**Introduction**

Matrices are used in many different areas of math. They are useful for calculating changes in two-dimensional motion and in making changes to large groups of data. This method of storage is compact, reliable, and versatile, as it can be converted into many different forms. The matrix multiplication system will able to provide results of any desired matrix size. Mainly the project focuses on matrix operations on square matrices. The operations performed on a pair of square matrices are, addition, subtraction and multiplication. The operations performed on a single square matrix are transpose, determinant, inverse and to check the type of matrix be it scaler, triangular, orthogonal, diagonal, zero, unit or symmetric.

**Features**

* Dynamic memory allocation for the input matrices leading to judicial usage of resources.
* Allows the input matrices of any order n.
* The user can select the operation to be performed by pressing a number on the keyboard. The significance of each number, i.e the operation which it performs will be displayed on the screen during the execution of the program.
* Operations carried on a single input matrix are:
  + Check the type of input matrix be it scaler, triangular, orthogonal, diagonal, zero, unit or symmetric.
  + Determinant
  + Inverse
  + Transpose
* Operations carried on two input matrices are:
  + Addition
  + Subtraction
  + Multiplication

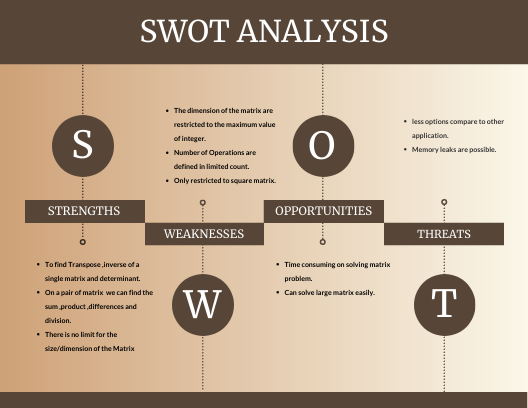
**Functional Requirements**

1. Menu-Driven program.
2. Feasible on Windows and Linux.
3. Software required-GCC compiler.
4. Language-C.

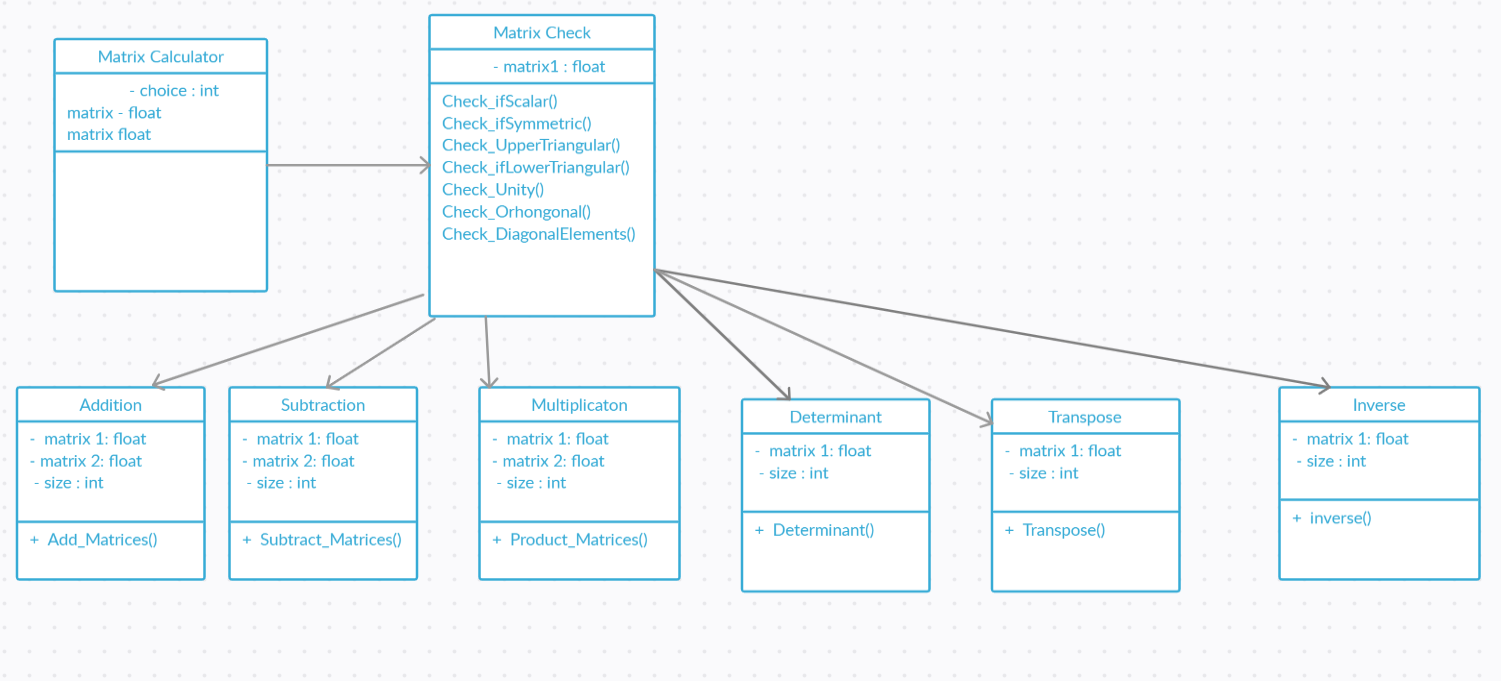
**Non Functional Requirements**

1. Quick response.
2. User friendly.
3. Portable.
4. Good Performance.

**SWOT Analysis**

****

**Design**

****

Class diagram

