|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Робототехника и комплексная автоматизация (РК)

КАФЕДРА Системы автоматизированного проектирования (РК-6)

**ОТЧЕТ О ВЫПОЛНЕНИИ ДОМАШНЕГО ЗАДАНИЯ**

**по курсу объектно-ориентированное программирование**

Студент Камалов Антон Павлович

Группа РК6-25Б

Тип задания Домашняя работа №1

Вариант 2М

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Камалов А.П.**

*подпись, дата фамилия, и.о.*

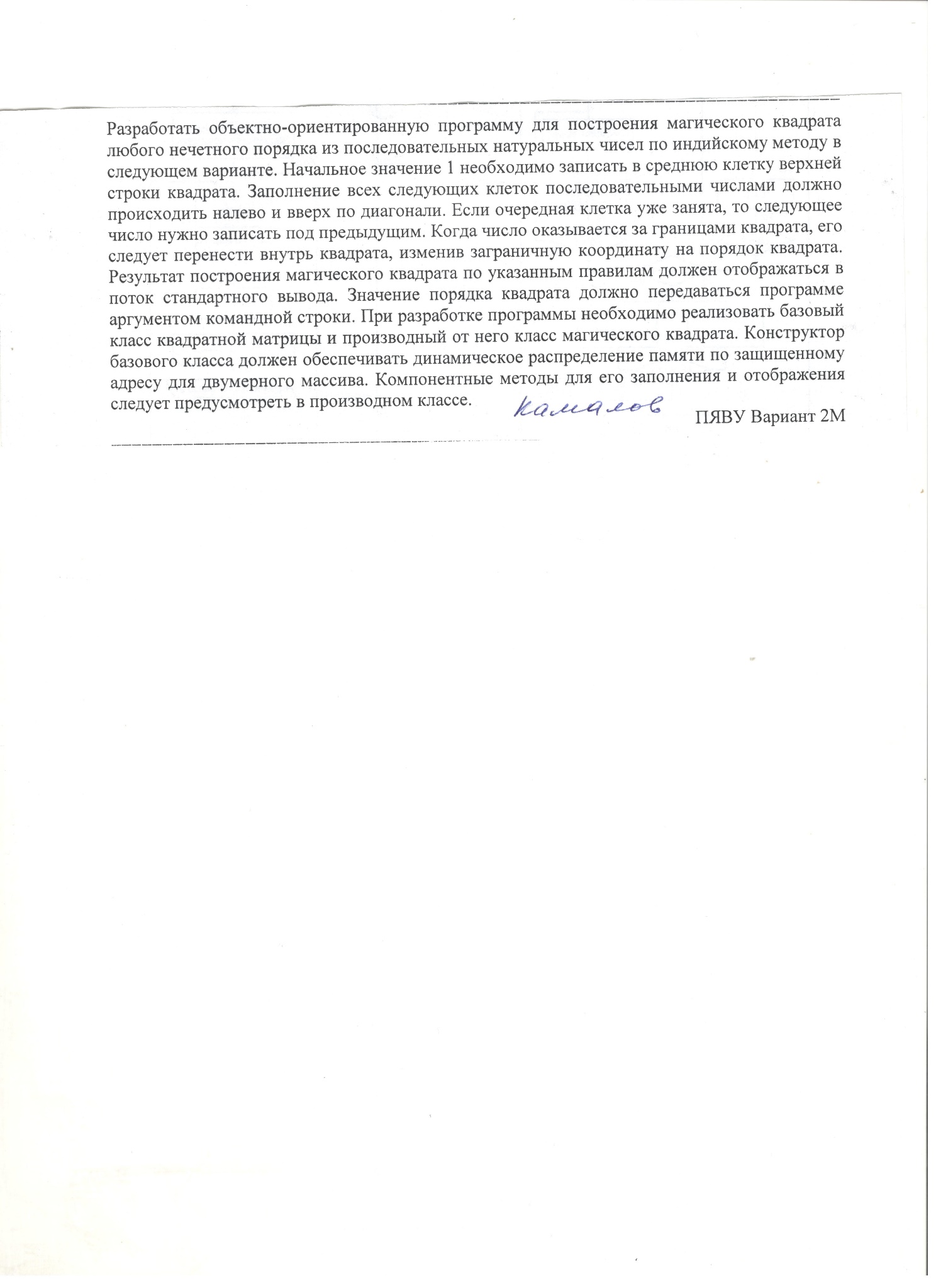
Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Волосатова Т.М.**

