf("\nThe Sum of the sparse matrix is:\n");
        printsparse(sum);


}
```

OUTPUT

```
Enter the number of non zero elements in matrix1: 3
Enter the number of rows and columns: 3 3
Enter the sparse matrix representation: 0 2 5
1 0 1
2 1 3

The entered sparse matrix is:

Row     Column  Value
3       3       3
0       2       5
1       0       1
2       1       3
Enter the number of non zero elements in matrix 2: 3
Enter the number of rows and columns: 3 3
Enter the sparse matrix representation: 0 0 4
1 2 2
2 1 1

The entered sparse matrix is:

Row     Column  Value
3       3       3
0       0       4
```

1 2 2
2 1 1

The Sum of the sparse matrix is:

Row	Column	Value
3	3	5
0	0	4
0	2	5
1	0	1
1	2	2
2	1	4

Experiment No 1.5

Date:

Transpose of Sparse matrix

AIM: Input the representation of a sparse matrix. Find the representation of its transpose.

ALGORITHM

PROGRAM

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

typedef struct
{
        int row;
        int col;
        int val;
}sparse;

void readsparse(sparse a[], int s)
{
    int i,p;
    a[0].val=s;
    printf("Enter the number of rows and columns: ");
    scanf("%d %d", &a[0].row, &a[0].col);

  printf("Enter the sparse matrix representation: ");
  for (int i = 1; i <= s; i++)
    scanf("%d %d %d", &a[i].row, &a[i].col, &a[i].val);

   printf("\nThe entered sparse matrix is:\n");
   printf("\nRow\tColumn\tValue\n");
   for(p=0;p<=a[0].val;p++)
    {
       printf("%d\t", a[p].row);
       printf("%d\t", a[p].col);
       printf("%d\n", a[p].val);
    }
}
void fast_transpose(sparse a[],  sparse b[])
{
        int row_terms[100],  start_pos[100];
        int i,  j,  p;
        int numTerms = a[0].val;
        int numCols = a[0].col;
        b[0].row = numCols;
        b[0].col = a[0].row;
        b[0].val = numTerms;
        if(numTerms>0)
        {
                for(i =0; i<numCols;  i++)
                        row_terms[i] = 0;

                for(i=1; i<=numTerms; i++)
                        row_terms[a[i].col]++;
```

```
                start_pos[0]=1;

                for(i=1; i<numCols; i++)
                        start_pos[i] = start_pos[i-1] + row_terms[i-1];

                for(i=1; i<=numTerms; i++)
                {
                   j = start_pos[a[i].col]++;
                   b[j].row = a[i].col;
                   b[j].col = a[i].row;
                   b[j].val = a[i].val;
                }
         }
        printf("\nThe Fast Transpose sparse matrix is:\n");
        printf("\nRow\tColumn\tValue\n");
        for(p=0; p<=a[0].val; p++)
        {
                printf("%d\t", b[p].row);
                printf("%d\t", b[p].col);
                printf("%d\n", b[p].val);
        }
}
void main()
{
        sparse a[100], b[100];
        int s1;
        printf("Enter the number of non zero elements in matrix1: ");
        scanf("%d",&s1);

        readsparse(a,s1);
        fast_transpose(a, b);
}
```

OUTPUT

```
Enter the number of non zero elements in matrix1: 3
Enter the number of rows and columns: 3 3
Enter the sparse matrix representation: 0 2 5
1 0 1
2 1 3

The entered sparse matrix is:

Row     Column  Value
3       3       3
0       2       5
1       0       1
2       1       3
```

The Fast Transpose sparse matrix is:

Row	Column	Value
3	3	3
0	1	1
1	2	3
2	0	5