\*\*Exercise Home Number 3\*\*   
\*\*Study Material: Pointers\*\*   
\*\*Submission Deadline: 15.12.24\*\*   
  
\*\*Exercise Description:\*\*   
In this exercise, you will create a set of functions that manipulate dynamically allocated arrays using pointers. You will work with a 2D array (matrix) of integers, and the aim is to implement functions that perform various operations on this matrix with the help of pointers.   
  
### Function Descriptions:  
  
1. `int\*\* createMatrix(int rows, int cols);`   
 This function will dynamically allocate memory for a 2D array (matrix) of integers with the specified number of `rows` and `cols`. It will return a pointer to the allocated 2D array.  
  
2. `void fillMatrix(int\*\* matrix, int rows, int cols);`   
 This function will fill the 2D array with integers. For simplicity, the values can be set to be the sum of their row and column indices (i + j).  
  
3. `void printMatrix(int\*\* matrix, int rows, int cols);`   
 This function will print the contents of the matrix in a formatted way.  
  
4. `void freeMatrix(int\*\* matrix, int rows);`   
 This function will deallocate the memory used for the matrix.  
  
### Example Implementation:   
  
- If `createMatrix(3, 4)` is called, it will create a 3x4 matrix.  
- If the `fillMatrix` function is called afterward, the resulting matrix will look like this:  
```  
0 1 2 3  
1 2 3 4  
2 3 4 5  
```  
- If you then call `printMatrix`, it will output the following:  
```  
0 1 2 3  
1 2 3 4  
2 3 4 5  
```  
  
### Instructions:  
  
1. Define the following constants at the top of your file:  
 ```c  
 #define MAX\_ROWS 50  
 #define MAX\_COLS 50  
 ```  
  
2. Assume input values for rows and cols are always within the bounds of MAX\_ROWS and MAX\_COLS.  
  
3. Write unit tests to confirm the correctness of your implemented functions, including edge cases like an empty matrix.  
  
4. Only submit the code for the four specified functions. You may add auxiliary functions as necessary but ensure to submit only the specified functions.  
  
5. Make sure your code passes all the pre-submission tests in the VPL system before submission. Exercises that fail pre-submission tests will receive an automatic score of 0.  
  
### Good luck!