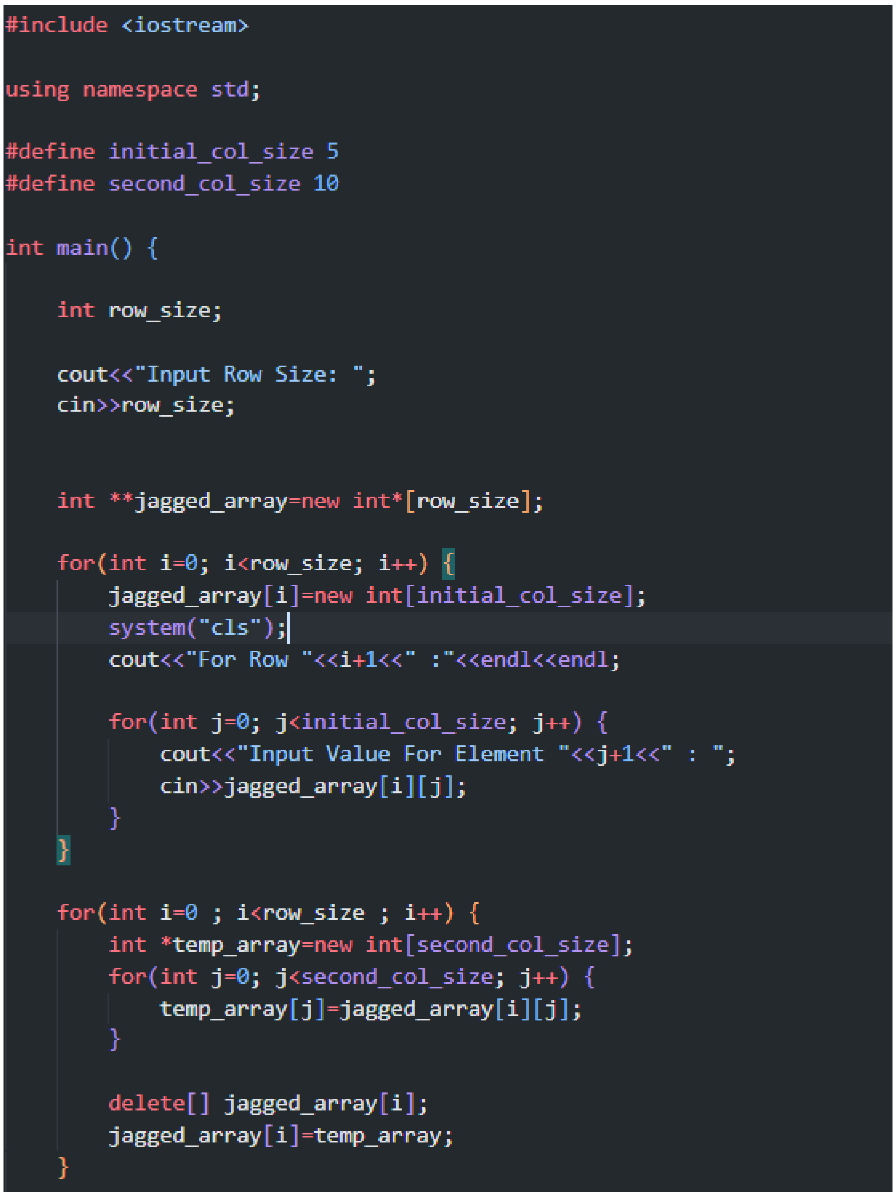
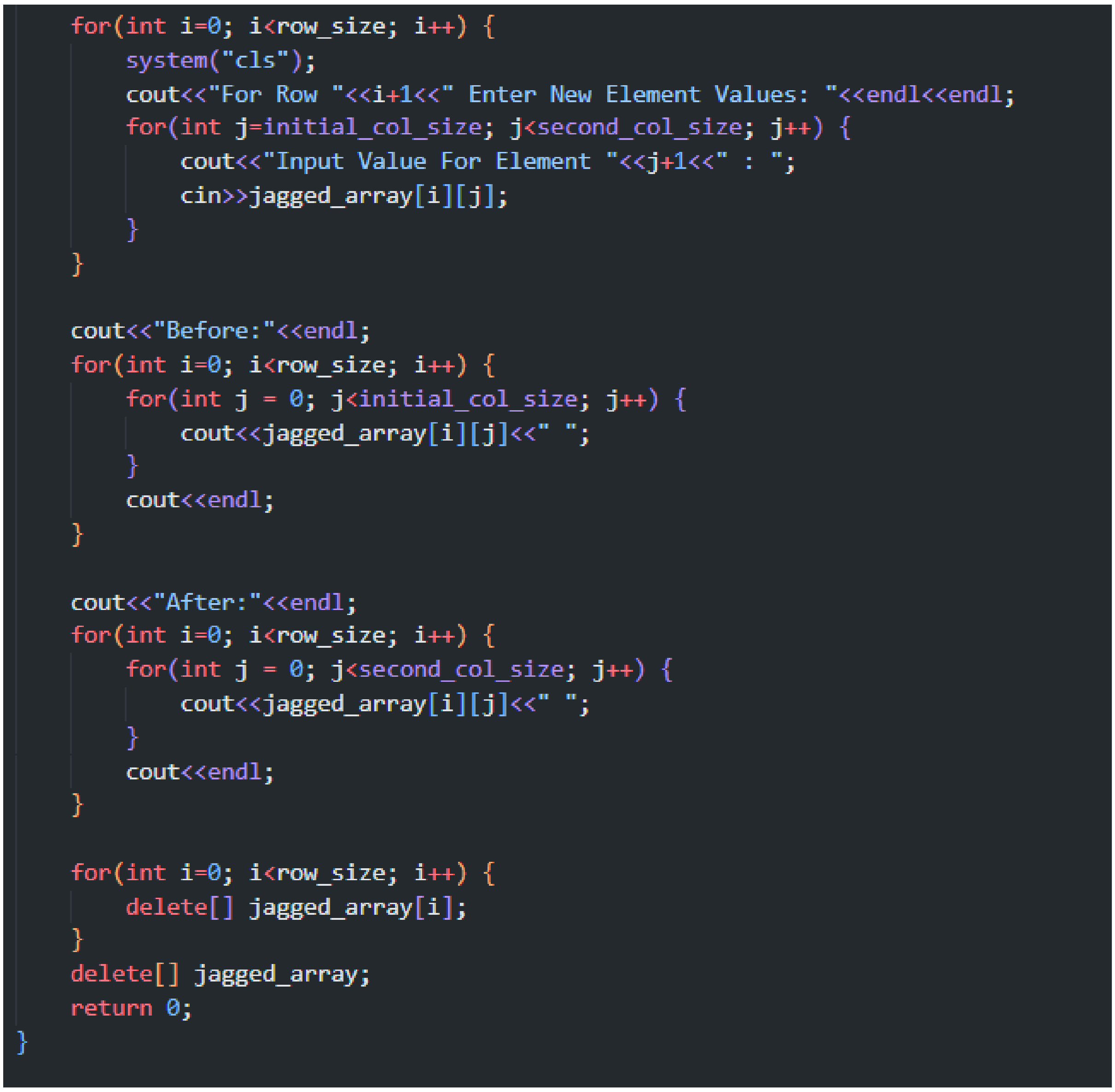
TASK 01:

