**ASSIGNMENT – 16**

**1.PROBLEM STATEMENT**

Write a program in C to dynamically allocate a two dimensional array

**2.ALGORITHMS**

Algorithm **Create\_2D\_Array**

**Input:** The required dimensions ‘row’ and ‘col’ of the two dimensional array.

**Output:** The pointer to the two dimensional array allocate dynamically.

**Remarks:** Dynamic allocation takes place using an array of pointers.

**Steps:**

1. An array of pointers named ‘arr’ with size equal to row, is dyamically allocated in heap
2. **For**(i=1 to row) **do**
3. The pointer arr[i] is pointed to a integer array of size col dynamically allocated in heap
4. The pointer ‘arr’ is returned.
5. **Stop**

**3.SOURCE CODE**

#include<stdio.h>

#include<stdlib.h>

//function to dynamically allocate a 2-D array using array of pointers

int\*\* creatematrix(int row,int col)

{

int i;

int\*\* arr;

arr=(int\*\*)malloc(row\*sizeof(int\*)); //creating array of pointers of size row

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

arr[i]=(int\*)malloc(col\*sizeof(int));//pointing arr[i] to integer arrays

return arr;

}

//function to take input in a matrix

void getmatrix(int \*\*arr,int row,int col)

{

int i,j;

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

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

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

}

//function to display a matrix

void dispmatrix(int \*\*arr,int row,int col)

{

int i,j;

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

{

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

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

printf("\n");

}

}

int main(void)

{

int\*\* arr;

int row,col;

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

scanf("%d",&row);

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

scanf("%d",&col);

arr=creatematrix(row,col); //dynamic allocation of 2-D array

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

getmatrix(arr,row,col);

printf("Entered elements: \n");

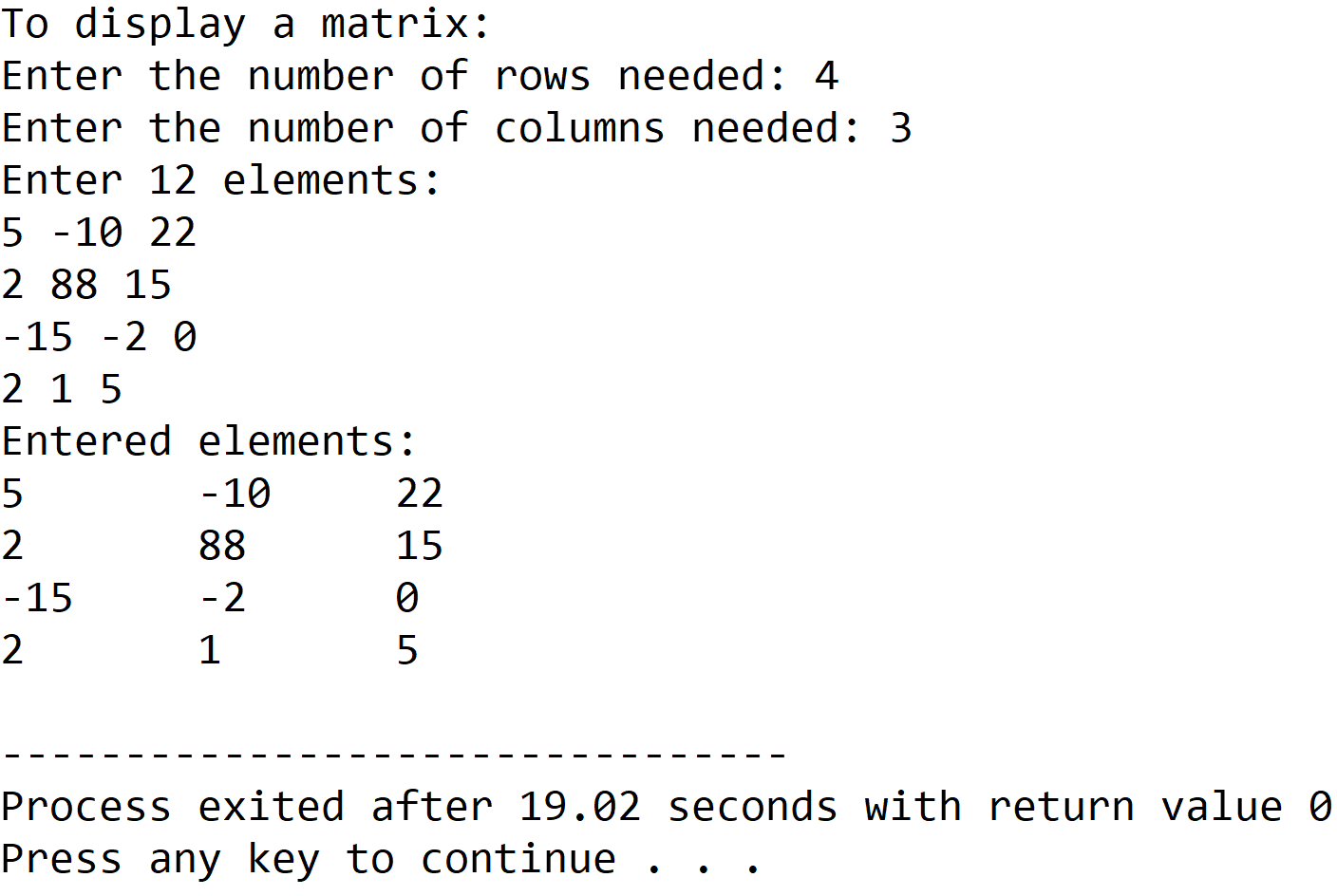
dispmatrix(arr,row,col);

return 0;