**Implementation**

* A menu driven program asking the user to enter any option given from the menu.
* Then the user is asked to enter any order n for the input matrix provided order should be greater than 1 or else it will ask to renter the order.
* If the option selected is apart from inverse, type, transpose, determinant then the user is asked to enter one more matrix.
* Then the required function is called and processed.
* The function returns success if it gets executed successfully or else returns the error.

For example, inverse function returns the error if determinant of the matrix is zero.

* The loop of menu continues till the user askes opts for the exit.
* For implementing modularity, we have created separate files each for function prototypes, their implementation and integration.
* Then the code is tested using various test cases to check whether the functions are working properly or not.

**Test Plan**

## High level test plan

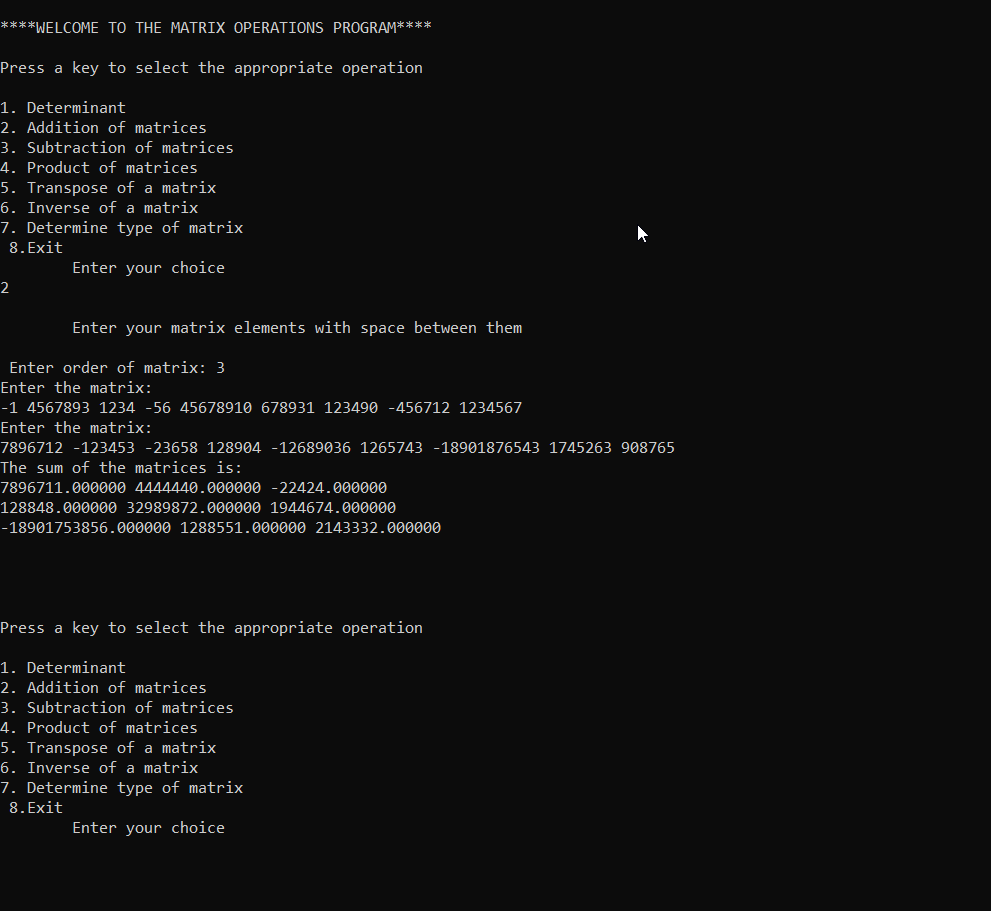
| **Test ID** | **Description** | **Expected Input** | **Expected Output** | **Actual Output** | **Type Of Test** |
| --- | --- | --- | --- | --- | --- |
| H\_01 | Addition of two matrices | matrix1, matrix2, n | SUCCESS | SUCCESS | Technical |
| H\_02 | Subtraction of two matrices | matrix1, matrix2, n | SUCCESS | SUCCESS | Technical |
| H\_03 | Multiplication of two matrices | matrix1, matrix2, n | SUCCESS | SUCCESS | Technical |
| H\_04 | Determinant of a matrix | matrix1, n | SUCCESS | SUCCESS | Technical |
| H\_05 | Transpose of a matrix | matrix1, n | SUCCESS | SUCCESS | Technical |
| H\_06 | Inverse of a matrix | matrix1, n | SUCCESS | SUCCESS | Technical |
| H\_07 | Power of -1 function | 3 | -1 | -1 | Technical |
| H\_08 | Power of -1 function | 2 | 1 | 1 | Technical |
| H\_09 | Determine type of a matrix | matrix1, n | SUCCESS | SUCCESS | Technical |

