1. **Цель работы**

Целью работы является изучение структуры данных двумерный массив.

1. **Задание на лабораторную работу**

Дана целочисленная квадратная матрица. Определить: 1. сумму элементов в тех столбцах, которые не содержат отрицательных элементов; 2. минимум среди сумм модулей элементов диагоналей, параллельных побочной диагонали матрицы

1. **Описание созданных функций**

**Имя**: sumWithoutMinus

**Назначение**: Рассчитывает сумму элементов в столбцах массива arr, которые не содержат отрицательных элементов.

**Входные данные**:

* arr: указатель на двумерный массив целых чисел
* rows: количество строк в массиве arr
* line: количество столбцов в массиве arr

**Выходные данные**:

* result: сумма элементов в столбцах без отрицательных элементов

**Побочные эффекты**: Отсутствуют

**Тестовые данные**: Отсутствуют

**Прототип**: int sumWithoutMinus(int \*\*arr,int row,int line);

**Псевдокод:**

1. Инициализирует переменную result со значением 0.
2. Выполняет цикл по переменной i от 0 до line - 1.
3. В каждой итерации проверяет, есть ли отрицательные элементы в столбце i массива arr:
   * Инициализирует переменную hasNegative со значением false.
   * Выполняет вложенный цикл по переменной j от 0 до rows - 1.
   * Если элемент arr[j][i] меньше 0, устанавливает значение hasNegative в true и выходит из вложенного цикла.
4. Если в столбце i нет отрицательных элементов (hasNegative равно false), выполняет еще один цикл по переменной j от 0 до rows - 1.
5. В каждой итерации прибавляет значение элемента arr[j][i] к переменной result.
6. Возвращается значение переменной result.

**Блок-схема:**

**Имя**: createArray

**Назначение**: Создает двумерный массив и заполняет его значениями, вводимыми пользователем.

**Входные данные**:

* rows: количество строк в массиве
* line: количество столбцов в массиве

**Выходные данные**:

* arr: указатель на созданный двумерный массив целых чисел

**Побочные эффекты**: Выделение памяти под массив, заполняет массив из ввода.

**Тестовые данные**: Вводные данные : rows = 3; lines = 3;

Выходные данные: Указатель на заполненный двумерный массив int[3][3];

**Прототип**: int\*\* createArray(int rows, int line);

**Псевдокод:**

1. Выделяет память для двумерного массива arr размером line.
2. Выполняет цикл по переменной i от 0 до rows - 1.
3. В каждой итерации выделяет память для каждой строки массива arr[i] размером line.
4. Выводит сообщение "Enter numbers in a line".
5. Выполняет вложенный цикл по переменной j от 0 до line - 1.
6. В каждой итерации считывает число из ввода и сохраняет его в переменную nums.
7. Присваивает значение nums элементу arr[i][j].
8. Выводит сообщение "Next row" для перехода к следующей строке.
9. Возвращается массив arr.

**Блок-схема:**

**Имя**: deleteArray

**Назначение**: Удаляет двумерный массив, освобождая выделенную под него память.

**Входные данные**:

* arr: указатель на двумерный массив целых чисел
* rows: количество строк в массиве

**Выходные данные**: Отсутствуют

**Побочные эффекты**: Освобождение памяти

**Тестовые данные**: Отсутствуют

**Прототип**: void deletArray(int\*\*arr, int rows);

**Псевдокод:**

1. Выполняет цикл по переменной i от 0 до rows - 1.
2. В каждой итерации освобождает память занятую для каждого элемента массива arr[i].
3. Освобождает память занятую для самого массива arr.
4. Завершает выполнение функции.

