Name: Saloni Vishwakarma

Roll no: 13 (C1-Batch)

PRACTICAL NO: 1

Aim: To study an Array ADT and to implement various operations on Matrix (two dimensional array)[include display array in row major and column major form, finding transpose , matrix addition and multiplication]

Code for matrix:

```
#include<stdio.h>
struct array{
   int mat[10][10],m,n;
}a,b,add,mul,t1,t2;
void main(){
   int i,j,k,row,col,B,w,I,J;
   printf("Enter the number of rows and columns:\n");
   scanf("%d %d",&a.m,&a.n);
   b.m=a.m;
   b.n=a.m;
   printf("Enter the elements of first matrix:\n ");
   for(i=0;i<a.m;i++)
      for(j=0;j<a.n;j++)
         scanf("%d",&a.mat[i][j]);
   printf("Enter the elements of second matrix:\n ");
   for(i=0;i<b.m;i++)
      for(j=0;j<b.n;j++)
         scanf("%d",&b.mat[i][j]);
   printf("First matrix is:\n ");
   for(i=0;i<a.m;i++)
```

```
{
 for(j=0;j<a.n;j++)
       {
          printf("%d\t",a.mat[i][j]);
       }
       printf("\n");
    }
    printf("Second matrix is:\n ");
    for(i=0;i<b.m;i++){
       for(j=0;j<b.n;j++){
          printf("%d\t",b.mat[i][j]);
       }
       printf("\n");
    }
 // Adding two matrices
    for(i=0;i<a.m;i++)
    {
       for(j=0;j<b.n;j++)
       {
          add.mat[i][j]=a.mat[i][j]+b.mat[i][j];
       }
    }
    printf("Resultant matrix after the addition is: \n");
    for(i=0;i<a.m;i++)
    {
       for(j=0;j<b.n;j++)
       {
          printf("%d\t",add.mat[i][j]);
       }
       printf("\n");
```

```
}
// multiplying the two matrices
for(i=0;i<a.m;i++)
{
  for(j=0;j<b.n;j++)
  {
     mul.mat[i][j]=0;
     for(k=0;k<b.n;k++)
     {
        mul.mat[i][j]+=a.mat[i][k]*b.mat[k][j];
     }
  }
}
printf("Resultant matrix after the multiplication is: \n");
for(i=0;i<a.m;i++)
{
  for(j=0;j<b.n;j++)
  {
     printf("%d\t",mul.mat[i][j]);
  }
  printf("\n");
}
// transpose of 1st matrix
for(i=0;i<a.m;i++)
{
  for(j=0;j<a.n;j++)
  {
     t1.mat[i][j]=a.mat[j][i];
  }
}
```

```
printf("Transpose of 1st matrix:\n");
for(i=0;i<a.m;i++)
{
  for(j=0;j<a.n;j++)
  {
     printf("%d\t",t1.mat[i][j]);
  }
  printf("\n");
}
// transpose of 2nd matrix
for(i=0;i<b.m;i++)
{
  for(j=0;j<b.n;j++)
  {
     t2.mat[i][j]=b.mat[j][i];
  }
}
printf("Transpose of 2nd matrix:\n");
for(i=0;i<b.m;i++)
{
  for(j=0;j<b.n;j++)
  {
     printf("%d\t",t2.mat[i][j]);
  }
  printf("\n");
}
printf("Enter the base address: ");
scanf("%d",&B);
printf("Enter the size of each element: ");
scanf("%d",&w);
```

```
    printf("\nEnter the position of the element you want to evaluate the address for: ");
    scanf("%d d",&I,&J);
    row=B+w*(a.m*(i-1)+(j-1));
    col=B=w*((i-1)+a.n*(j-1));
    printf("\nLocation of a[%d][%d] in row major form: %d",I,J,row);
    printf("\nLocation of a[%d][%d] in column major form: %d",I,J,col);
}
```

OUTPUT:

```
Enter the number of rows and columns: 3 3
Enter the elements of first matrix:
 1 2 3 4 5 6 7 8 9
Enter the elements of second matrix:
 9 8 7 6 5 4 3 2 1
First matrix is:
 1      2      3
4       5      6
7       8      9
Second matrix is:
 9      8      7
6       5      4
3       2      1
Resultant matrix after the addition is:
10      10      10
10      10      10
10      10      10
Resultant matrix after the multiplication is:
30      24      18
84      69      54
138     114     90
Transpose of 2nd matrix:
9       6       3
8       5       2
7       4       1
Enter the base address: 250
Enter the size of each element: 4

Enter the position of the element you want to evaluate the address for: 2 3

Location of a[2][0] in row major form: 282
Location of a[2][0] in column major form: 32
Process returned 45 (0x2D)   execution time : 57.003 s
Press any key to continue.
```