## Low level test plan

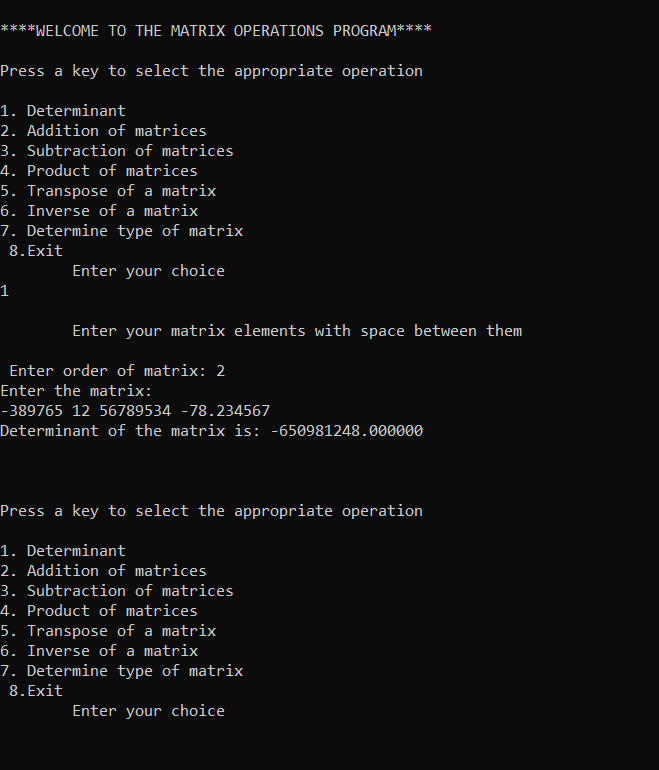
| **Test ID** | **HLT ID** | **Description** | **Expected Input** | **Expected Output** | **Actual Output** | **Type Of Test** |
| --- | --- | --- | --- | --- | --- | --- |
| L\_01 | H\_01, H\_02, H\_03 | Tested on functions which accept two matrices as input | matrix1, matrix2, n | SUCCESS | SUCCESS | Technical |
| L\_02 | H\_04, H\_05, H\_06, H\_09 | Tested on functions which accept single matrix as input | matrix1, n | SUCCESS | SUCCESS | Technical |
| L\_03 | H\_07, H\_08 | Tested on function which returns an integer value | 3 | -1 | -1 | Technical |