**Блок-схема:**

**Имя**: print

**Назначение**: Выводит содержимое двумерного массива на экран.

**Входные данные**:

* arr: указатель на двумерный массив целых чисел
* rowsSize: количество строк в массиве
* lineSize: количество столбцов в массиве

**Выходные данные**: Отсутствуют

**Побочные эффекты**: Вывод на экран

**Тестовые данные**: Отсутствуют

**Прототип**: void print(int\*\* arr, int rowsSize, int lineSize);

**Псевдокод:**

1. Выполняет цикл по переменной i от 0 до rowsSize - 1.
2. В каждой итерации выводит новую строку.
3. Выполняет вложенный цикл по переменной j от 0 до lineSize - 1.
4. В каждой итерации выводит значение элемента arr[i][j], за которым следует символ табуляции.
5. Завершает выполнение функции.

**Блок-схема:**

**Имя**: sumElementsDiagonal

**Назначение**: Рассчитывает сумму элементов по диагоналям в двумерном массиве и возвращает минимальную из этих сумм.

**Входные данные**:

* arr: указатель на двумерный массив целых чисел
* rowsSize: количество строк в массиве
* lineSize: количество столбцов в массиве

**Выходные данные**:

* result: минимальная сумма элементов по диагоналям

**Побочные эффекты**: Отсутствуют

**Тестовые данные**: Отсутствуют

**Прототип**: int sumElementsDiagonal(int\*\* arr, int rows, int line);

**Псевдокод:**

1. Инициализирует переменные result, minSumLeft и minSumRight со значениями 0 и INT\_MAX.
2. Проверяет, если rowsSize или lineSize меньше или равно 1, выводит сообщение об ошибке и возвращает 0.
3. Первый цикл осуществляет итерации по переменным i и y для вычисления суммы элементов по левой диагонали.
4. Во внутренних циклах происходит накопление абсолютного значения элементов массива arr и сохранение суммы в переменной result.
5. Если result меньше minSumLeft, обновляется значение minSumLeft и сбрасывается значение result.
6. Второй цикл осуществляет итерации по переменным i и y для вычисления суммы элементов по правой диагонали.
7. Во внутренних циклах также происходит накопление абсолютного значения элементов массива arr и сохранение суммы в переменной result.
8. Если result меньше minSumRight, обновляется значение minSumRight и сбрасывается значение result.
9. Затем происходит сравнение minSumRight и minSumLeft, и возвращается меньшее значение из них.
10. Завершается выполнение функции.

