**ASSIGNMENT – 9**

**1.PROBLEM STATEMENT**

Write a program in C to transpose a matrix.

**2.ALGORITHMS**

Algorithm **Transpose\_Matrix**

**Input:** The pointer ‘arr’ to the 2-D array input by the user, the dimensions of the 2-D array ‘row’ and ‘col’.

**Output:**The transpose of the 2-D array ‘arr’.

**Remarks:**The transpose operation is perfomed inplace.

**Steps:**

1. mid=1 //points to diagonal elements
2. **While**(mid<row) **do** //traversing the square part of the matrix
3. **For**(i=mid+1 to row) **do** //traversing non diagonal elements
4. **Swap**(arr[i][j],arr[j][i])//interchange the elements
5. **EndFor**
6. mid=mid+1//point mid to next diagonal element
7. **EndWhile**
8. **If**(row!=col) **then** //if input matrix is not a square matrix
9. j=row // points to non square part
10. **While**(j<col) **do** //traversing the non square part
11. **For**(i=1 to row) **do**
12. **Swap**(arr[i][j],arr[j][i])//interchange
13. **EndFor**
14. j=j+1//point j to next non square row
15. **EndWhile**
16. **EndIf**
17. **Stop**

Algorithm **Swap**

**Input:** The two variables ‘a’ and ‘b’ whose contents are to be swapped.

**Output:** The contents of ‘a’ and ‘b’ interchanged with one another.

**Remarks:** ‘a’ and ‘b’ must be passed as pointers.

**Steps:**

1. a = a+b //new value of a
2. b = a-b //b holds old value of a
3. a = a-b //a holds old value of b
4. **Stop**

**3.SOURCE CODE**

#include<stdio.h>

#include<stdlib.h>

//function to take input in a matrix

void getmatrix(int (\*arr)[20],int row,int column)

{

int i,j;

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

{

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

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

}

}

//function to display a matrix

void dispmatrix(int (\*arr)[20],int row,int column)

{

int i,j;

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

{

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

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

printf("\n");

}

}

//function for interchanging variable values

void swap(int\*a,int\*b)

{

\*a=\*a+\*b;

\*b=\*a-\*b;

\*a=\*a-\*b;

}

//function for input validation

void validate(int row,int col)

{

if(row<1 || col<1)

{

printf("Invalid Dimensions\nPlease Enter a Valid Dimension");

exit(1);

}

}

void trans(int(\*arr)[20],int row,int col)

{

int i,j,mid;

mid=0;

while(mid<row) //traversing the square part of the input matrix

{

for(i=mid+1;i<row;i++)

swap(&arr[mid][i],&arr[i][mid]);

mid++;

}

if(row!=col) //if input matrix is not a square matrix

{

j=row;

while(j<col)//traversing the non square part of the matrix

{

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

swap(&arr[i][j],&arr[j][i]);

j++;

}

}

}

int main(void)

{

int arr[20][20],row,col;

printf("Enter the number of rows needed: ");

scanf("%d",&row);

printf("Enter the number of columns needed: ");

scanf("%d",&col);

validate(row,col); //input validation

printf("Enter %d elements of the matrix: \n",row\*col);

getmatrix(arr,row,col);

printf("\nElements of the matrix: \n");

dispmatrix(arr,row,col); //displaying input matrix

trans(arr,row,col);

printf("\nTranspose: \n");

