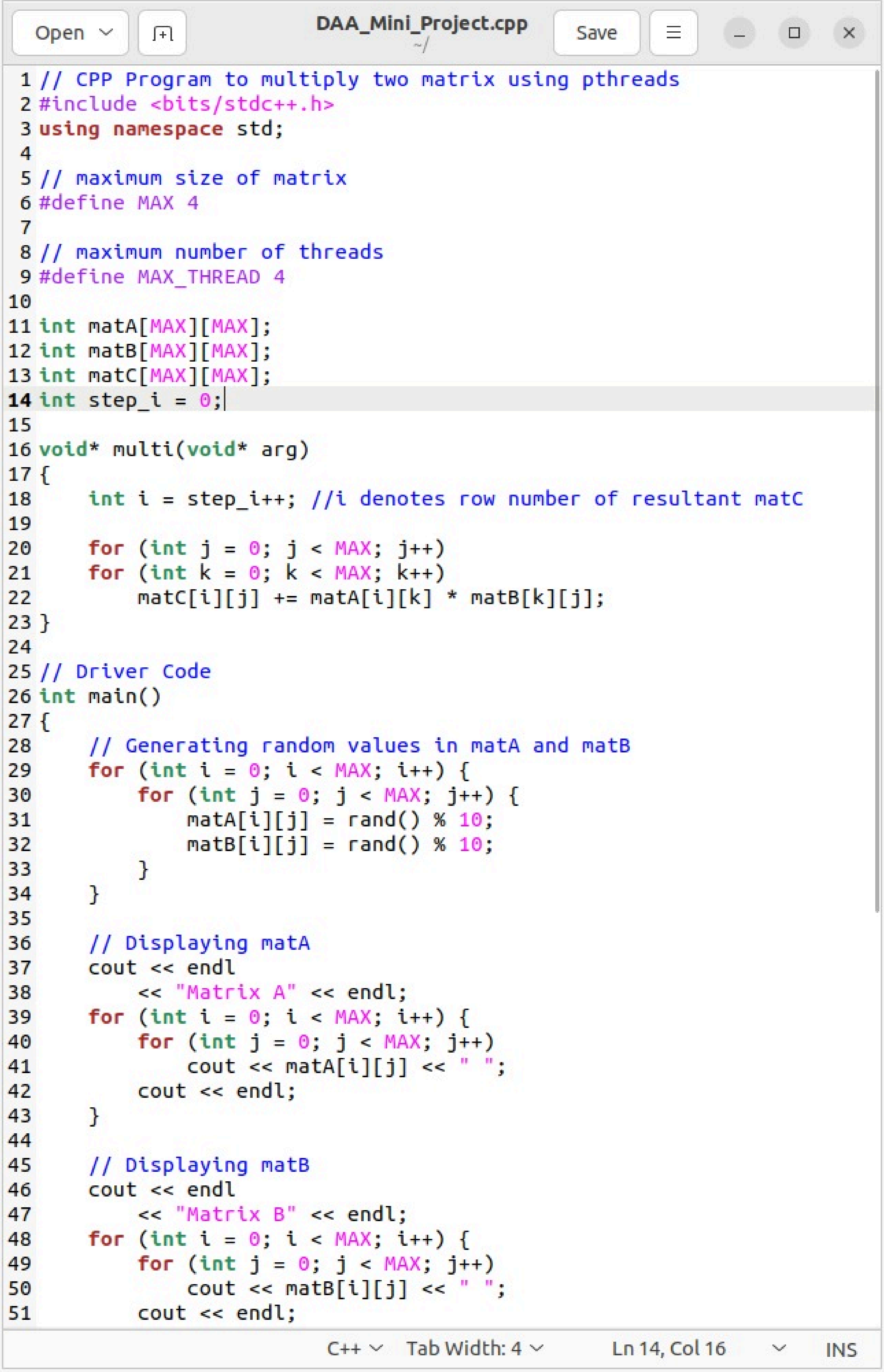
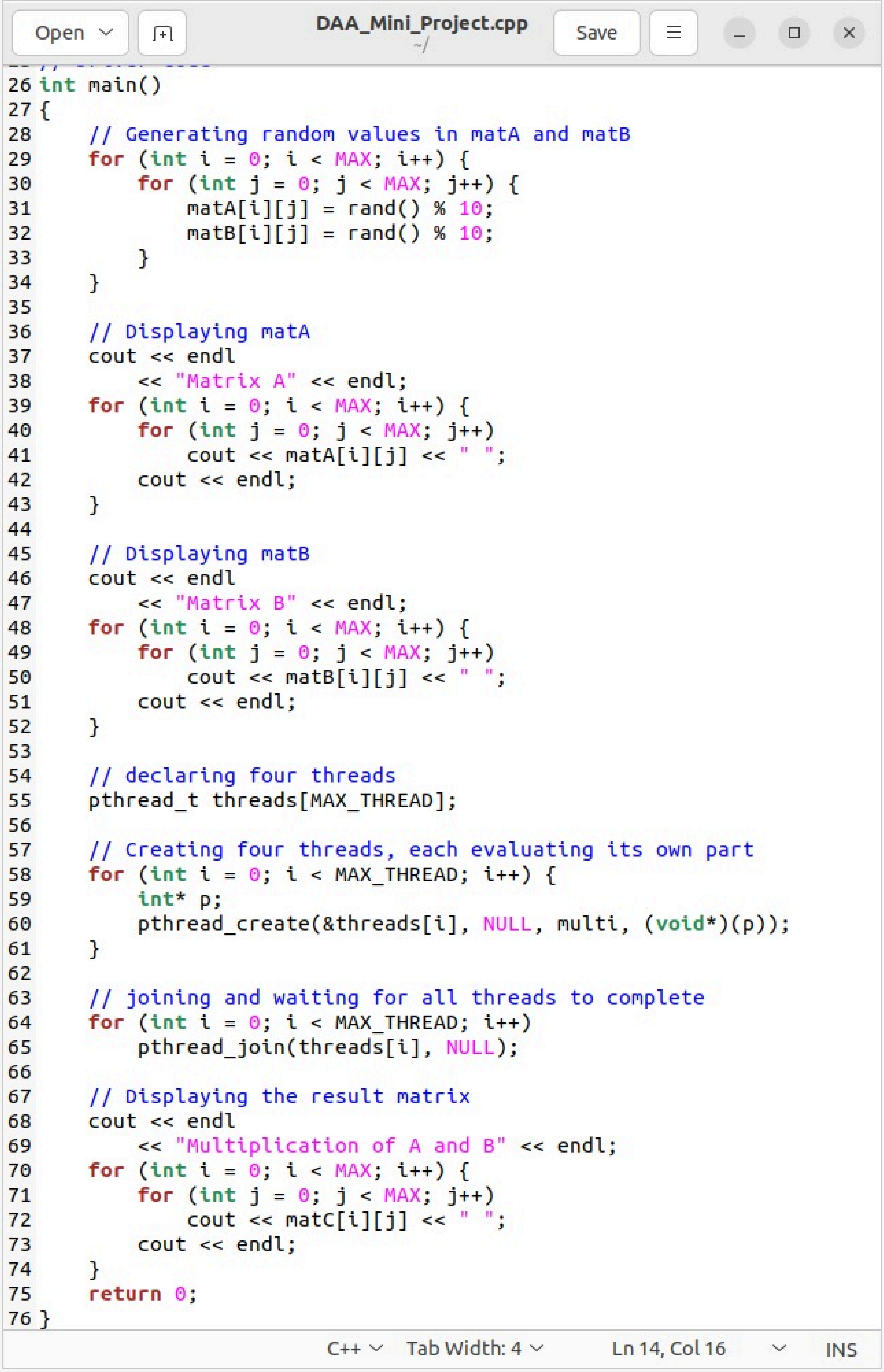
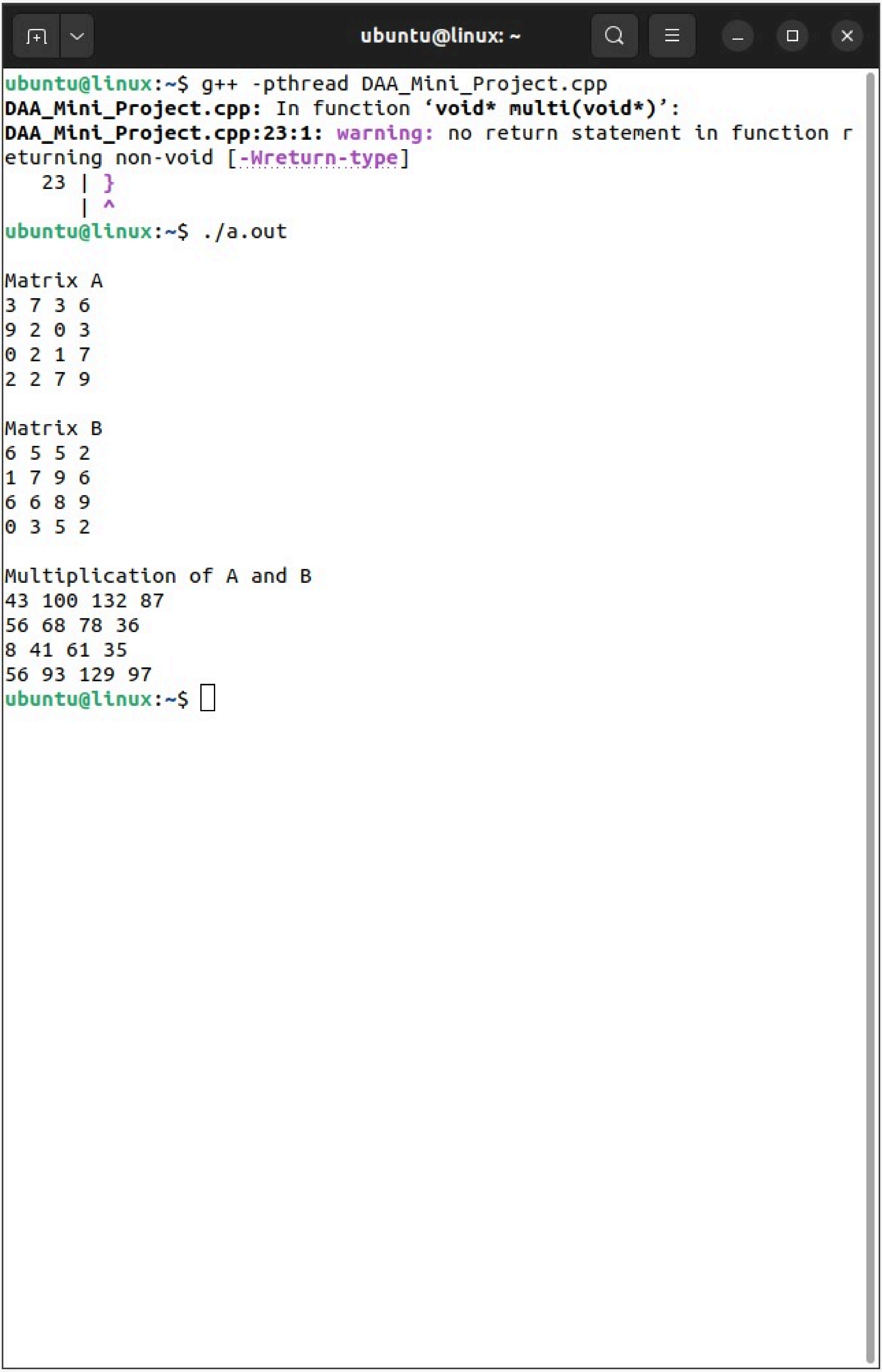


```cpp
// CPP Program to multiply two matrix using pthreads
#include <bits/stdc++.h>
using namespace std;

// maximum size of matrix
#define MAX 4

// maximum number of threads
#define MAX_THREAD 4

int matA[MAX][MAX];
int matB[MAX][MAX];
int matC[MAX][MAX];
int step_i = 0;

void* multi(void* arg)
{
    int i = step_i++; //i denotes row number of resultant matC

    for (int j = 0; j < MAX; j++)
    for (int k = 0; k < MAX; k++)
        matC[i][j] += matA[i][k] * matB[k][j];
}

// Driver Code
int main()
{
    // Generating random values in matA and matB
    for (int i = 0; i < MAX; i++) {
        for (int j = 0; j < MAX; j++) {
            matA[i][j] = rand() % 10;
            matB[i][j] = rand() % 10;
        }
    }

    // Displaying matA
    cout << endl
        << "Matrix A" << endl;
    for (int i = 0; i < MAX; i++) {
        for (int j = 0; j < MAX; j++)