Code for Linear array:

```
#include<stdio.h>
struct array{
    int arr[50],n;
}a,b,c;
void main(){
   int ch;
   printf("1.Insertion\t 2.Deletion\t 3.Sorting\t 4.Searching\t 5.Merging");
   printf("\nEnter the case number: ");
   scanf("%d",&ch);
   switch(ch)
   {
    case 1:
    {
      int i,x,index;
      printf("Enter the size of the array: ");
      scanf("%d",&a.n);
      printf("\nEnter the elements of the array: ");
      for(i=0;i<a.n;i++)
        scanf("%d",&a.arr[i]);
      printf("\nPrinting the elements of the array: ");
      for(i=0;i<a.n;i++)
        printf("%d ",a.arr[i]);
      printf("\nEnter the element needs to be inserted: ");
      scanf("%d",&x);
      printf("\nEnter the index where element needs to be inserted: ");
      scanf("%d",&index);
      a.n++;
      if(index>a.n)
        printf("\nArray Overflow. No insertion.");
```

```
  else {
      for(i=a.n-1;i>=index;i--)
        a.arr[i+1]=a.arr[i];
      a.arr[index]=x;
      printf("Printing the elements of the array: ");
      for(i=0;i<a.n;i++)
        printf("%d ",a.arr[i]);
  }
  break;
}
case 2:
{
  int i,index;
  printf("Enter the size of the array: ");
  scanf("%d",&a.n);
  printf("\nEnter the elements of the array: ");
  for(i=0;i<a.n;i++)
    scanf("%d",&a.arr[i]);
  printf("\nPrinting the elements of the array: ");
  for(i=0;i<a.n;i++)
    printf("%d ",a.arr[i]);
  printf("\nEnter the index from where element needs to be deleted: ");
  scanf("%d",&index);
  if(index>a.n)
    printf("\nArray Underflow. No deletion.");
  else{
      for(i=index+1;i<a.n;i++)
        a.arr[i-1]=a.arr[i];
      a.n--;
      printf("Successful Deletion");
```

```
    printf("\nPrinting the elements of the array: ");
    for(i=0;i<a.n;i++)
      printf("%d ",a.arr[i]);
  }
  break;
}
case 3:
{
  int i,j,temp;
  printf("Enter the size of the array: ");
  scanf("%d",&a.n);
  printf("\nEnter the elements of the array: ");
  for(i=0;i<a.n;i++)
    scanf("%d",&a.arr[i]);
  printf("\nPrinting the elements of the array: ");
  for(i=0;i<a.n;i++)
    printf("%d ",a.arr[i]);
  printf("\nBubble Sort");
  for(i=0;i<a.n-1;i++) {
     for(j=0;j<a.n-i-1;j++) {
        if(a.arr[j]>a.arr[j+1]){
           temp=a.arr[j];
           a.arr[j]=a.arr[j+1];
           a.arr[j+1]=temp;
        }
     }
  }
  printf("\nPrinting the elements of the array in ascending order: ");
     for(i=0;i<a.n;i++)
       printf("%d ",a.arr[i]);
```

```
  break;
}
case 4:
{
  int i,x,flag=0;
  printf("Enter the size of the array: ");
  scanf("%d",&a.n);
  printf("\nEnter the elements of the array: ");
  for(i=0;i<a.n;i++)
    scanf("%d",&a.arr[i]);
  printf("\nPrinting the elements of the array: ");
  for(i=0;i<a.n;i++)
    printf("%d ",a.arr[i]);
  printf("\nEnter the element needs to be searched: ");
  scanf("%d",&x);
  for(i=0;i<a.n;i++){
      if(x==a.arr[i]){
          flag=1;
          break;
      }
  }
  if(flag=1)
    printf("Element found.");
  else
    printf("Element not found.");
  break;
}
case 5:
{
  int i,j;
```

```
    printf("Enter the size of 1st array: ");
    scanf("%d",&a.n);
    printf("Enter the size of 2nd array: ");
    scanf("%d",&b.n);
    c.n=a.n+b.n;
    printf("\nEnter the elements of 1st array: ");
    for(i=0;i<a.n;i++)
      scanf("%d",&a.arr[i]);
      printf("\nEnter the elements of 2nd array: ");
    for(i=0;i<b.n;i++)
      scanf("%d",&b.arr[i]);
    printf("\nPrinting the elements of 1st array: ");
    for(i=0;i<a.n;i++)
      printf("%d ",a.arr[i]);
    printf("\nPrinting the elements of 2nd array: ");
    for(i=0;i<b.n;i++)
      printf("%d ",b.arr[i]);
    for(i=0;i<a.n;i++)
        c.arr[i]=a.arr[i];
    for(j=0;j<b.n;j++)
        c.arr[i+j]=b.arr[j];
    printf("\nPrinting the merged array: ");
        for(i=0;i<c.n;i++)
          printf("%d ",c.arr[i]);
    break;
   }
   default:
    printf("Enter the correct case number");
  }
}
```