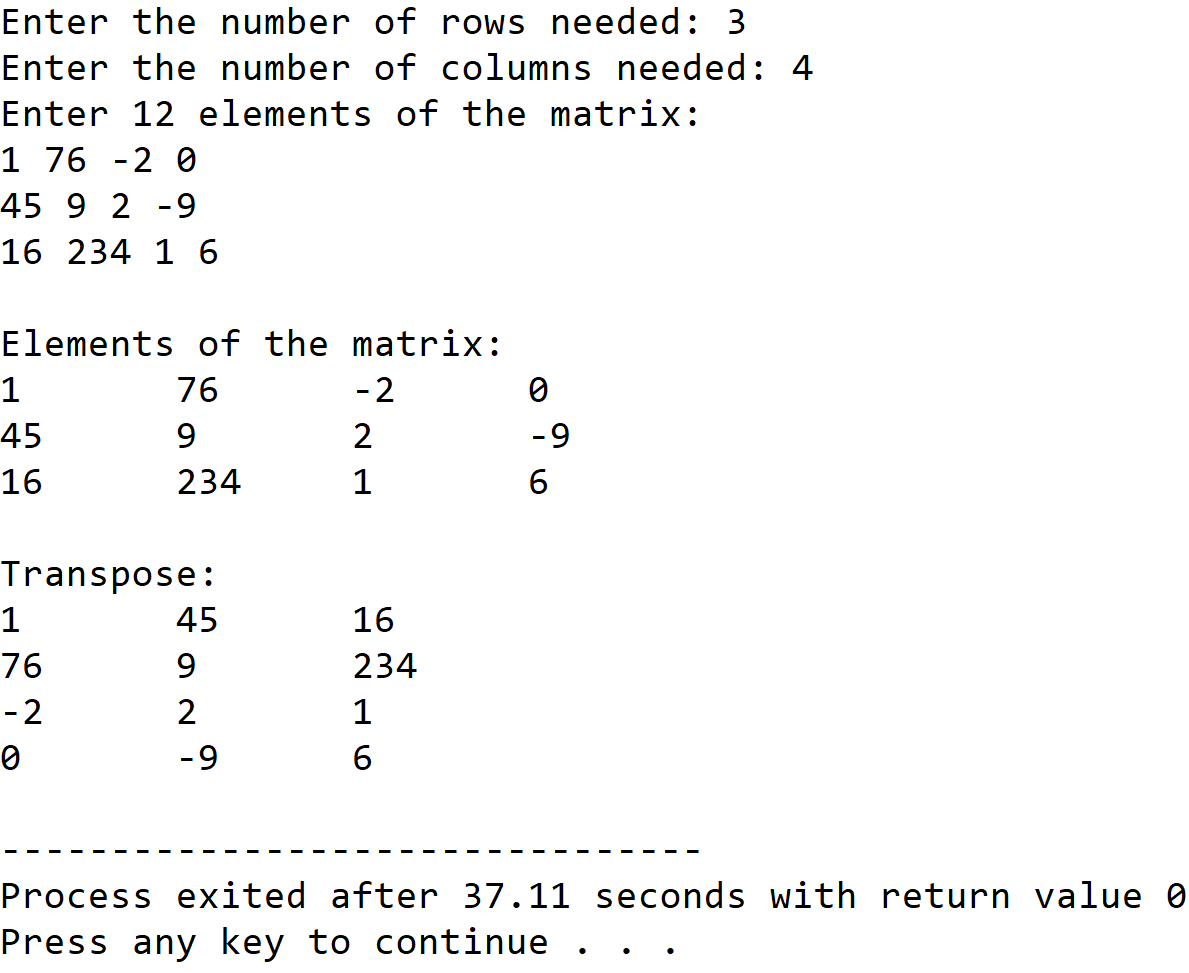
dispmatrix(arr,col,row);//displaying transposed input matrix

return 0;