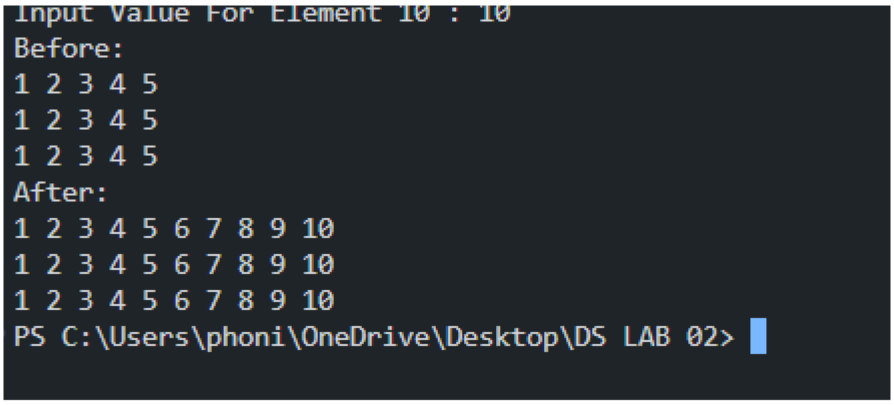


```
#include <iostream>

using namespace std;

#define initial_col_size 5
#define second_col_size 10

int main() {

    int row_size;

    cout<<"Input Row Size: ";
    cin>>row_size;


    int **jagged_array=new int*[row_size];

    for(int i=0; i<row_size; i++) {
        jagged_array[i]=new int[initial_col_size];
        system("cls");
        cout<<"For Row "<<i+1<<" :"<<endl<<endl;

        for(int j=0; j<initial_col_size; j++) {
            cout<<"Input Value For Element "<<j+1<<" : ";
            cin>>jagged_array[i][j];
        }
    }

    for(int i=0 ; i<row_size ; i++) {
        int *temp_array=new int[second_col_size];
        for(int j=0; j<second_col_size; j++) {
            temp_array[j]=jagged_array[i][j];
        }

        delete[] jagged_array[i];
        jagged_array[i]=temp_array;
    }
```



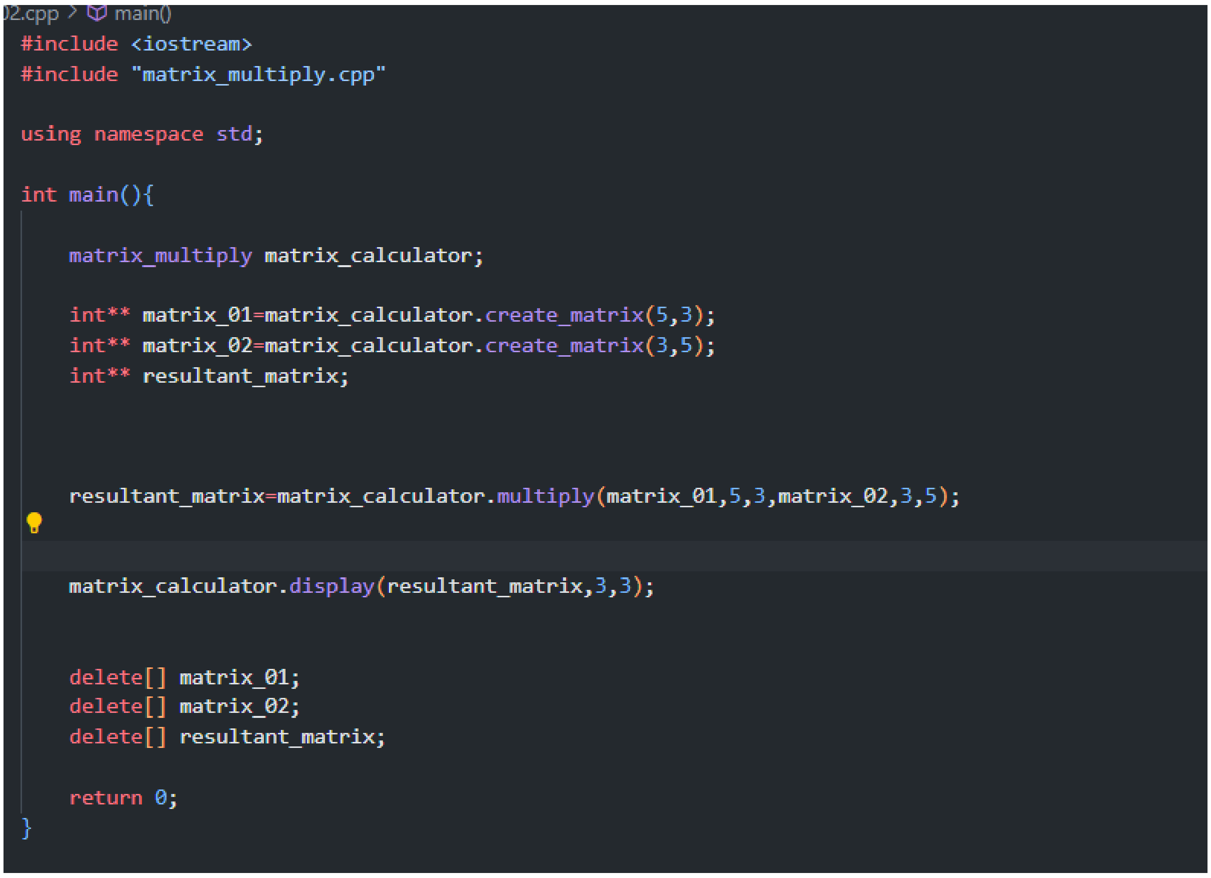
```cpp
    for(int i=0; i<row_size; i++) {
        system("cls");
        cout<<"For Row "<<i+1<<" Enter New Element Values: "<<endl<<endl;
        for(int j=initial_col_size; j<second_col_size; j++) {
            cout<<"Input Value For Element "<<j+1<<" : ";
            cin>>jagged_array[i][j];
        }
    }

    cout<<"Before:"<<endl;
    for(int i=0; i<row_size; i++) {
        for(int j = 0; j<initial_col_size; j++) {
            cout<<jagged_array[i][j]<<" ";
        }
        cout<<endl;
    }

    cout<<"After:"<<endl;
    for(int i=0; i<row_size; i++) {
        for(int j = 0; j<second_col_size; j++) {
            cout<<jagged_array[i][j]<<" ";
        }
        cout<<endl;
    }

    for(int i=0; i<row_size; i++) {
        delete[] jagged_array[i];
    }
    delete[] jagged_array;
    return 0;
}
```

OUTPUT:

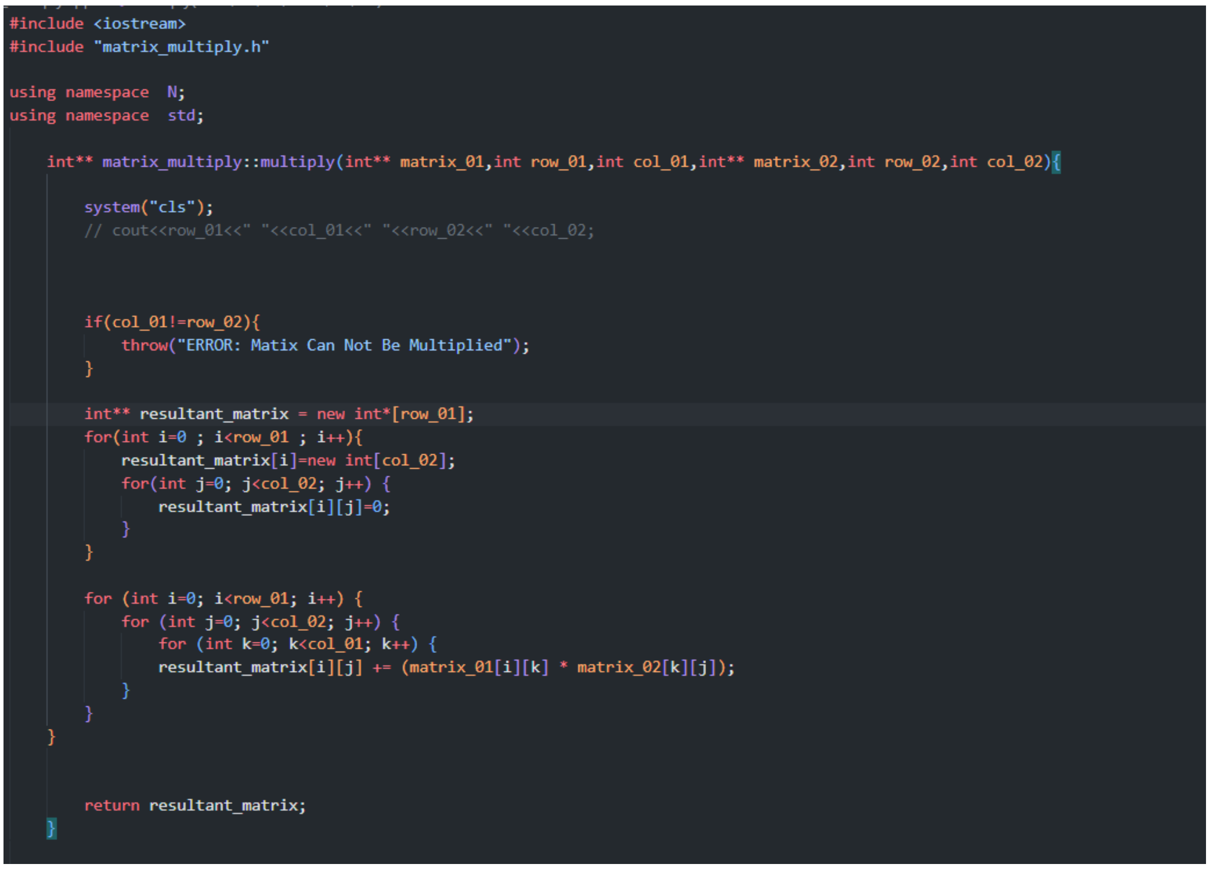


```
Input Value For Element 10 : 10
Before:
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
After:
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10
PS C:\Users\phoni\OneDrive\Desktop\DS LAB 02>
```

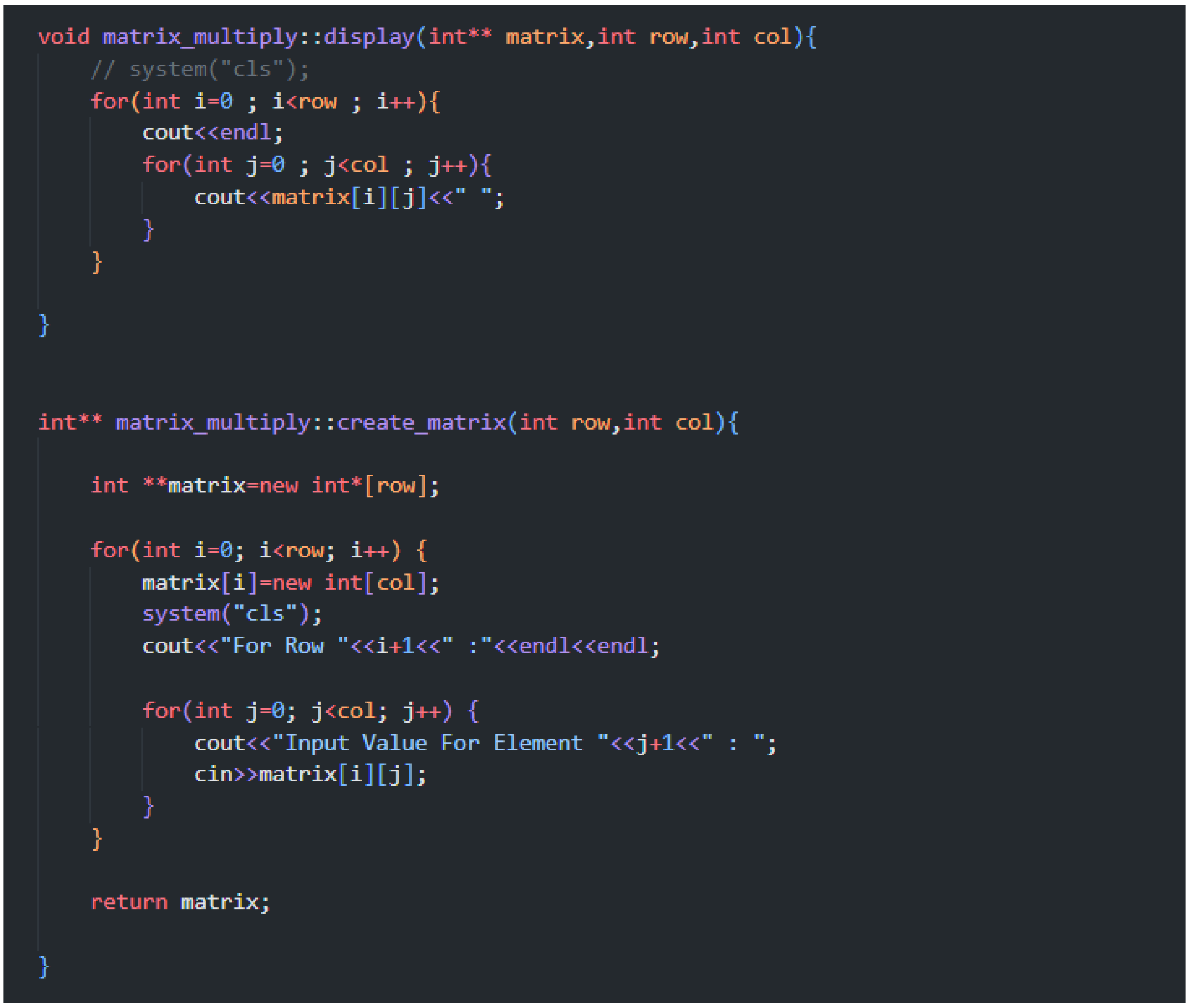
TASK 02:



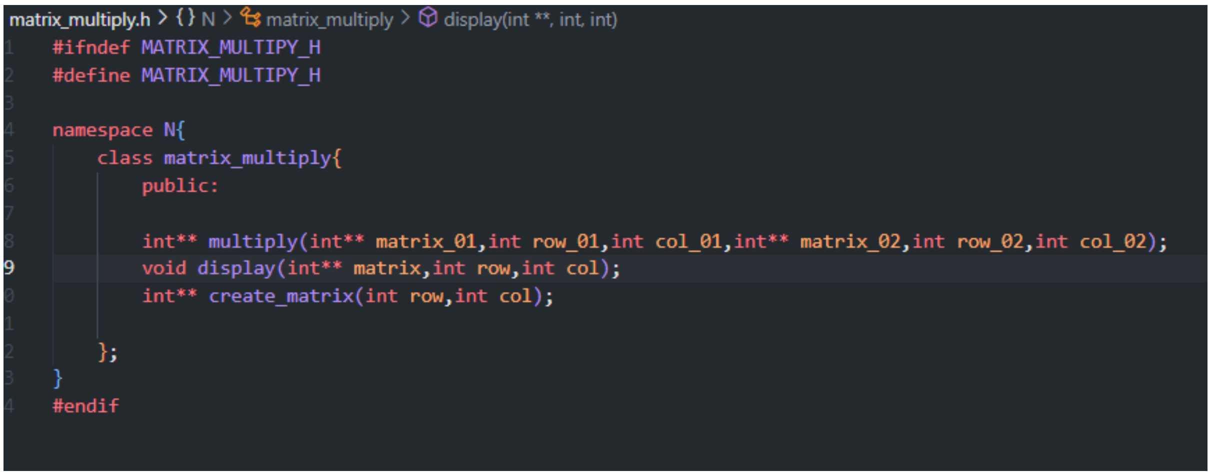
```cpp
#include <iostream>
#include "matrix_multiply.cpp"

using namespace std;

int main(){

    matrix_multiply matrix_calculator;

    int** matrix_01=matrix_calculator.create_matrix(5,3);
    int** matrix_02=matrix_calculator.create_matrix(3,5);
    int** resultant_matrix;



    resultant_matrix=matrix_calculator.multiply(matrix_01,5,3,matrix_02,3,5);


    matrix_calculator.display(resultant_matrix,3,3);


    delete[] matrix_01;
    delete[] matrix_02;
    delete[] resultant_matrix;

    return 0;
}
```



```cpp
#include <iostream>
#include "matrix_multiply.h"

using namespace  N;
using namespace  std;

    int** matrix_multiply::multiply(int** matrix_01,int row_01,int col_01,int** matrix_02,int row_02,int col_02){

        system("cls");
        // cout<<row_01<<" "<<col_01<<" "<<row_02<<" "<<col_02;



        if(col_01!=row_02){
            throw("ERROR: Matix Can Not Be Multiplied");
        }

        int** resultant_matrix = new int*[row_01];
        for(int i=0 ; i<row_01 ; i++){
            resultant_matrix[i]=new int[col_02];
            for(int j=0; j<col_02; j++) {
                resultant_matrix[i][j]=0;
            }
        }

        for (int i=0; i<row_01; i++) {
            for (int j=0; j<col_02; j++) {
                for (int k=0; k<col_01; k++) {
                resultant_matrix[i][j] += (matrix_01[i][k] * matrix_02[k][j]);
            }
        }
    }


        return resultant_matrix;
    }
```

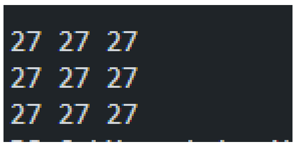


```cpp
void matrix_multiply::display(int** matrix,int row,int col){
    // system("cls");
    for(int i=0 ; i<row ; i++){
        cout<<endl;
        for(int j=0 ; j<col ; j++){
            cout<<matrix[i][j]<<" ";
        }
    }

}


int** matrix_multiply::create_matrix(int row,int col){

    int **matrix=new int*[row];

    for(int i=0; i<row; i++) {
        matrix[i]=new int[col];
        system("cls");
        cout<<"For Row "<<i+1<<" :"<<endl<<endl;

        for(int j=0; j<col; j++) {
            cout<<"Input Value For Element "<<j+1<<" : ";
            cin>>matrix[i][j];
        }
    }

    return matrix;

}
```


matrix_multiply.h > {} N > matrix_multiply > display(int **, int, int)

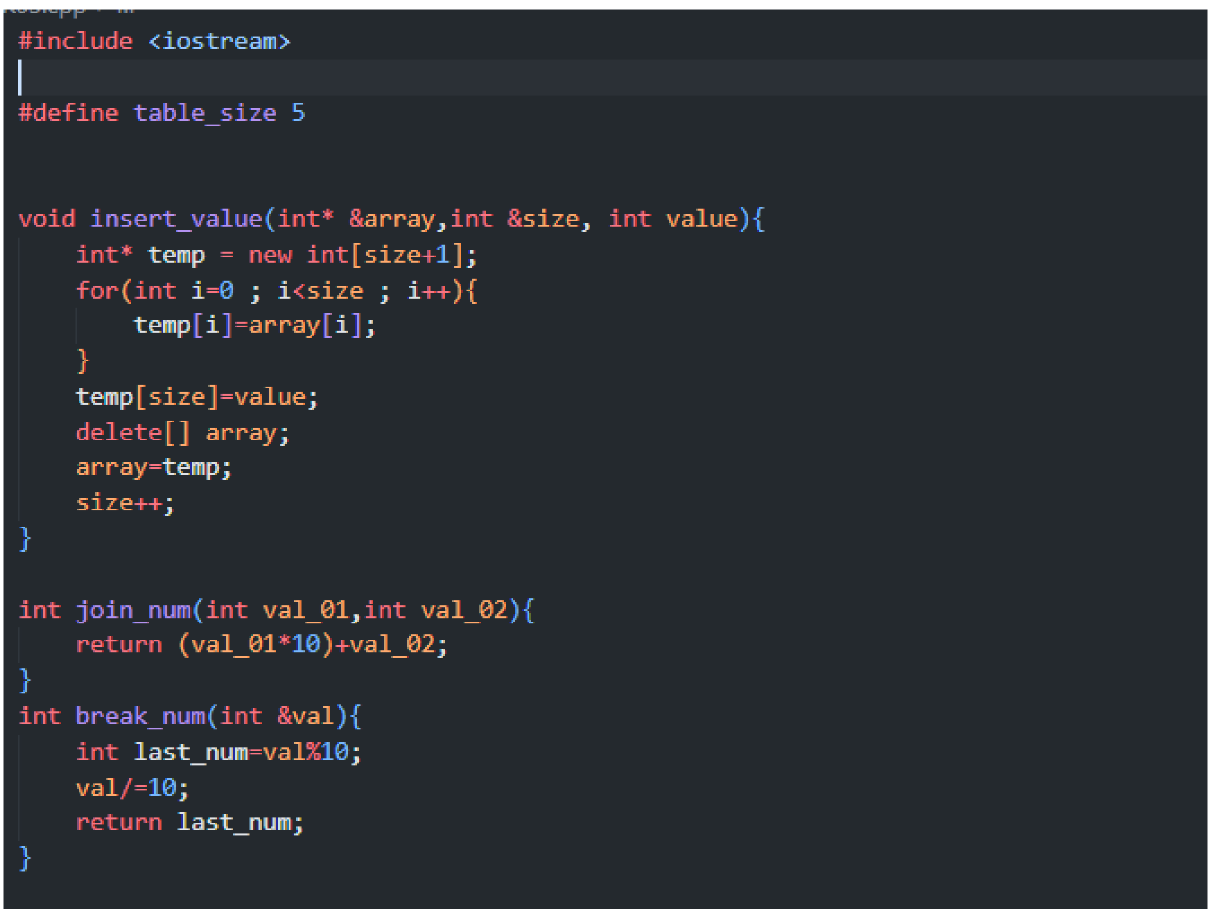
```cpp
#ifndef MATRIX_MULTIPY_H
#define MATRIX_MULTIPY_H

namespace N{
    class matrix_multiply{
        public:

        int** multiply(int** matrix_01,int row_01,int col_01,int** matrix_02,int row_02,int col_02);
        void display(int** matrix,int row,int col);
        int** create_matrix(int row,int col);

    };
}
#endif
```