**Output**

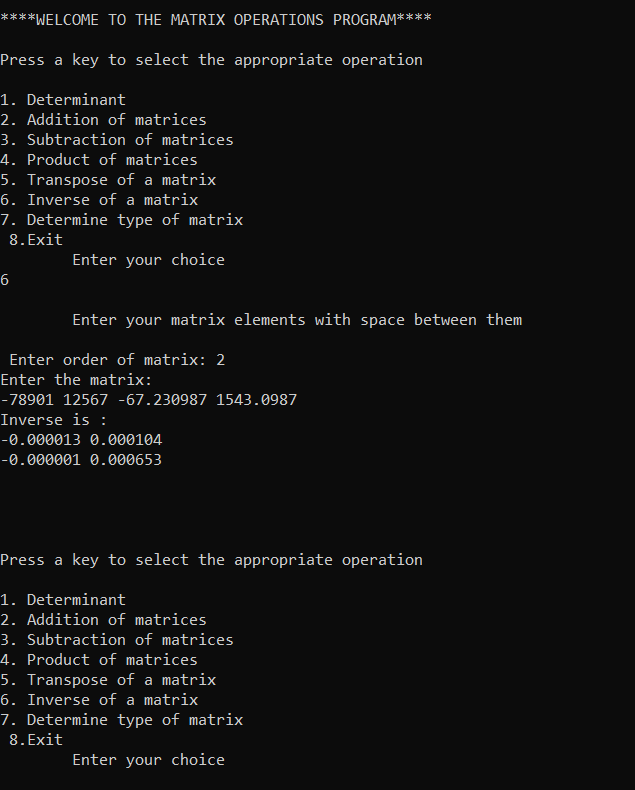
1. Addition

****

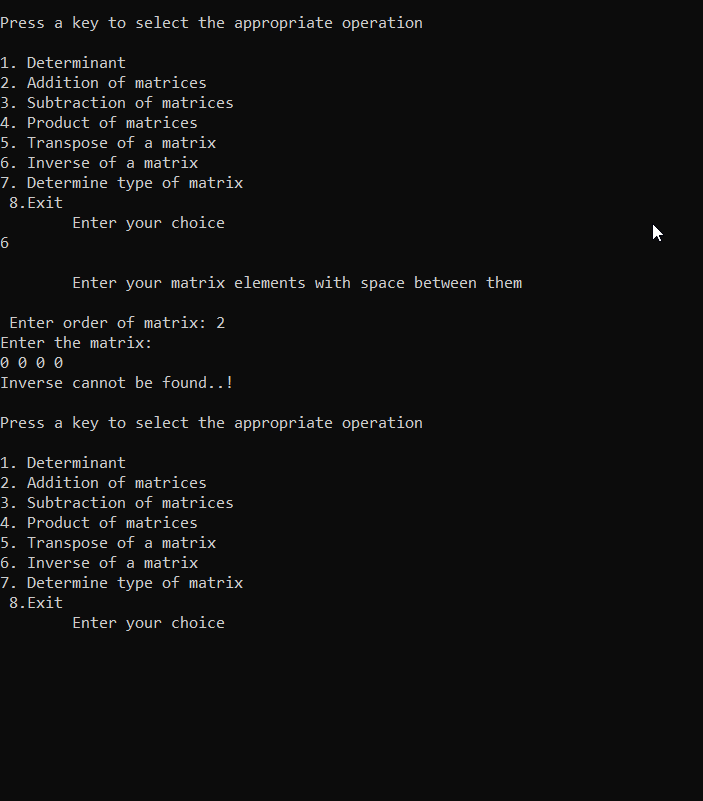
1. Determinant

****

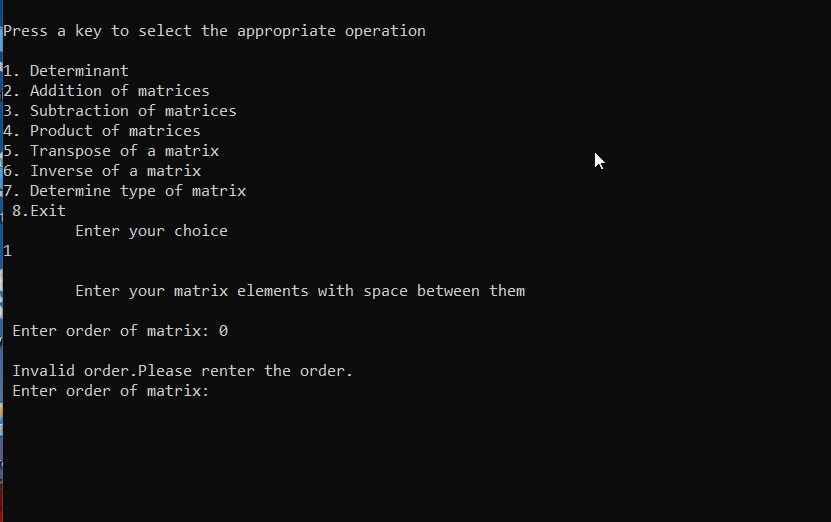
