```
//write a c program to print user define matrix

//BATCH:PPA9

//NAME:Nikita vilas tupe


solution:

#include<stdio.h>

//main function

void main()

{

        int i,j,m,n,a[10][10];

        printf("enter the no of rows:");

        scanf("%d",&m);


        printf("enter the no of columns:");

        scanf("%d",&n);


        printf("enter the elements:\n");

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

        {

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

                {

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

                }

                printf("your matrix is:\n\n");

                for(i=0;i<m;i++)
                {
                        for(j=0;j<n;j++)
                        {
                                printf("%d\t",a[i][j]);
                        }
                        printf("\n");
                }
        }
}
        output:
enter the no of rows:2
enter the no of columns:2
enter the elements:
1
2
3
4
your matrix is :
1       2
3       4
```

****

```
//write a c program to substract two matrices
//BATCH:PPA9
```

```
//NAME:Nikita vilas tupe


#include<stdio.h>

//main function

void main()

{

    int i,j ,n;

      int a[3][3],b[3][3],c[3][3];

    for(i=0;i<3;i++)

      {

        for(j=0;j<3;j++)

        {

              printf("enter elements of first matrix :");

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

                  }

              }

              for(i=0;i<3;i++)

              {

                  for(j=0;j<3;j++)

                      {

                      printf("enter elements in second     matrix:");

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

                            }

              }

              printf("first matrix\n");
```

```
for(i=0;i<3;i++)

{

            for(j=0;j<3;j++)

            {

                printf("second matrix\n");

                        for(i=0;i<3;i++)

                        {

                          for(j=0;j<3;j++)

                          {

                                      printf("%d",b[i][j]);

                          }

                          for(i=0;i<3;i++)

                          {

                                      for(j=0;j<3;j++)

                                      {

                                                  c[i][j]=a[i][j]-b[i][j];

                                      }

                          }

                          printf("substraction of two matrices\n");

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

                          {

                                      for(j=0;j<3;j++)

                                      {

                                                  printf("%d",c[i][j]);

                                      }
```

```
                                        printf("\n");
                                 }
                               }
                              }
}
}
```

```
output:enter elements of first matrix:1
2
  3
  4
  5
  6
  7
  8
  9
enter elements of second matrix:9
8
7
6
5
4
3
2
1
first matrix:
```

```
1       2           3

4       5           6

7         8         9

second matrix:

9    8    7

6    5    4

3    2    1

substraction of two matrix:

-8    -6      -4

-2      0      -2

4           6        8
```

****

```
    //write a c program to add two matrices

//BATCH:PPA9

//NAME:nikita vilas tupe

solution:

#include <stdio.h>

  //main function

    int main()

{

      int    m, n, c, d,a[10][10], b[10][10], sum[10][10];


      printf("Enter the number of rows and columns of matrix\n");

      scanf("%d%d", &m, &n);

      printf("Enter the elements of matrix a\n");
      for (c = 0; c < m; c++)
          for (d = 0; d < n; d++)
              scanf("%d", &a[c][d]);

      printf("Enter the elements of    matrix b\n");

      for (c = 0; c < m; c++)
          for (d = 0 ; d < n; d++)
              scanf("%d", &b[c][d]);

      printf("Sum of entered matrices:-\n");

      for (c = 0; c < m; c++) {
          for (d = 0 ; d < n; d++) {
              a[c][d] = a[c][d] + b[c][d];
              printf("%d\t", sum[c][d]);
          }
          printf("\n");
      }
        return 0;
```

```
output: enter the number of rows and columns in the matrix:
2
2
enter the elements of first matrix:
```

```
1      2

3      4

enter the elements of second matrix:

5      6

2        1

sum of entered matrices:

6      8

5      5
```

****

```
    //write a c program to search an element in 2d array matrix

//BATCH:PPA9

//NAME:Nikita vilas tupe

solution:

#include<stdio.h>

int main()

{

              int i ,j;

              int count=0;

              int rows, columns, searchelements;

          printf("Enter the number of Row and Column: \n");

    scanf("%d %d", &rows, &columns);

      int array[rows][columns];

        printf("Enter %d elements: \n", (rows*columns));

    for(int i=0; i<rows; i++){

        for(int j=0; j<columns; j++){
            scanf("%d", &array[i][j]);
        }
    }

    //Enter the search element
    printf("Enter the element to get the position: \n");
    scanf("%d", &searchelements);

        for(int i=0; i<rows; i++){
        for(int j=0; j<columns; j++){
            if(array[i][j] == searchelements)
            {
              printf("(%d, %d) \n", i, j);
                count++;
            }
        }
    }
    if(count==0)
    printf("Not found \n");
  return 0;
}
```

```
output:Enter the number of Row and Column:
2    2
Enter 4 elements:
1    2      3      4
```

```
Enter the element to get the position:

5

Not found
```