OUTPUT:

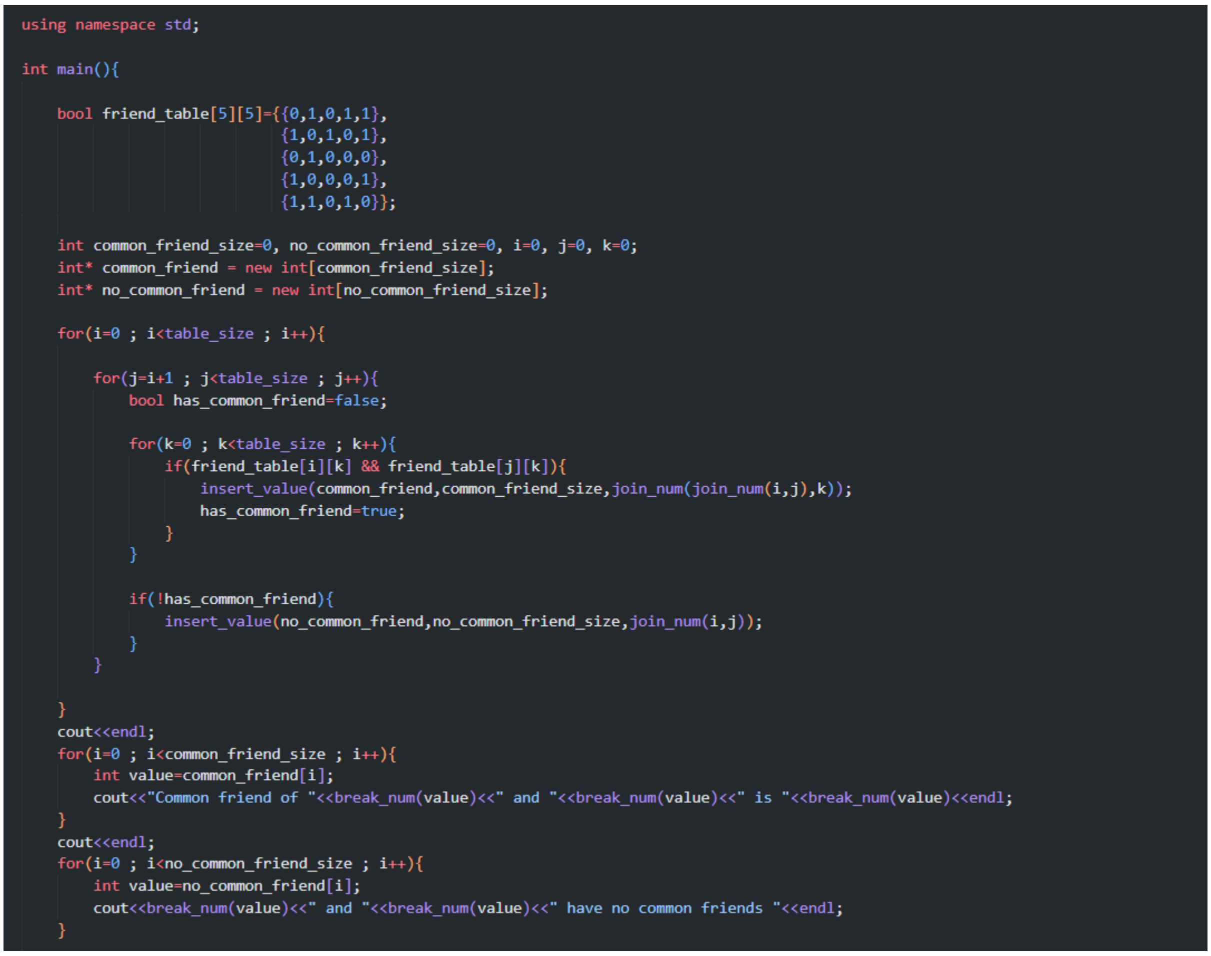


```
27 27 27
27 27 27
27 27 27
```

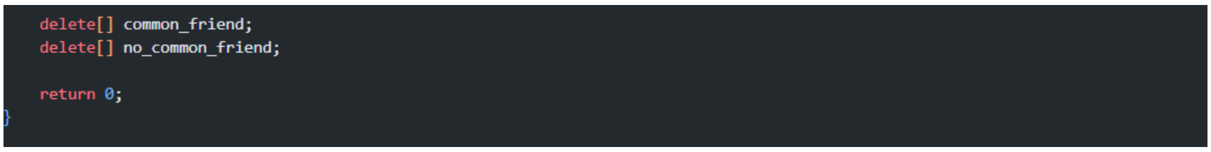
TASK 03:



```cpp
#include <iostream>

#define table_size 5


void insert_value(int* &array,int &size, int value){
    int* temp = new int[size+1];
    for(int i=0 ; i<size ; i++){
        temp[i]=array[i];
    }
    temp[size]=value;
    delete[] array;
    array=temp;
    size++;
}

int join_num(int val_01,int val_02){
    return (val_01*10)+val_02;
}
int break_num(int &val){
    int last_num=val%10;
    val/=10;
    return last_num;
}
```