**Блок-схема:**

1. **Текст программы**
2. #define \_CRTDBG\_MAP\_ALLOC
3. #include <stdlib.h>
4. #include <crtdbg.h>
5. #ifdef \_DEBUG
6. #ifndef DBG\_NEW
7. #define DBG\_NEW new ( \_NORMAL\_BLOCK , \_\_FILE\_\_ , \_\_LINE\_\_ )
8. #define newDBG\_NEW
9. #endif
10. #endif
11. #include <iostream>
12. int sumWithoutMinus(int \*\*arr,int row,int line);
13. int\*\* createArray(int rows, int line);
14. void deletArray(int \*\*arr, int rows);
15. int sumElementsDiagonal(int\*\* arr, int rows, int line);
16. void print(int\*\* arr, int rowsSize, int lineSize);
17. int main()
18. {
19. int rows, line;
20. std::cout << "Enter the dimension of the matrix NxN" << std::endl;
21. std::cout << "Enter N:";
22. std::cin >> rows;
23. std::cout << std::endl;
24. line = rows;
25. int\*\* newArr = createArray(rows, line);
26. print(newArr, rows, line);
27. std::cout << "\n";
28. std::cout << "Sum whiout minus elements: " << sumWithoutMinus(newArr, rows, line) << std::endl;
29. std::cout << "Sum Elements Diagonals: " << sumElementsDiagonal(newArr, rows, line);
30. deletArray(newArr,rows);
31. newArr = nullptr;
32. \_CrtSetReportMode(\_CRT\_WARN, \_CRTDBG\_MODE\_FILE);
33. \_CrtSetReportFile(\_CRT\_WARN, \_CRTDBG\_FILE\_STDOUT);
34. \_CrtSetReportMode(\_CRT\_ERROR, \_CRTDBG\_MODE\_FILE);
35. \_CrtSetReportFile(\_CRT\_ERROR, \_CRTDBG\_FILE\_STDOUT);
36. \_CrtSetReportMode(\_CRT\_ASSERT, \_CRTDBG\_MODE\_FILE);
37. \_CrtSetReportFile(\_CRT\_ASSERT, \_CRTDBG\_FILE\_STDOUT);
38. \_CrtDumpMemoryLeaks();
39. return 0;
40. }
41. int sumWithoutMinus(int\*\* arr, int rows, int line) {
42. int result = 0;
43. for (int i = 0; i < line; i++) {
44. bool hasNegative = false;
45. for (int j = 0; j < rows; j++) {
46. if (arr[j][i] < 0) {
47. hasNegative = true;
48. break;
49. }
50. }
51. if (!hasNegative) {
52. for (int j = 0; j < rows; j++) {
53. result += arr[j][i];
54. }
55. }
56. }
57. return result;
58. }
59. int\*\* createArray(int rows, int line){
60. int nums;
61. int\*\* arr = new int\* [line];
62. for (int i = 0; i < rows; i++)
63. {
64. arr[i] = new int[line];
65. std::cout << "Enter numbers in a line" << std::endl;
66. for (int j = 0; j < line; j++)
67. {
68. std::cin >> nums;
69. arr[i][j] = nums;
70. }
71. std::cout << "Next row" << std::endl;
72. }
73. return arr;
74. }
75. void deletArray(int\*\* arr, int rows){
76. for (int i = 0; i < rows; i++)
77. {
78. delete[] arr[i];
79. }
80. delete[] arr;
81. }
82. int sumElementsDiagonal(int\*\* arr, int rowsSize, int lineSize) {
84. int result = 0;
86. int minSumLeft = INT\_MAX;
87. int minSumRight = INT\_MAX;
88. if (rowsSize <= 1 || lineSize <= 1 ) {
89. std::cout << "Error. return: 0";
90. return 0;
91. }
93. for (int i = 0, y = lineSize - 2; y >= 0;y--) {
94. for (int k = i,j = y; k < rowsSize-1 && j>=0;) {
95. for (; j >= 0;j--,k++)
96. {
97. result += abs(arr[k][j]);
98. }
99. }
100. if (result < minSumLeft) {
101. minSumLeft = result;
102. result = 0;
103. }
104. }
106. result = 0;
107. for (int i = 1, y = lineSize-1; i < rowsSize; i++) {
108. for (int k = i; k < rowsSize;) {
110. for (int j = y; k < rowsSize; k++,j--)
111. {
112. result +=abs(arr[k][j]);
113. }
114. }
115. if (result < minSumRight) {
116. minSumRight = result;
117. result = 0;
118. }
119. }
121. if (minSumRight < minSumLeft) {
122. return minSumRight;
123. }
125. return minSumLeft;
126. }
127. void print(int\*\* arr,int rowsSize,int lineSize) {
128. for (int i = 0; i < rowsSize; i++) {
130. std::cout << '\n';
131. for (int j = 0; j < lineSize; j++)
132. {
133. std::cout << arr[i][j] << '\t';
134. }
135. }
136. }
137. **Пример выполнения программы**
138. **Анализ результатов и выводы**

В результате выполнения лабораторной работы были изучены способы написания сложных математических функций и использования указателей на двумерные массивы. Различные функции были реализованы для создания массива, заполнения его элементами, вывода содержимого на экран, а также для вычисления минимальной суммы элементов по диагоналям.

Достоинства кода:

* Код хорошо читаемый и отформатированный, что облегчает понимание его структуры и логики.
* Каждое задание реализовано в виде отдельной функции, что делает код модульным и удобным для переиспользования в других проектах.

Недостатки кода:

* Отсутствие проверок на корректность входных данных может привести к ошибкам выполнения программы в случае некорректного ввода данных пользователем. Рекомендуется добавить проверки на валидность входных параметров, чтобы предотвратить нежелательное поведение программы.
* Нет комментариев, поясняющих логику и функциональность кода. Добавление комментариев поможет другим разработчикам быстрее понять назначение каждой функции и ее роль в программе.