Условие

Определить индексы минимального элемента (в предположении, что он единственный) в каждой из трех матриц А(7x5), B(3x7), C(4x5).

Классы программы

|  |  |
| --- | --- |
| **Название** | **Наследуется от** |
| MatrMin | - |
| newMatrMin | Matrix |

Поля класса Matrix

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип** | **Назначение** |
| matr | Указатель на вещественный массив | Элементы матрицы |
| a | Целочисленный | Число строк матрицы |
| b | Целочисленный | Число столбцов матрицы |
| min | С плавающей точкой | Минимальный элемент массива |
| c | Символьный | Имя матрицы |
| row | Целочисленный | Строковый индекс элемента матрицы |
| column | Целочисленный | Столбцовый индекс элемента матрицы |

Методы класса Matrix

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип возвращаемого значения** | **Назначение** |
| input | - | Ввод элементов матрицы |
| output | - | Вывод минимального элемента указанного массива с его индексами |

Методы класса newMatrix

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип возвращаемого значение** | **Назначение** |
| Minelem | - | Вычисление минимального элемента и его индексов у конкретной матрицы |

Консольная программа с использованием класса

Объявление класса начинается с зарезервированного ключевого слова class, после которого написано имя этого класса – MatrMin. Далее объявлен спецификатор доступа private, дающий доступ к полям класса только методам текущего класса. Для методов класса объявлен спецификатор доступа public, то есть методы можно использовать в любой точке программы, где доступно описание класса. Конструктор MatrMin(имя конструктора такое же, как и имя класса) - специальный метод класса, предназначенный для инициализации элементов класса и распределения памяти под поля класса. Деструктор ~MatrMin - специальный метод класса, который служит для уничтожения класса. Методы (функции) класса, нужные для данной задачи: input – функция ввода матрицы, Minelem – функция для определения минимального элемента и его индексов, output – функция, выводящая значение минимального элемента массива и его индексы.

#include<iostream>

#include <conio.h>

#include<iomanip>

#include<string>

#include <time.h>

using namespace std;

class MatrMin

{

private:

float \*\*matr;

int a, b;

double min;

char c;

int row, column;

public:

MatrMin(int a, int b, char c);

~MatrMin();

void input();

void output();

void Minelem();

};

MatrMin::MatrMin(int a, int b, char c)

{

this->a = a;

this->b = b;

this->c = c;

matr = new float\*[a];

for (int i = 0; i < a; i++)

{

matr[i] = new float[b];

}

}

void MatrMin::input()

{

int i, j;

srand(time(NULL));

setlocale(LC\_ALL, "Russian");

cout << "Матрица " << c << endl;

for (i = 0; i < a; i++)

{

for (j = 0; j < b; j++)

{

matr[i][j] = (rand() % 200) - 100;

cout <<setw(5)<< matr[i][j] << " ";

}

cout << endl;

}

}

void MatrMin::output()

{

setlocale(LC\_ALL, "Russian");

cout << "Минимальный элемент матрицы " << c << " =" << min << "\n Его индексы " << row << ", " << column<<endl;

}

MatrMin::~MatrMin()

{

for (int i = 0; i < a; i++)

{

delete[] matr[i];

}

delete[] matr;

}

void MatrMin::Minelem()

{

min = matr[0][0];

for (int i = 0; i < a; i++)

{

for (int j = 0; j < b; j++)

{

if (min >= matr[i][j])

{

min = matr[i][j];

row = i + 1;

column = j + 1;

}

}

}

}

int main()

{

MatrMin \*A, \*B, \*C; //A[7][5], B[3][7], C[4][5]

A = new MatrMin(7, 5, 'A');

B = new MatrMin(3,7, 'B');

C = new MatrMin (4, 5, 'C');

A->input();

B->input();

C->input();

cout << endl;

A->Minelem();

A->output();

cout << endl;

B->Minelem();

B->output();

cout << endl;

C->Minelem();

C->output();

\_getch();

}

Определение класса-наследника

В классе-предке MatrMin уровень доступа к полям класса изменен с private на protected, чтобы доступ к полям класса имели не только методы текущего класса, но и его классов-наследников. Методы класса MatrMin по-прежнему имеют уровень доступа public. Создан производный класс (наследник) класса MatrMin, его имя newMatrMin. В класс-наследник добавлен метод – Minelem.

#include<iostream>

#include <conio.h>

#include<iomanip>

#include<string>

#include <time.h>

using namespace std;

class MatrMin

{

protected:

char c;

int row, column;

double min;

public:

MatrMin(int a, int b, char c);

~MatrMin();

void input();

void output();

int a, b;

float \*\*matr;

};

MatrMin::MatrMin(int a, int b, char c)

{

this->a = a;

this->b = b;

this->c = c;

matr = new float\*[a];

for (int i = 0; i < a; i++)

{

matr[i] = new float[b];

}

}

void MatrMin::input()

{

int i, j;

srand(time(NULL));

setlocale(LC\_ALL, "Russian");

cout << "Матрица " << c << endl;

for (i = 0; i < a; i++)

{

for (j = 0; j < b; j++)

{

matr[i][j] = (rand() % 200) - 100;

cout <<setw(5)<<matr[i][j] << " ";

}

cout << endl;

}

}

void MatrMin::output()

{

setlocale(LC\_ALL, "Russian");

std::cout << "Минимальный элемент матрицы " << c << " =" << min << "\n Его индексы " << row << ", " << column << endl;

}

MatrMin::~MatrMin()

{

for (int i = 0; i < a; i++)

{

delete[] matr[i];

}

delete[] matr;

}

class newMatrMin : public MatrMin {

public:

void Minelem();

newMatrMin(int a, int b, char c);

~newMatrMin();

};

newMatrMin::newMatrMin(int a, int b, char c) : MatrMin(a, b, c) {}

newMatrMin::~newMatrMin() {}

void newMatrMin::Minelem()

{

min = matr[0][0];

for (int i = 0; i < a; i++)

{

for (int j = 0; j < b; j++)

{

bool srav = min >= matr[i][j];

if (srav)

{

min = matr[i][j];

row = i;

column = j;

}

}

}

}

int main()

{

newMatrMin \*A = new newMatrMin(7, 5, 'A'); //A[7][5], B[3][7], C[4][5]

newMatrMin \*B = new newMatrMin(3, 7, 'B');

newMatrMin \*C = new newMatrMin(4, 5, 'C');

A->input();

B->input();

C->input();

std::cout << endl;

A->Minelem();

A->output();

std::cout << endl;

B->Minelem();

B->output();

std::cout << endl;

C->Minelem();

C->output();

A->~newMatrMin();

B->~newMatrMin();

C->~newMatrMin();

\_getch();

}