1. Inverse

****

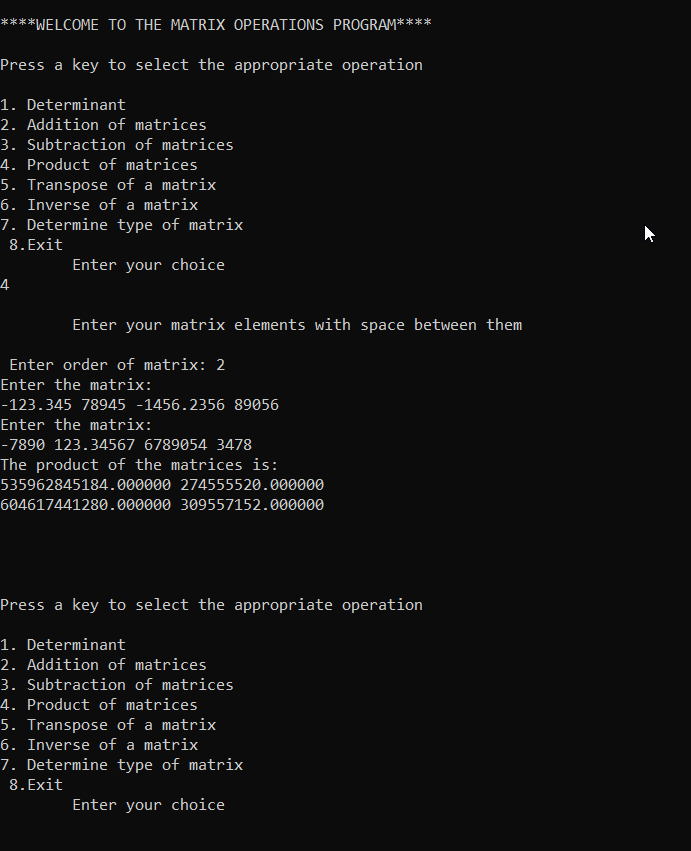
1. Inverse

****

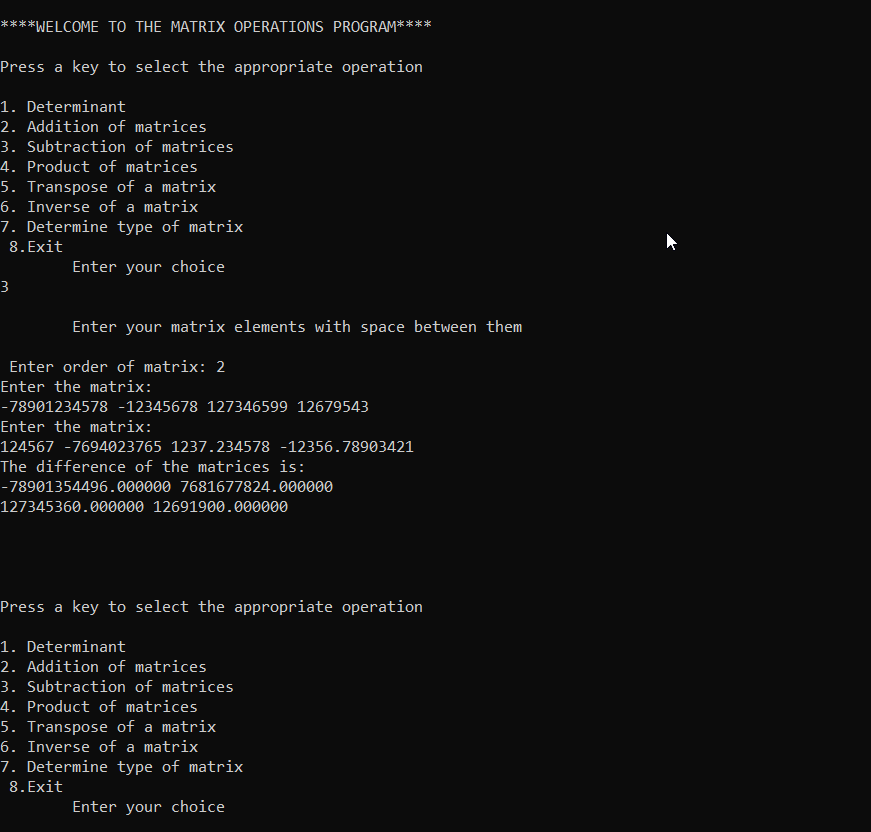
1. Checking the order of matrix

