Experiment 10
Date 17-11-2022

Sparse Matrix

Aim

Write a program to read two matrices in normal representation . Do the following operations as menu driven , implemented with separate functions:
a) Convert two matrices to tuple form and display it.
b) Find the transposes of a matrix represented in tuple form and display them in tuple form.
c) Find the sum of the two matrices in tuple form and display the sum in tuple form.

1 Sparse Matrix operations Algorithm

1.1 Algorithm

```
1.Start
2.Define array of structures s1,s2,sum,t1 having members row,col and value
3.Input from user which operation to perform
4.Read matrix A from user
5.To convert a matrix to tuple form, define a function Tuple(a,s1,m,n)
6.Start of function Tuple(a,s1,m,n)
7.Let count=0,k=1
8.Let s1[0].row=m,s1[0].col=n
9.For i=0 to i=m-1, do steps 10-16
10.For j=0 to j=n-1, do step 11-15
11.If a[i][j] is not equal to 0, then goto step 12
12.Let s1[k].row=i
13.Let s1[k].col=j
14.Let s1[k].value=a[i][j]
15.Let k=k+1
16.Let count=count+1
17.Let s1[0].value=count
18.End of function Tuple(a,s1,m,n)
19.Call Tuple(a,s1,m,n)
20.For displaying matrix in Tuple form, define a function Display(s1)
21.Start of function Display(s1)
22.For i=0 to i=s1[0].value, do step 23
```

```
23.Print s1[i].row,s1[i].col,s1[i].value
24.End of function Display(s1)
25.Call Display(s1)
26.Read matrix A from user
27.Call Tuple(a,s1,m,n)
28.For finding the transpose of a matrix in tuple form,
define a function Transpose(s1,t1)
29.Start of function Transpose(s1,t1)
30.Create two arrays rowTerms[] and startPos[]
31.Let t1[0].row=s1[0].col
32.Let t1[0].col=s1[0].row
33.Let t1[0].value=s1[0].value
34.If t1[0].value>0, goto step 35 else goto step 48
35.For i=0 to i=(s1[0].col)-1, do step 36
36.Let rowTerms[i]=0
37.For i=1 to i=s1[0].value, do step 38
38.Let rowTerms[s1[i].col]=rowTerms[s1[i].col]+1

39.Let startPos[0]=1
40.For i=1 to i=(s1[0].col)-1 do step 41
41.Let startPos[i]=startPos[i-1]+rowTerms[i-1]
42.For i=1 to i=s1[0].value, do steps 43-47
43.Let j=startPos[s1[i].col]
44.Let t1[j].row=s1[i].col
45.Let t1[j].col=s1[i].row
46.Let t1[j].value=s1[j].value
47.Let j=j+1
48.End of function Transpose(s1,t1)
49.Call function Transpose(s1,t1)
50.Call function Display(t1)
51.For finding the sum of two matrices in tuple form,
define a function Sum(s1,s2,sum)
52.Read matrices A and B from user
53.Call Tuple(a,s1,m,n), Tuple(b,s2,p,q)
54.Start of function Sum(s1,s2,sum)
55.Let i=j=k=1
56.Let sum[0].row=sum[0].row, sum[0].col=s1[0].col
57.While i<= s1[0].value and j<=s2[0].value, repeat steps 58-87
58.If s1[i].row=s2[j].row and s1[i].col=s2[j].col, goto step 59, else goto step 65
59.Let sum[k].value=s1[i].value+s2[j].value
60.Let sum[k].row=s1[i].row
61.Let sum[k].col=s1[i].col
62.Let i=i+1
63.Let j=j+1
64.Let k=k+1
65.If s1[i].row<s2[j].row or(s1[i].row=s2[j].row and s1[i].col<s2[j].col)
```

```
goto step 66 else goto step 71
66.Let sum[k].value=s1[i].value
67.Let sum[k].row=s1[i].row
68.Let sum[k].col=s1[i].col
69.Let i=i+1
70.Let k=k+1
71.Let sum[k].value=s2[j].value
72.Let sum[k].row=s2[j].row
73.Let sum[k].col=s2[j].col
74.Let j=j+1
75.Let k=k+1
76.While i<=s1[0].value, repeat steps 77-81
77.Let sum[k].value=s1[i].value
78.Let sum[k].row=s1[i].value
79.Let sum[k].col=s1[i].col
80.Let i=i+1
81.Let k=k+1
82.While j<=s2[0].value, repeat steps 83-87
83.Let sum[k].value=s2[j].value
84.Let sum[k].row=s2[j].value
85.Let sum[k].col=s2[j].col
86.Let j=j+1
87.Let k=k+1
88.Let sum[0].value=k-1
89.End of function Sum(s1,s2,sum)
90.Call Display(sum)
91.Ask user whether to repeat
92.If choice='yes', then goto step 3
93.Stop
```