****

```
    // C Program to check whether a matrix is upper triangular matrix or not

//BATCH:PPA9

//NAME:Nikita vilas tupe


solution:

#include <stdio.h>

int main()

{

        int i,j;

        int A[10][10],m,n;

        int row, col, isUpper;


                printf("Enter no. of rows :: ");

                scanf("%d", &m);

                printf("\nEnter no. of cols :: ");

                scanf("%d",&n);

                printf("Enter values to the matrix :: \n");


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

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

                    {

                              printf("\nEnter a[%d][%d] value :: ",i,j);

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

            }

    }

printf("\nThe given matrix is :: \n\n");


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

            {

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

                    {

                            printf("\t%d", A[i][j]);

                    }

                    printf("\n\n");

            }


    // Checks whether the matrix is Upper triangular


    isUpper = 1;

    for(row=0; row<m; row++)

    {

            for(col=0; col<n; col++)

            {

                  if(col<row && A[row][col]!=0)
```

```
                {
                        isUpper = 0;
                }
        }
  }
if(isUpper==1)
  {
        printf("\nThis is a Upper triangular matrix.\n");

        for(row=0; row<m; row++)
        {
                for(col=0; col<n; col++)
                {
                        if(A[row][col] != 0)
                        {
                                printf("\t%d ", A[row][col]);
                        }
                        else
                        {
                                printf("\t");
                        }

                }

                printf("\n\n");
                }
        }
        else
        {
                printf("\nThis is Not a Upper triangular matrix.");
        }

        return 0;
}
```

```
output: Enter no. of rows :: 3

Enter no. of cols :: 3
Enter values to the matrix ::

Enter a[0][0] value :: 1

Enter a[0][1] value :: 2

Enter a[0][2] value :: 3

Enter a[1][0] value :: 4

Enter a[1][1] value :: 5

Enter a[1][2] value :: 5
Enter a[2][0] value :: 6

Enter a[2][1] value :: 6

Enter a[2][2] value :: 7

The given matrix is ::

          1         2         3

          4         5         5

          6         6         7
This is Not a Upper triangular matrix.
```

***// C Program to check whether a matrix is lower triangular matrix or not

//BATCH:PPA9

//NAME:Nikita vilas tupe

solution:

```
#include <stdio.h>

int main()

{

    int i,j;

    int A[10][10],m,n;
int row, col, isLower;


printf("Enter no. of rows :: ");
        scanf("%d", &m);
        printf("\nEnter no. of cols :: ");
        scanf("%d",&n);
        printf("\nEnter values to the matrix :: \n");
        for (i = 0; i < m; i++)
        {
                for (j = 0; j < n; j++)
                {
                          printf("\nEnter a[%d][%d] value :: ",i,j);
                          scanf("%d", &A[i][j]);
        }
}
isLower = 1;
for(row=0; row<m; row++)
{
        for(col=0; col<n; col++)
        {
          if(col>row && A[row][col]!=0)
                {
                        isLower = 0;
                }
        }
```

```
    }


    if(isLower == 1)

    {

        printf("\nMatrix is Lower triangular matrix: \n\n");

       for(row=0; row<m; row++)

        {

            for(col=0; col<n; col++)

            {

                if(A[row][col]!=0)

                {

                    printf("\t%d", A[row][col]);

                }

            }

         printf("\n\n");

        }

    }

    else

    {

        printf("\nMatrix is not a Lower triangular matrix");

    }


    return 0;

}
```

```
output:Enter no. of rows :: 3

Enter no. of cols :: 3

Enter values to the matrix ::

Enter a[0][0] value :: 2

Enter a[0][1] value :: 3

Enter a[0][2] value :: 5

Enter a[1][0] value :: 7

Enter a[1][1] value :: 9

Enter a[1][2] value :: 8

Enter a[2][0] value :: 6

Enter a[2][1] value :: 4

Enter a[2][2] value :: 8

Matrix is not a Lower triangular matrix
```

```
*********************************************************************
  // C Program to print transpose matrix of gievn matrix

//BATCH:PPA9

//NAME:Nikita vilas tupe


solution:

#include <stdio.h>

int main()

{

        int matrix[10][10], transpose[10][10];

        int i, j, m, n;

  printf("Enter number of rows : ");

        scanf("%d", &m);

        printf("Enter number of columns : ");

        scanf("%d", &n);




        printf("\nEnter the elements of matrix\n");

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

        {

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

                {

                        printf("Enter data in [%d][%d]: ", i, j);

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

                }
        }



        printf("\n");

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

        {

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

                {

                        printf("%d\t", matrix[i][j]);

                }

                printf("\n");

        }


        //Transpose of the matrix

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

        {

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

                {

                        transpose[j][i] = matrix[i][j];

                }

        }

printf("\nTranspose matrix\n");

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

        {

                for (j = 0; j < m; j++)
        {
            printf("%d\t", transpose[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

```
output:Enter number of rows : 2
Enter number of columns : 3

Enter the elements of matrix
Enter data in [0][0]: 11
Enter data in [0][1]: 12
Enter data in [0][2]: 12
Enter data in [1][0]: 14
Enter data in [1][1]: 15
Enter data in [1][2]: 16

11      12      12
14      15      16

Transpose matrix
11      14
12      15
12      16
```

****

//write a c program to check whether given matrix is identity matrix or not

// batch:PPA9

//name:nikita vilas tupe

solution:

```
#include<stdio.h>
  int main()
{
        int i, j, rows, columns, a[10][10], Flag = 1;
                printf("\n Please Enter Number of rows and columns    :    ");
        scanf("%d %d", &i, &j);
                printf("\n Please Enter the Matrix Elements \n");
        for(rows = 0; rows < i; rows++)
        {
                for(columns = 0; columns < j; columns++)
        {
                scanf("%d", &a[rows][columns]);
        }
        }
                        for(rows = 0; rows < i; rows++)
        {
                for(columns = 0; columns < j; columns++)
        {
                if(a[rows][columns] != 1 && a[columns][rows] != 0)
                {
                        Flag = 0;

                        break;

                        }

                }

        }

        if(Flag == 1)

        {

                printf("\n The Matrix that you entered is an Identity Matrix ");

        }

        else

        {

                printf("\n The Matrix that you entered is Not an Identity Matrix ");

        }

              return 0;

}

              output:             Please Enter Number of rows and columns   :   3   3


 Please Enter the Matrix Elements

1          3           0

2         4           0

7       8           0


 The Matrix that you entered is Not an Identity Matrix
```
****