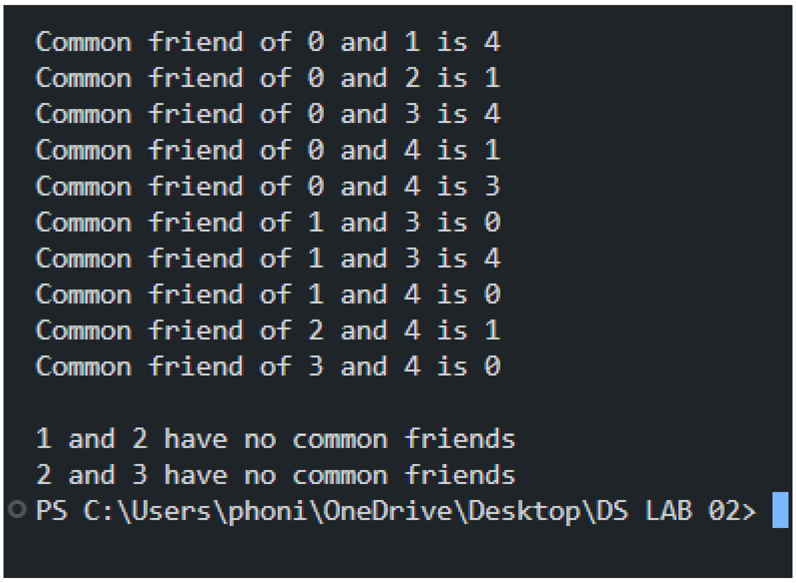


```cpp
using namespace std;

int main(){

    bool friend_table[5][5]={{0,1,0,1,1},
                             {1,0,1,0,1},
                             {0,1,0,0,0},
                             {1,0,0,0,1},
                             {1,1,0,1,0}};

    int common_friend_size=0, no_common_friend_size=0, i=0, j=0, k=0;
    int* common_friend = new int[common_friend_size];
    int* no_common_friend = new int[no_common_friend_size];

    for(i=0 ; i<table_size ; i++){

        for(j=i+1 ; j<table_size ; j++){
            bool has_common_friend=false;

            for(k=0 ; k<table_size ; k++){
                if(friend_table[i][k] && friend_table[j][k]){
                    insert_value(common_friend,common_friend_size,join_num(join_num(i,j),k));
                    has_common_friend=true;
                }
            }

            if(!has_common_friend){
                insert_value(no_common_friend,no_common_friend_size,join_num(i,j));
            }
        }

    }
    cout<<endl;
    for(i=0 ; i<common_friend_size ; i++){
        int value=common_friend[i];
        cout<<"Common friend of "<<break_num(value)<<" and "<<break_num(value)<<" is "<<break_num(value)<<endl;
    }
    cout<<endl;
    for(i=0 ; i<no_common_friend_size ; i++){
        int value=no_common_friend[i];
        cout<<break_num(value)<<" and "<<break_num(value)<<" have no common friends "<<endl;
    }
```



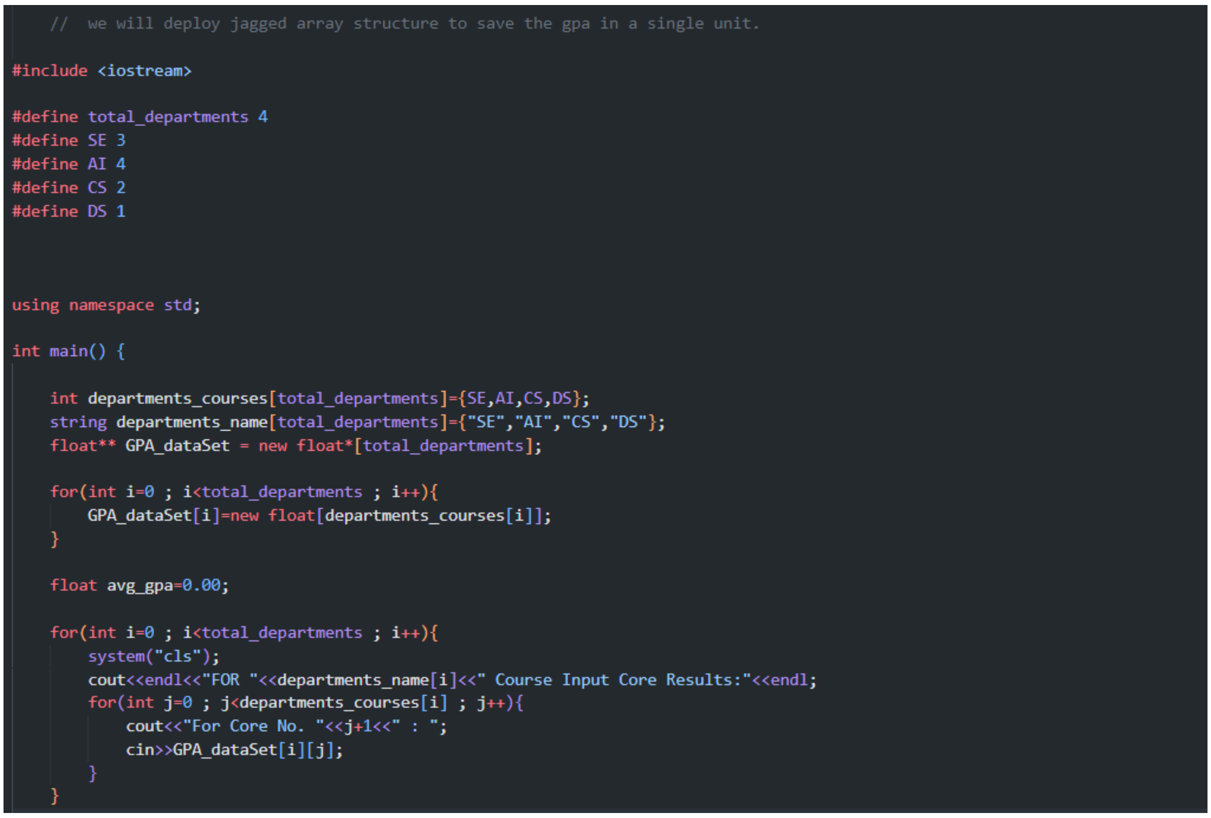
```cpp
    delete[] common_friend;
    delete[] no_common_friend;

    return 0;
}
```

OUTPUT:

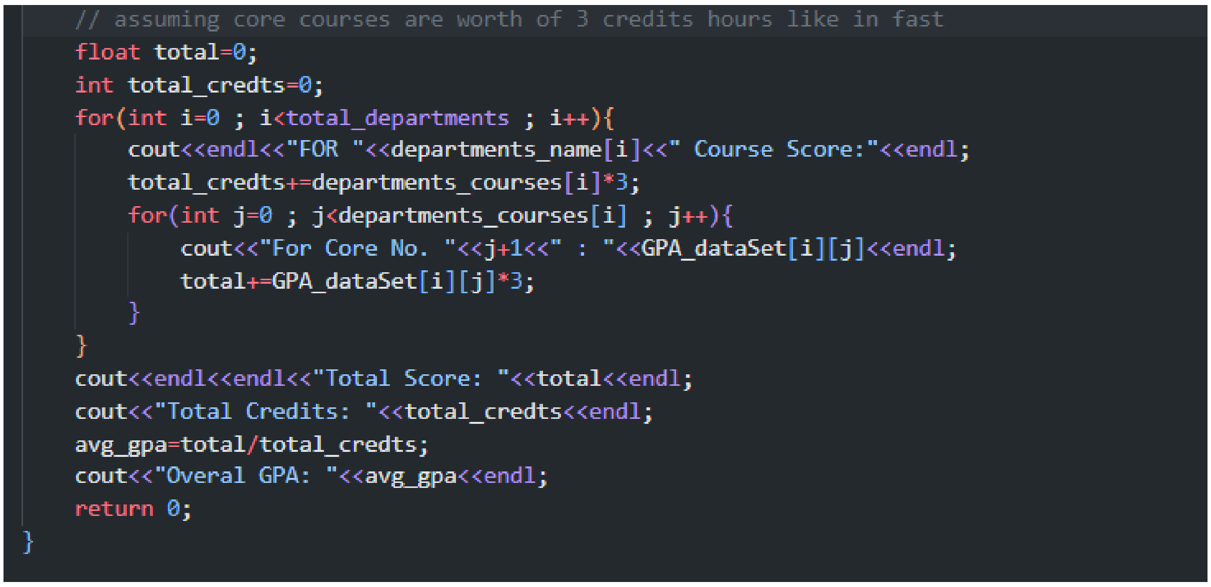


```
Common friend of 0 and 1 is 4
Common friend of 0 and 2 is 1
Common friend of 0 and 3 is 4
Common friend of 0 and 4 is 1
Common friend of 0 and 4 is 3
Common friend of 1 and 3 is 0
Common friend of 1 and 3 is 4
Common friend of 1 and 4 is 0
Common friend of 2 and 4 is 1
Common friend of 3 and 4 is 0

1 and 2 have no common friends
2 and 3 have no common friends
PS C:\Users\phoni\OneDrive\Desktop\DS LAB 02>
```

