```c
//sample code for dynamic 2D array (matrix)
// to run this code:
//      1. gcc -o d dmatrix.c
//      2. ./d <command line argument>
//              e.g. ./d 2
//                   1 2
//                   3 4

#include<stdio.h>
#include<malloc.h>
#include<stdlib.h>
//#define N 5

//void init(int a[N][N]){ // VALID
//void init(int a[][]){ //INVALID
void init(int **a,int N){ //VALID
        int i,j;

        for(i=0;i<N;i++)
                for(j=0;j<N;j++)
                        a[i][j]=N*i+j+1;
}

void output(int *a[],int N){
        int i,j;

        for(i=0;i<N;i++){
                for(j=0;j<N;j++)
                        printf("%3i ",a[i][j]);
                printf("\n");
        }
}
/*
void output2(int a[N][N]){
        int i,j;
        int *b;

        b=(int *)a; //type cast
        for(i=0;i<N*N;i++){
                printf("%3i\n",b[i]);
                //if (i%N==N-1) printf("\n");
        }
}
*/

int main(int argc, char *  argv[]){
        //int m[N][N]; //static allocation
        int i;
        int **m;
        int N;

        //print all command line
        //for(i=0;i<argc;i++)
        //      printf("%s\n",argv[i]);

        //convert second command line argument to int
        //e.g. ./d 5
        //convert "5" to integer 5
        N=atoi(argv[1]);

        //printf("%i\n",sizeof(int *)); //try this at home -2hours FB

        //dynamic allocation
        m=(int **)malloc(N*sizeof(int *));
        for(i=0;i<N;i++)
                m[i]=(int *)malloc(N*sizeof(int));

        init(m,N);
        output(m,N);
        //output2(m); //can't treat the dynamic matrix as a 1D array

        //dynamic deallocation
        for(i=0;i<N;i++)
                free(m[i]);
        free(m);

}
```