TEST CASES:

1.Insertion: Insert an element in an Array.

a. Successful Insertion.

```
1.Insertion      2.Deletion      3.Sorting       4.Searching     5.Merging
Enter the case number: 1
Enter the size of the array: 4

Enter the elements of the array: 1 2 3 4

Printing the elements of the array: 1 2 3 4
Enter the element needs to be inserted: 56

Enter the index where element needs to be inserted: 7

Array Overflow. No insertion.
Process returned 30 (0x1E)   execution time : 12.809 s
Press any key to continue.
```

b. Array Overflow. No insertion.

```
1.Insertion      2.Deletion      3.Sorting       4.Searching     5.Merging
Enter the case number: 1
Enter the size of the array: 6

Enter the elements of the array: 1 2 3 4 5 6

Printing the elements of the array: 1 2 3 4 5 6
Enter the element needs to be inserted: 45

Enter the index where element needs to be inserted: 3
Printing the elements of the array: 1 2 3 45 4 5 6
Process returned 7 (0x7)   execution time : 41.774 s
Press any key to continue.
```

1. Deletion: Delete an element in an Array.

a. Successful Deletion.

```
1.Insertion      2.Deletion      3.Sorting       4.Searching     5.Merging
Enter the case number: 2
Enter the size of the array: 5

Enter the elements of the array: 23 4 5 6 7

Printing the elements of the array: 23 4 5 6 7
Enter the index from where element needs to be deleted: 3
Successful Deletion
Printing the elements of the array: 23 4 5 7
Process returned 4 (0x4)   execution time : 20.622 s
Press any key to continue.
```

b. Array Underflow. No Deletion.

```
1.Insertion      2.Deletion      3.Sorting       4.Searching     5.Merging
Enter the case number: 2
Enter the size of the array: 5

Enter the elements of the array: 23 4 5 6 7

Printing the elements of the array: 23 4 5 6 7
Enter the index from where element needs to be deleted: 7

Array Underflow. No deletion.
Process returned 30 (0x1E)   execution time : 44.413 s
Press any key to continue.
```

2. Sorting: Arranging the elements of an Array in ascending order or descending order.

a. Use any one of the following sorting operations: Bubble sort or Insertion Sort.

```
1.Insertion      2.Deletion      3.Sorting       4.Searching     5.Merging
Enter the case number: 3
Enter the size of the array: 5

Enter the elements of the array: 23 12 4 6 1

Printing the elements of the array: 23 12 4 6 1
Bubble Sort
Printing the elements of the array in ascending order: 1 4 6 12 23
Process returned 5 (0x5)   execution time : 22.441 s
Press any key to continue.
```

3. Searching: Find an element in an Array.

a. Element found.

```
1.Insertion      2.Deletion      3.Sorting       4.Searching     5.Merging
Enter the case number: 4
Enter the size of the array: 5

Enter the elements of the array: 2 3 4 5 6

Printing the elements of the array: 2 3 4 5 6
Enter the element needs to be searched: 6
Element found.
Process returned 14 (0xE)   execution time : 37.782 s
Press any key to continue.
```

b. Element not found.

```
1.Insertion      2.Deletion      3.Sorting       4.Searching     5.Merging
Enter the case number: 4
Enter the size of the array: 6

Enter the elements of the array: 1 2 3 4 5 6

Printing the elements of the array: 1 2 3 4 5 6
Enter the element needs to be searched: 8
Element found.
Process returned 14 (0xE)   execution time : 42.904 s
Press any key to continue.
```

4. Merging: Merging two arrays into one.

a. Successful merging and printing its elements.