TASK 04:

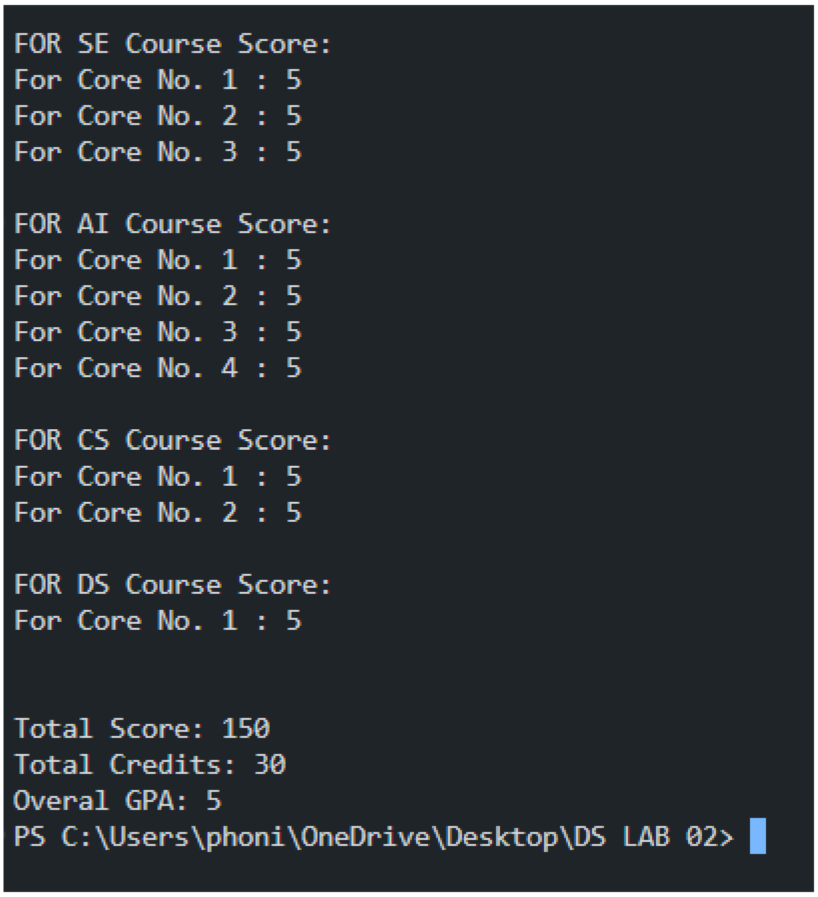


```cpp
    //  we will deploy jagged array structure to save the gpa in a single unit.

#include <iostream>

#define total_departments 4
#define SE 3
#define AI 4
#define CS 2
#define DS 1



using namespace std;

int main() {

    int departments_courses[total_departments]={SE,AI,CS,DS};
    string departments_name[total_departments]={"SE","AI","CS","DS"};
    float** GPA_dataSet = new float*[total_departments];

    for(int i=0 ; i<total_departments ; i++){
        GPA_dataSet[i]=new float[departments_courses[i]];
    }

    float avg_gpa=0.00;

    for(int i=0 ; i<total_departments ; i++){
        system("cls");
        cout<<endl<<"FOR "<<departments_name[i]<<" Course Input Core Results:"<<endl;
        for(int j=0 ; j<departments_courses[i] ; j++){
            cout<<"For Core No. "<<j+1<<" : ";
            cin>>GPA_dataSet[i][j];
        }
    }
```



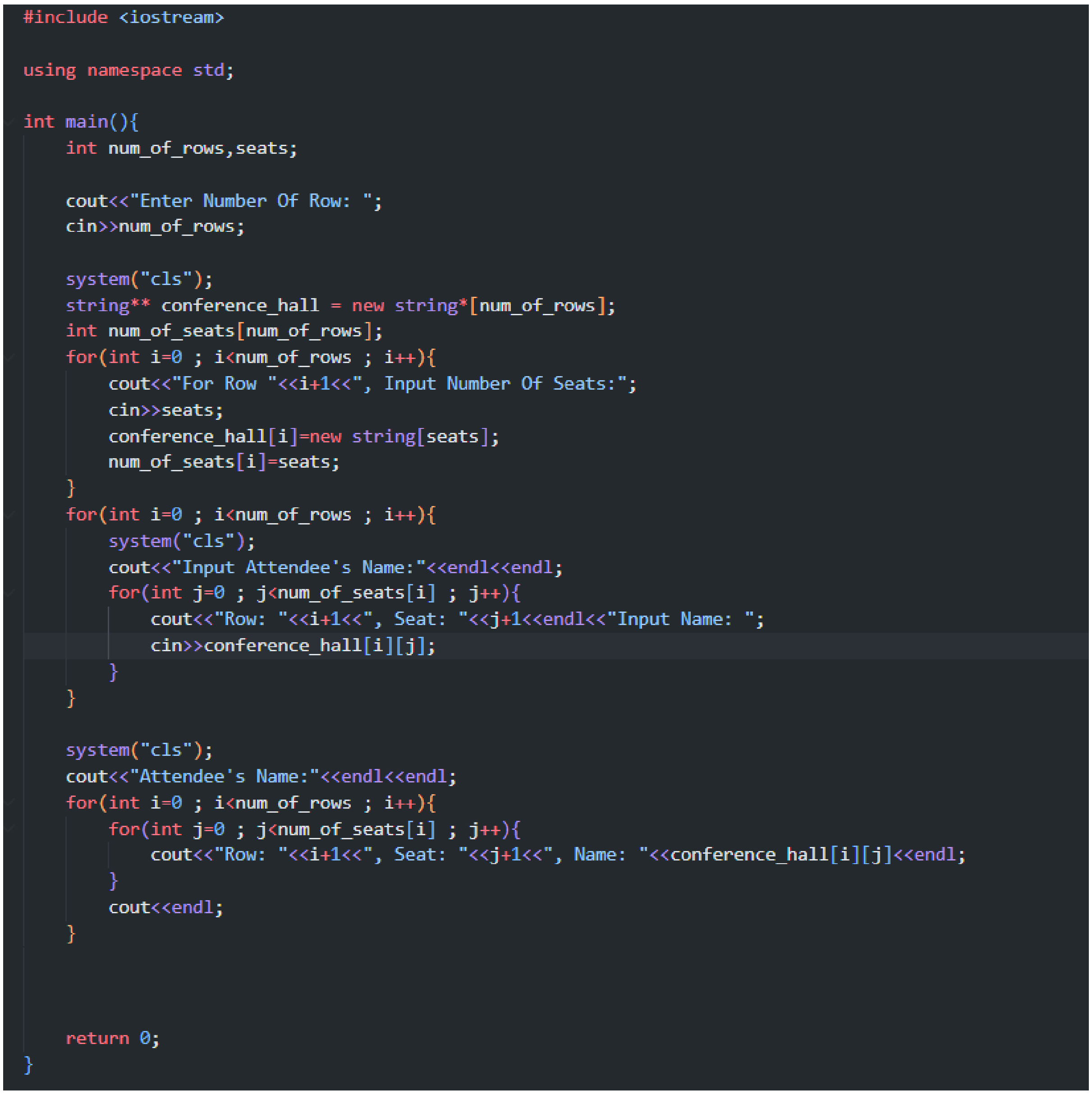
```
    // assuming core courses are worth of 3 credits hours like in fast
    float total=0;
    int total_credts=0;
    for(int i=0 ; i<total_departments ; i++){
        cout<<endl<<"FOR "<<departments_name[i]<<" Course Score:"<<endl;
        total_credts+=departments_courses[i]*3;
        for(int j=0 ; j<departments_courses[i] ; j++){
            cout<<"For Core No. "<<j+1<<" : "<<GPA_dataSet[i][j]<<endl;
            total+=GPA_dataSet[i][j]*3;
        }
    }
    cout<<endl<<endl<<"Total Score: "<<total<<endl;
    cout<<"Total Credits: "<<total_credts<<endl;
    avg_gpa=total/total_credts;
    cout<<"Overal GPA: "<<avg_gpa<<endl;
    return 0;
}
```