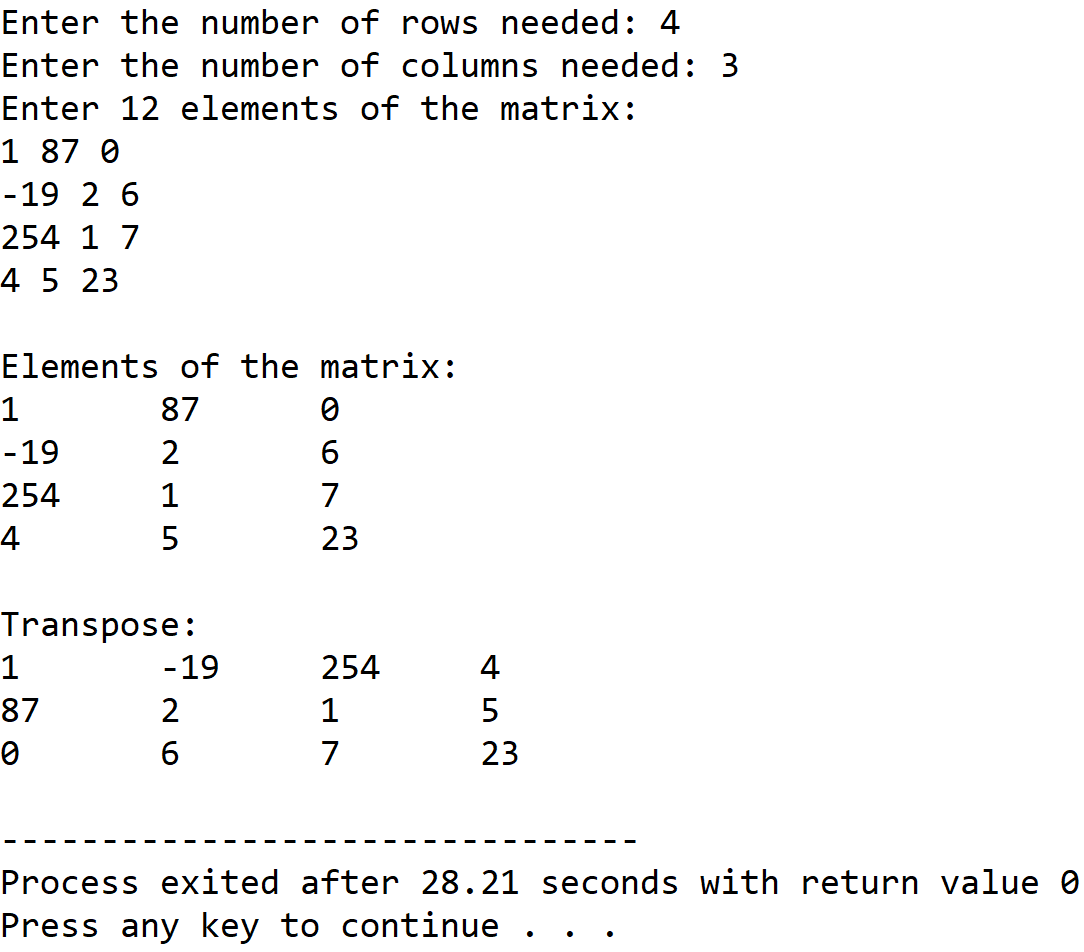
}

**4.OUTPUT**

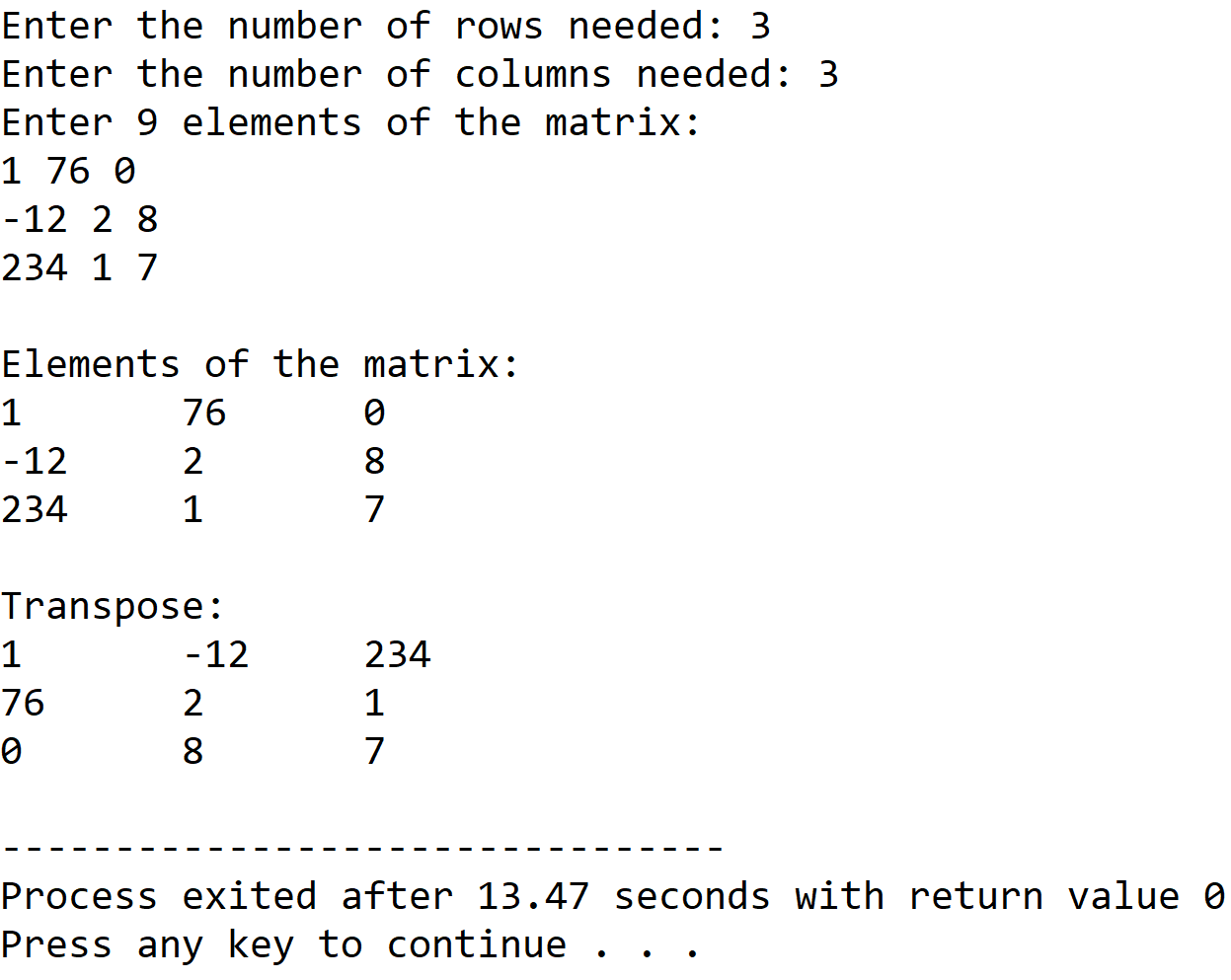
**SET 1:** Wide input matrix(row<col)



**SET 2:** Tall input matrix(row>col)



**SET 3:** Square matrix



**5.DISCUSSIONS**

**Variable Description:**

* **arr:** pointer to the 2-D array holding the matrix.
* **row,col**: dimensions of the matrix.
* **mid**:index of diagonal elements of square part of input matrix.
* **i,j**: loop counters

**Limitations:**

* The two dimensional array used in the program is statically allocated, which means that there could be wastage of memory or lack of required memory.

**Uses:**

* The program can be used to transpose any matrix containing integer values. The program can find in fields of data science in which transpose is a necessary operation for other operations like finding inverse of a matrix.

**Future Scope:**

* The two dimensional array used in the program can be allocated dyamically by the use of an array of pointers.

**Teacher’s Signature**