1.2 Program

```
#include<stdio.h>


#define MAX_TERMS 101


typedef struct {

int row;

int col;
```

```
int value;
} term;

term t1[MAX_TERMS];
term t2[MAX_TERMS];
term sum[MAX_TERMS];
term t[MAX_TERMS];
//int avail=0;


void convert(int a[][10],int m,int n,term t[])
{
int i,j;
int k=0;
t[k].row=m;
t[k].col=n;
k=1;
for(i=0;i<m;i++)
for(j=0;j<n;j++)
{
if(a[i][j]!=0)
{
t[k].row=i;
t[k].col=j;
```

```
t[k].value=a[i][j];

k++;

}

}

t[0].value=k-1;


}

void display(term t[])

{

printf("Row Col Values\n");

for(int i=0;i<=t[0].value;i++)

printf("%d %d %d\n",t[i].row,t[i].col,t[i].value);

}

void readmatrix(int a[][10],int m,int n)

{

for(int i=0;i<m;i++)

for(int j=0;j<n;j++)

scanf("%d",&a[i][j]);

}


void fasttranspose(term t1[],term t[])

{

int numcols,numterms,j;

numcols=t1[0].col;
```

```
numterms=t1[0].value;

t[0].row=numcols;

t[0].value=numterms;

t[0].col=t1[0].row;

int rowterms[20],startpos[20];

int i;

for(i=0;i<numcols;i++)

rowterms[i]=0;

for(i=1;i<=numterms;i++)

{

rowterms[t1[i].col]++;

}

startpos[0]=1;

for(i=1;i<numcols;i++)

{

startpos[i]=startpos[i-1]+rowterms[i-1];

}

for(i=1;i<=numterms;i++)

{

j=startpos[t[i].col]++;

t[j].row=t1[i].col;

t[j].col=t1[i].row;

t[j].value=t1[i].value;
```

```
}

}

void add(term t1[],term t2[],term sum[]){

int r1=t1[0].row,c1=t1[0].col,r2=t2[0].row,c2=t2[0].col,m=1,n=1,s=1;

int i,j;

if(r1!=r2 || c1!=c2){

return;

}

else{

sum[0].row=r1;sum[0].col=c1;

for(i=0;i<r1;i++)

for(j=0;j<c1;j++)

if (t1[m].row==i && t1[m].col==j && t2[n].row==i && t2[n].col==j){

sum[s].row=t1[m].row;

sum[s].col=t1[m].col;

sum[s].value=t1[m].value+t2[n].value;

m++;n++;s++;

}

else if (t1[m].row==i && t1[m].col==j){

sum[s].row=t1[m].row;

sum[s].col=t1[m].col;

sum[s].value=t1[m].value;

m++;s++;
```

```
}




else if (t2[n].row==i && t2[n].col==j){
sum[s].row=t2[n].row;
sum[s].col=t2[n].col;
sum[s].value=t2[n].value;
n++;s++;
}
sum[0].value=s-1;
}
}

void main()
{
for(int l=0;l<MAX_TERMS;l++)
{
t1[l].row=0;
t1[l].col=0;
t1[l].value=0;
}
int choice;
int a[10][10];
```

```
int b[10][10];
int m,n,p,q;
printf("1.READ TWO MATRIX AND DISPLAY X\n2.TRANSPOSE AND DISPLAY\n3.SUM AND DISPLAY\n4.EXIT\
do
{
scanf("%d",&choice);
switch(choice)
{
case 1:{
printf("Enter the order of matrix A:");
scanf("%d %d",&m,&n);
printf("Enter the elements of matrix A\n");
readmatrix(a,m,n);
printf("Enter the order of matrix B: ");
scanf("%d %d",&p,&q);
printf("Enter the elements of matix B\n");
readmatrix(b,p,q);
convert(a,m,n,t1);
convert(b,p,q,t2);
display(t1);
display(t2);
break;
}
```

```
case 2:{
printf("Enter the order of matrix A:");
scanf("%d %d",&m,&n);
printf("Enter the elements of matrix A\n");
readmatrix(a,m,n);
convert(a,m,n,t1);
fasttranspose(t1,t);
display(t);
break;
}
case 3:{ printf("Enter the order of matrix A:");
scanf("%d %d",&m,&n);
printf("Enter the elements of matrix A\n");
readmatrix(a,m,n);
printf("Enter the order of matrix B: ");
scanf("%d %d",&p,&q);
printf("Enter the elements of matix B\n");
readmatrix(b,p,q);
if(m!=p||n!=q)
printf("Addition not possible\n");
else
{
convert(a,m,n,t1);
```

```
convert(b,p,q,t2);

add(t1,t2,sum);

display(sum);

}

break;

}

case 4:{

return ;

}

}

}while(choice!=4);


}
```

1.3 Sample Input and Output

```
1.READ TWO MATRIX AND DISPLAY X
2.TRANSPOSE AND DISPLAY
3.SUM AND DISPLAY
4.EXIT
1
Enter the order of matrix A:2 2
Enter the elements of matrix A
1 0
0 2
Enter the order of matrix B: 2 3
Enter the elements of matix B
1 0 0
0 0 3
Row Col Values
2 2 2
0 0 1
1 1 2
Row Col Values
2 3 2
0 0 1
1 2 3
```

```
1.READ TWO MATRIX AND DISPLAY X
2.TRANSPOSE AND DISPLAY
3.SUM AND DISPLAY
4.EXIT
2
Enter the order of matrix A:2 2
Enter the elements of matrix A
1 0
0 0
Row Col Values
2 2 1
0 0 1
```

```
1.READ TWO MATRIX AND DISPLAY X
2.TRANSPOSE AND DISPLAY
3.SUM AND DISPLAY
4.EXIT
3
Enter the order of matrix A:2 2
Enter the elements of matrix A
1 2
0 0
Enter the order of matrix B: 2 2
Enter the elements of matix B
0 0
0 4
Row Col Values
2 2 3
0 0 1
0 1 2
1 1 4
```


Figure 1: text.txt File

1.4 Result

Successfully read two matrices in normal representation. Using separate functions,converted two matrices to tuple form and displayed it, displayed the transpose of a matrix represented in tuple form, computed the sum of the two matrices in tuple form and displayed the sum in tuple form