****

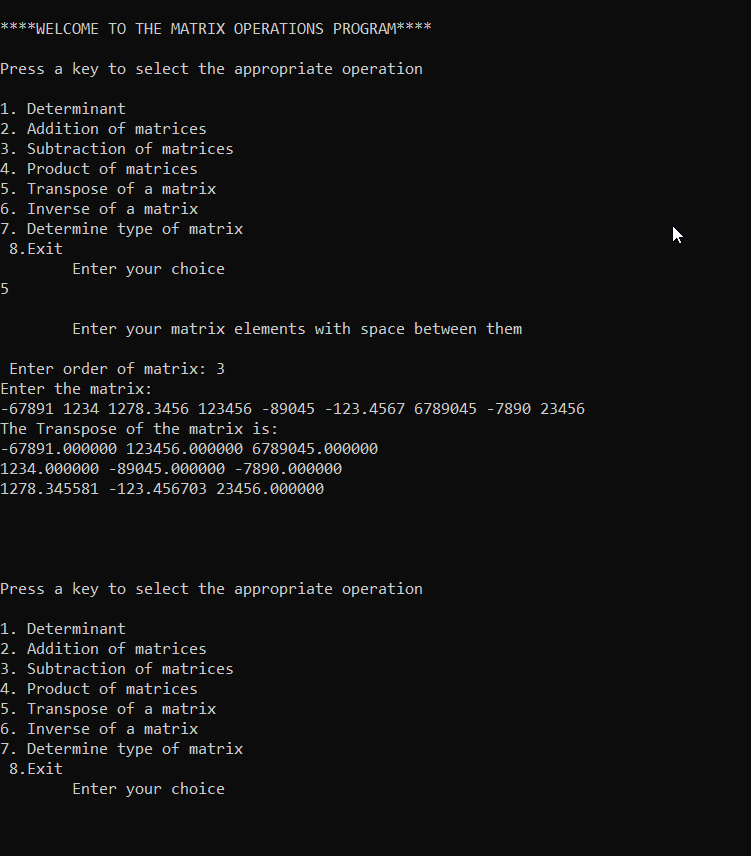
1. Product of matrices

****

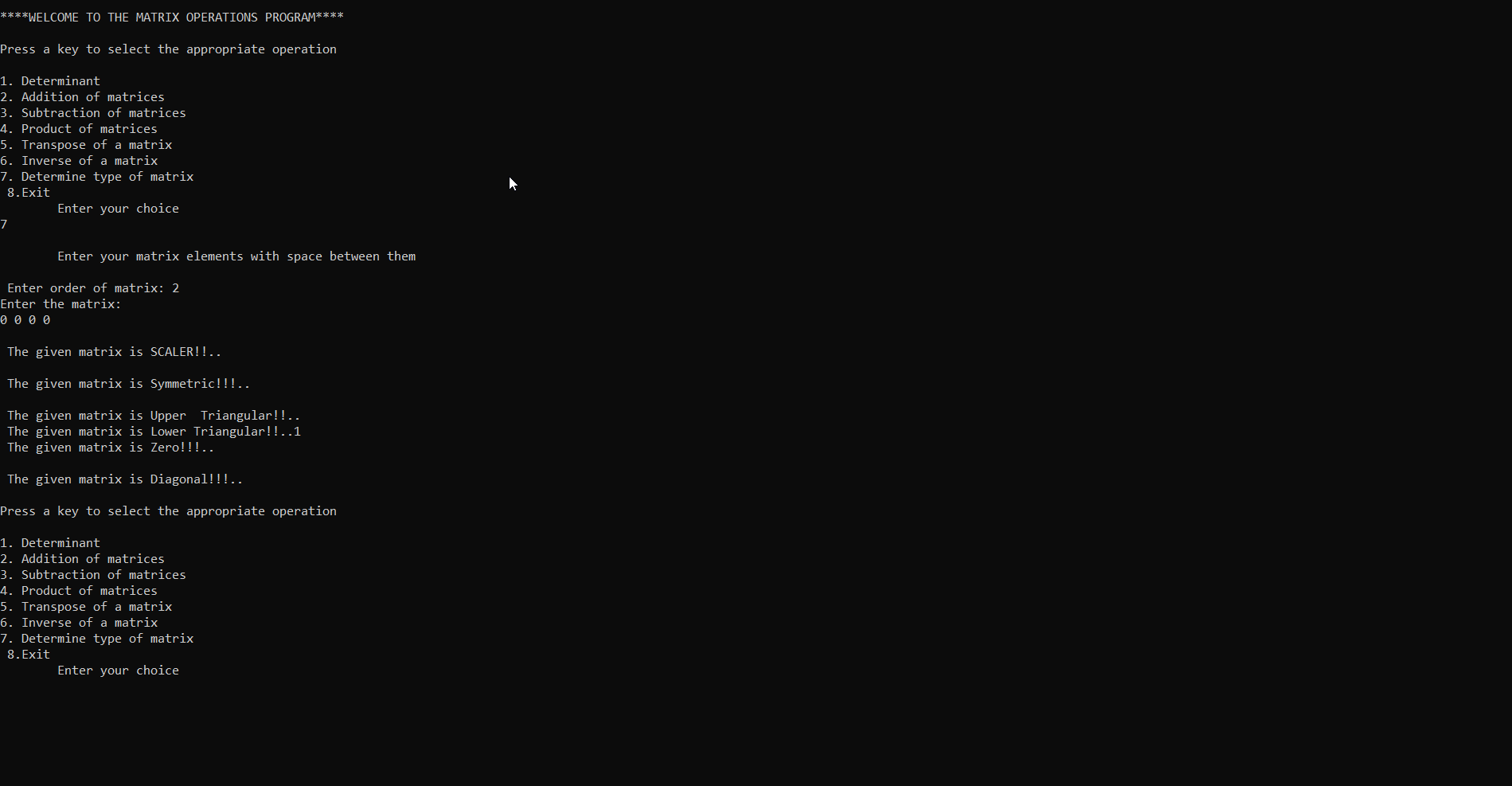
1. Subtraction

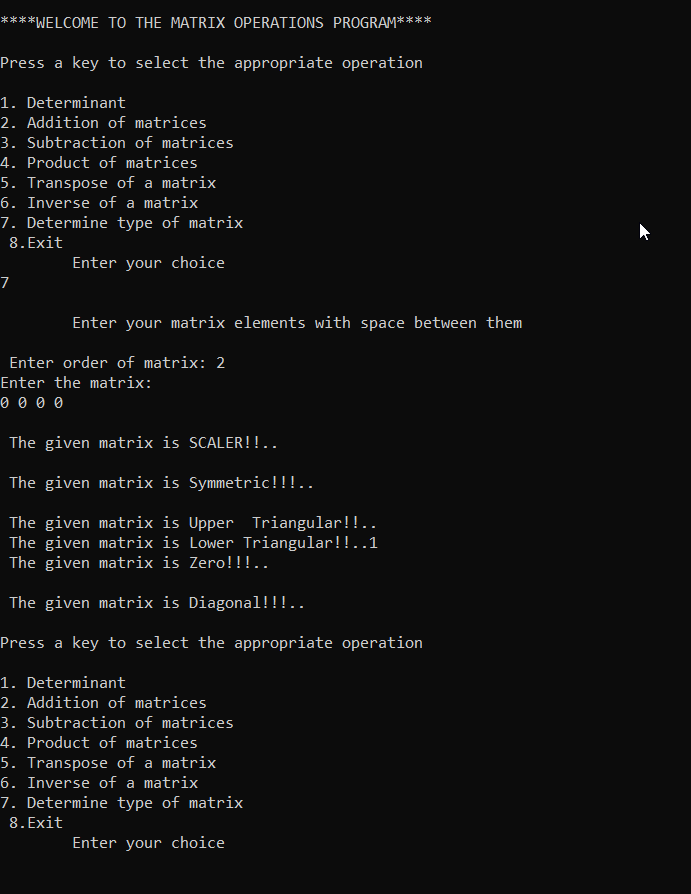
****

1. Transpose of the matrix

****

1. Type of matrix

****

****

**Conclusion**

All the functions were implemented and tested successfully.

-