Лабораторная №9. Матрицы в динамической памяти.

Темы, рассматриваемые в лабораторной работе:

Особенности размещения матриц в динамической памяти. Алгоритмы обработки матриц. Особенности передачи матриц-параметров.

Матрица в динамической памяти размещается как массив указателей на строки. Каждая строка - динамический массив:

```
double **matr=nullptr;
int   rows=3, columns=5;
matr = new  double* [rows];
for (int i=0; i<rows; ++i) matr[i] = new double[columns];
```

Освобождение занимаемой матрицей динамической памяти производится в обратном порядке:

```
for (int i=0; i<rows; ++i) delete[] matr[i];

delete[]matr;
```

Важно следить за корректным освобождением динамической памяти.

В случае размещения матрицы в динамической памяти передавать ее как параметр в функцию можно используя указатель на указатель.

Например, функция вычисления суммы всех элементов матрицы

```
double sum (double **matr, int rows, int columns);
```

Если требуется удаление (или добавление) строк/столбцов, то в динамической памяти создается новая матрица нужного размера, заполняется нужными значениям. После этого исходная матрица удаляется и меняется указатель.

Например:

```
void change_matr (double ** &matr, int& rows, int& cols){
  int new_rows, new_cols;
//определение размера новой матрицы new_rows, new_cols
  double **temp_matr=nullptr;
  temp_matr = new  double* [new_rows];
  for (int i=0; i<new_rows; ++i) temp_matr[i] = new double[new_cols];
// заполнение значений новой матрицы
 . . .
// удаление исходной матрицы
   for (int i=0; i<rows; ++i) delete[] matr[i];
   delete[]matr;
// «подмена» матрицы
   matr = temp_matr;
}
```

Задания:

Решить следующие задачи с использованием динамических массивов. Ошибки обрабатывать с помощью исключений. В функции `main` продемонстрировать использование функций и выполнить контроль утечки памяти путем проверки снимков кучи до и после вызова функции.

1. Функция ввода размера матрицы, размещение ее в динамической памяти и ввод элементов матрицы

```
void read_new(double** &m, int &rows, int &cols);
```

2. Функция удаления матрицы, размещенной в динамической памяти

```
void delete_matrix(double **& m, int rows);
```

3. Функция вычисления отношения сумм элементов на главной и побочной диагоналях квадратной матрицы.

4. Функция, добавляющая в матрицу столбец, содержащий минимальный положительный элемент в строке. Если в строке нет положительных элементов, столбец должен содержать значение 0.

5. Функция для добавления в матрицу строки (вставляемая строка и позиция вставки K передаются как параметры функции вместе с самой матрицей и ее исходными размерами).

6. Функция удаления строки матрицы с заданным номером.

7. Функция, удаляющая из матрицы строку и столбец, на пересечении которых стоит максимальный элемент матрицы. Считать, что функция выполняется для первого из максимальных элементов, если их несколько.

8.Удалить из матрицы строки, удовлетворяющие условию, заданному функцией-параметром. Использовать в качестве параметра функции: строка состоит из одних нулей, в строке нет положительных элементов.