OUTPUT:



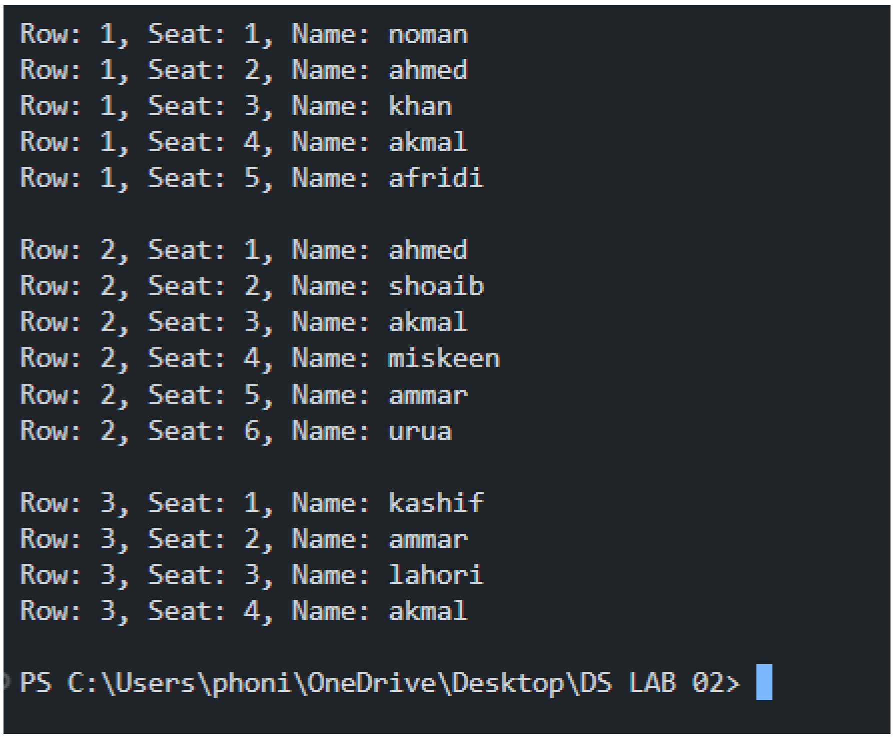
```
FOR SE Course Score:
For Core No. 1 : 5
For Core No. 2 : 5
For Core No. 3 : 5

FOR AI Course Score:
For Core No. 1 : 5
For Core No. 2 : 5
For Core No. 3 : 5
For Core No. 4 : 5

FOR CS Course Score:
For Core No. 1 : 5
For Core No. 2 : 5

FOR DS Course Score:
For Core No. 1 : 5


Total Score: 150
Total Credits: 30
Overal GPA: 5
PS C:\Users\phoni\OneDrive\Desktop\DS LAB 02>
```

TASK 05:



```cpp
#include <iostream>

using namespace std;

int main(){
    int num_of_rows,seats;

    cout<<"Enter Number Of Row: ";
    cin>>num_of_rows;

    system("cls");
    string** conference_hall = new string*[num_of_rows];
    int num_of_seats[num_of_rows];
    for(int i=0 ; i<num_of_rows ; i++){
        cout<<"For Row "<<i+1<<", Input Number Of Seats:";
        cin>>seats;
        conference_hall[i]=new string[seats];
        num_of_seats[i]=seats;
    }
    for(int i=0 ; i<num_of_rows ; i++){
        system("cls");
        cout<<"Input Attendee's Name:"<<endl<<endl;
        for(int j=0 ; j<num_of_seats[i] ; j++){
            cout<<"Row: "<<i+1<<", Seat: "<<j+1<<endl<<"Input Name: ";
            cin>>conference_hall[i][j];
        }
    }

    system("cls");
    cout<<"Attendee's Name:"<<endl<<endl;
    for(int i=0 ; i<num_of_rows ; i++){
        for(int j=0 ; j<num_of_seats[i] ; j++){
            cout<<"Row: "<<i+1<<", Seat: "<<j+1<<", Name: "<<conference_hall[i][j]<<endl;
        }
        cout<<endl;
    }



    return 0;
}
```

OUTPUT:



```
Row: 1, Seat: 1, Name: noman
Row: 1, Seat: 2, Name: ahmed
Row: 1, Seat: 3, Name: khan
Row: 1, Seat: 4, Name: akmal
Row: 1, Seat: 5, Name: afridi

Row: 2, Seat: 1, Name: ahmed
Row: 2, Seat: 2, Name: shoaib
Row: 2, Seat: 3, Name: akmal
Row: 2, Seat: 4, Name: miskeen
Row: 2, Seat: 5, Name: ammar
Row: 2, Seat: 6, Name: urua

Row: 3, Seat: 1, Name: kashif
Row: 3, Seat: 2, Name: ammar
Row: 3, Seat: 3, Name: lahori
Row: 3, Seat: 4, Name: akmal

PS C:\Users\phoni\OneDrive\Desktop\DS LAB 02>
```