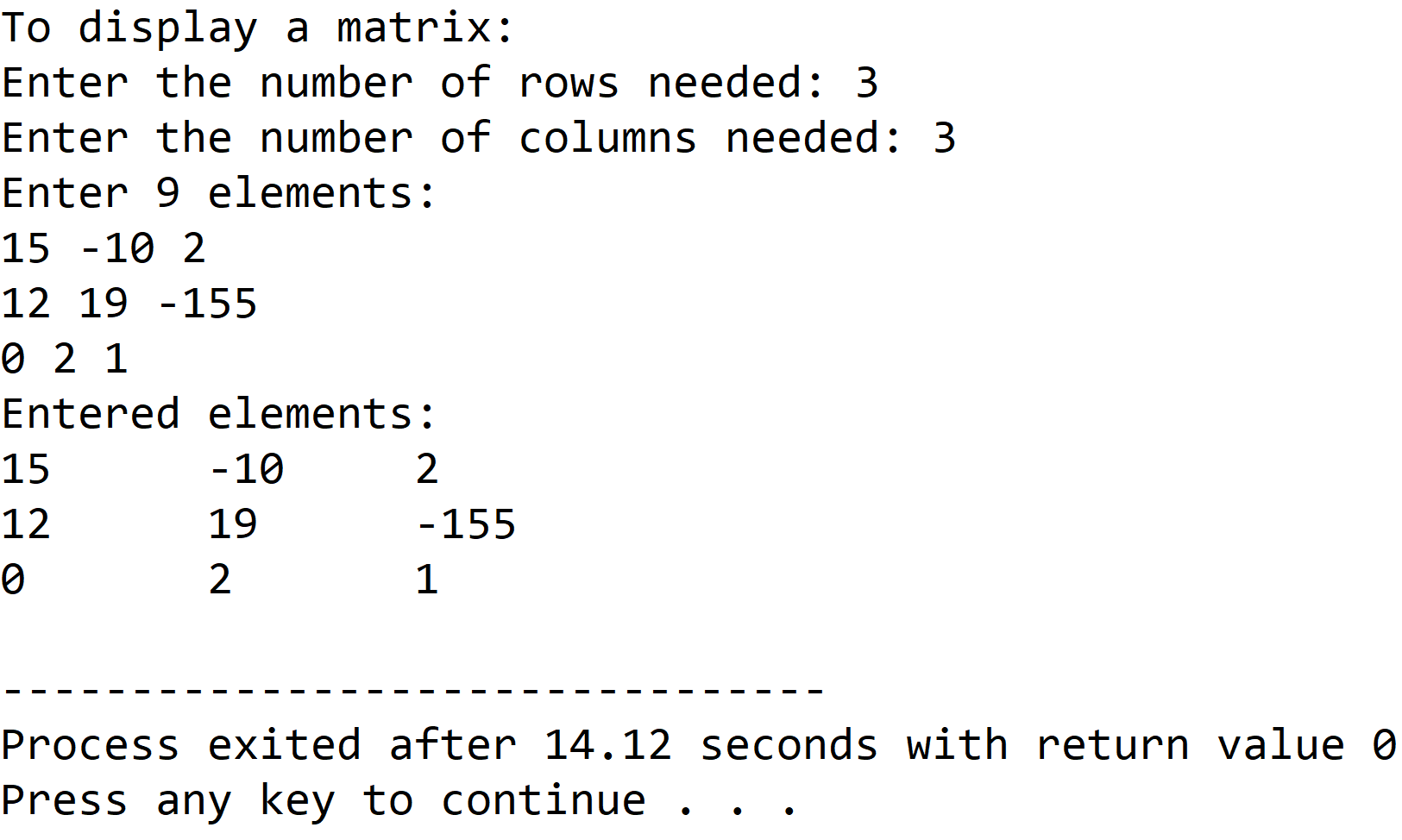
}

**4.OUTPUT**

**SET 1:** Input/Output of a 4x3 matrix



**SET 2 :** Input/Output of a 3x3 matrix



**5.DISCUSSIONS**

**Variable Description:**

* **\*\*arr:** pointer to an integer pointer to hold array of integer pointers.
* **row,col:** dimensions of the 2-D array.
* **i,j:** loop counters.

**Limitations:**

* Even though a dynamic memory allocation procedure has been used in the program, the size of the 2-D array cannot be changed once it is constructed in the memory, thus the array still presents itself as a static data structure.

**Uses:**

* Traditional syntax for creating a 2-D array involves a single 1-D array storing all the elements using a specific indexing scheme. This program shows how a 2-D array can be allocated dynamically with the size being determined at runtime.

**Future Scope:**

* The integer arrays used in the program can be replaced with a linkedlist to eliminate the dependency of the program on contiguous blocks of memory.

**Teacher’s Signature**