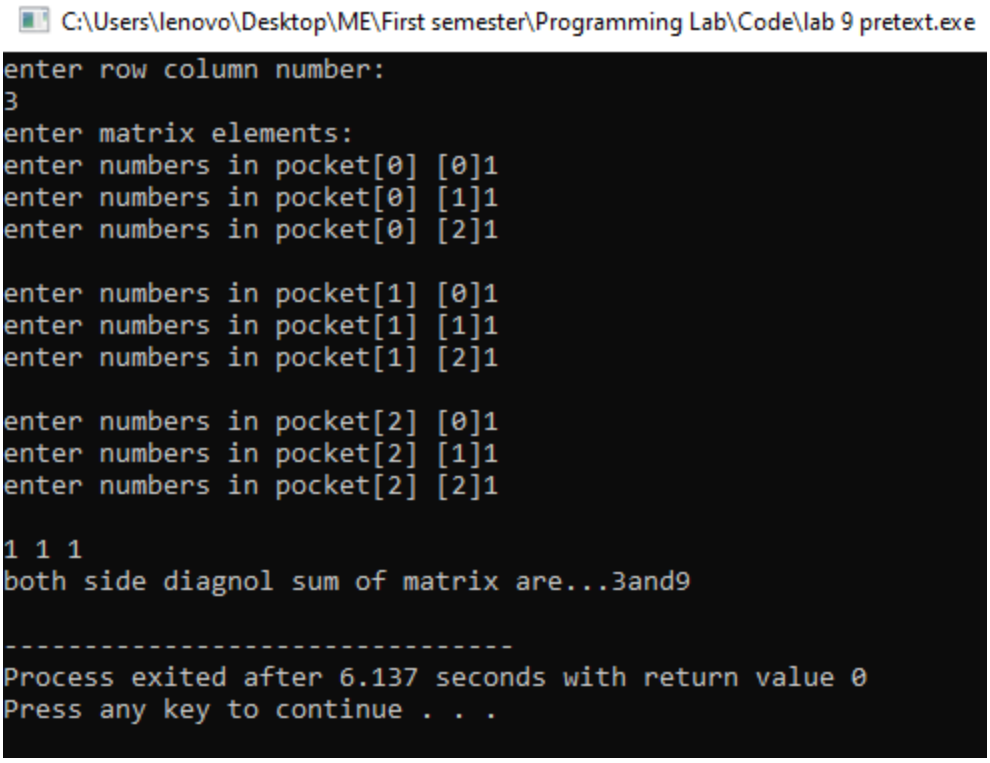
//task

// inverse of 3 by 3 matrix

```
#include <iostream>
using namespace std;

int main() {
   float matrix[3][3], inverseMatrix[3][3];
   int det = 0;

   cout << "Enter the elements of the 3x3 matrix:" << endl;
   for (int i = 0; i < 3; i++) {
      for (int j = 0; j < 3; j++) {
         cin >> matrix[i][j];
      }
   }

   // Calculate the determinant
   det = matrix[0][0] * (matrix[1][1] * matrix[2][2] - matrix[2][1] * matrix[1][2]) -
       matrix[0][1] * (matrix[1][0] * matrix[2][2] - matrix[1][2] * matrix[2][0]) +
       matrix[0][2] * (matrix[1][0] * matrix[2][1] - matrix[1][1] * matrix[2][0]);

   if (det == 0) {
      cout << "Inverse does not exist." << endl;
   } else {
      // Calculate the adjoint and the inverse matrix
```

```
    inverseMatrix[0][0] = (matrix[1][1] * matrix[2][2] - matrix[2][1] * matrix[1][2]) / det;
    inverseMatrix[0][1] = -(matrix[1][0] * matrix[2][2] - matrix[1][2] * matrix[2][0]) / det;
    inverseMatrix[0][2] = (matrix[1][0] * matrix[2][1] - matrix[1][1] * matrix[2][0]) / det;
    inverseMatrix[1][0] = -(matrix[0][1] * matrix[2][2] - matrix[0][2] * matrix[2][1]) / det;
    inverseMatrix[1][1] = (matrix[0][0] * matrix[2][2] - matrix[0][2] * matrix[2][0]) / det;
    inverseMatrix[1][2] = -(matrix[0][0] * matrix[2][1] - matrix[0][1] * matrix[2][0]) / det;
    inverseMatrix[2][0] = (matrix[0][1] * matrix[1][2] - matrix[0][2] * matrix[1][1]) / det;
    inverseMatrix[2][1] = -(matrix[0][0] * matrix[1][2] - matrix[0][2] * matrix[1][0]) / det;
    inverseMatrix[2][2] = (matrix[0][0] * matrix[1][1] - matrix[0][1] * matrix[1][0]) / det;

    cout << "\nInverse matrix:" << endl;
    for (int i = 0; i < 3; i++) {
      for (int j = 0; j < 3; j++) {
        cout << inverseMatrix[i][j] << "  ";
      }
      cout << endl;
    }
  }

  return 0;
}
```


C:\Users\lenovo\Desktop\ME\First semester\Programming Lab\Code\lab 9 pretext.exe
enter row column number:
3
enter matrix elements:
enter numbers in pocket[0] [0]1
enter numbers in pocket[0] [1]1
enter numbers in pocket[0] [2]1
enter numbers in pocket[1] [0]1
enter numbers in pocket[1] [1]1
enter numbers in pocket[1] [2]1
enter numbers in pocket[2] [0]1
enter numbers in pocket[2] [1]1
enter numbers in pocket[2] [2]1
1 1 1
both side diagnol sum of matrix are...3and9

Process exited after 6.137 seconds with return value 0
Press any key to continue . . .