            cout << matA[i][j] << " ";
        cout << endl;
    }

    // Displaying matB
    cout << endl
        << "Matrix B" << endl;
    for (int i = 0; i < MAX; i++) {
        for (int j = 0; j < MAX; j++)
            cout << matB[i][j] << " ";
        cout << endl;
```



```cpp
int main()
{
    // Generating random values in matA and matB
    for (int i = 0; i < MAX; i++) {
        for (int j = 0; j < MAX; j++) {
            matA[i][j] = rand() % 10;
            matB[i][j] = rand() % 10;
        }
    }

    // Displaying matA
    cout << endl
        << "Matrix A" << endl;
    for (int i = 0; i < MAX; i++) {
        for (int j = 0; j < MAX; j++)
            cout << matA[i][j] << " ";
        cout << endl;
    }

    // Displaying matB
    cout << endl
        << "Matrix B" << endl;
    for (int i = 0; i < MAX; i++) {
        for (int j = 0; j < MAX; j++)
            cout << matB[i][j] << " ";
        cout << endl;
    }

    // declaring four threads
    pthread_t threads[MAX_THREAD];

    // Creating four threads, each evaluating its own part
    for (int i = 0; i < MAX_THREAD; i++) {
        int* p;
        pthread_create(&threads[i], NULL, multi, (void*)(p));
    }

    // joining and waiting for all threads to complete
    for (int i = 0; i < MAX_THREAD; i++)
        pthread_join(threads[i], NULL);

    // Displaying the result matrix
    cout << endl
        << "Multiplication of A and B" << endl;
    for (int i = 0; i < MAX; i++) {
        for (int j = 0; j < MAX; j++)
            cout << matC[i][j] << " ";
        cout << endl;
    }
    return 0;
}
```



```
ubuntu@linux:~$ g++ -pthread DAA_Mini_Project.cpp
DAA_Mini_Project.cpp: In function ‘void* multi(void*)’:
DAA_Mini_Project.cpp:23:1: warning: no return statement in function r
eturning non-void [-Wreturn-type]
   23 | }
      | ^
ubuntu@linux:~$ ./a.out

Matrix A
3 7 3 6
9 2 0 3
0 2 1 7
2 2 7 9

Matrix B
6 5 5 2
1 7 9 6
6 6 8 9
0 3 5 2

Multiplication of A and B
43 100 132 87
56 68 78 36
8 41 61 35
56 93 129 97
ubuntu@linux:~$ 
```