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Москва, 2023 г.*

**Задание**

****

**Описание программы**

**Входные данные:** размер магического квадрата (int n).

**Выходные данные:** магический квадрат n-го порядка.

**Класс Square**

*Информационные поля:*

\_degree (unsigned) – порядок магического квадрата;

\_table (unsigned \*\*) – двумерный массив для хранения чисел;

*Конструкторы и деструкторы:*

Square (unsigned) – конструктор с параметром;

~Square () – деструктор;

**Класс Magic**

*Конструкторы и деструкторы:*

Magic (unsigned) – конструктор инициализации;

*Методы:*

void indian () – публичный метод заполнения магического квадрата числами по индийскому методу;

void print () – публичный метод вывода содержимого магического квадрата;

int width () – защищённый метод выравнивания выведенного числа по ширине;

int reflect (int) – защищённый метод отражения числа в пределах магического квадрата;

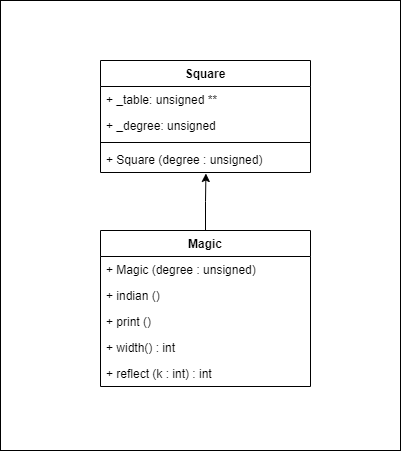


Рисунок 1. Диаграмма классов

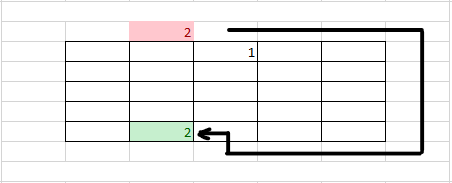


Рисунок 2. Заполнение магического квадрата 5х5, когда следующее число выходит за его пределы

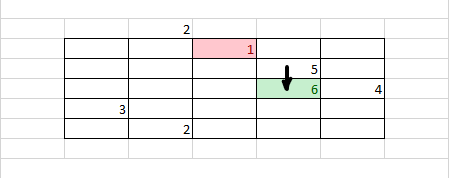


Рисунок 3. Заполнение магического квадрата 5х5, когда следующее число оказывается на месте существующего

**Алгоритм решения:**

1. Проверяем количество аргументов командной строки. Если их меньше 2, то программа завершает работу с кодом возврата -1;
2. Преобразуем первый аргумент в целое число n с помощью функции atoi ();
3. Проверяем, что n – нечётное число. В противном случае программа завершает работу с кодом возврата -2;
4. Создаём объект класса Magic – m, который генерирует магический квадрат размером n x n.
5. Заполняем магический квадрат с помощью функции indian (). Если следующее число выходит за пределы квадрата (см. рисунок 2), то вызывается функция reflect (), а если оказывается на месте существующего (см. рисунок 3) – присваиваем ячейке ниже его значение.
6. Вызываем функцию print(), выводящее содержимое магического квадрата на экран с выравниванием по ширине, рассчитанной с использованием функции width().

**Список литературы**

1. Пол Айра. Объектно-ориентированное программирование на C++: пер. с англ. 2-е изд. СПБ.: Невский Диалект; М.: Издательство БИНОМ, 2001. 462 с. [Pohl Ira. Object-Oriented Programming Using C++. 2nd ed. Addison-Wesley, 1996. 576 p.].
2. Страуструп Б. Язык программирования C++: пер. с англ / под ред. Н.Н. Мартынова. Специальное изд. М.: Бином, 2011. 1035 c. [Stroustrup B. The C++ Programming Language. Special ed. Addison-Wesley, 2000. 1029p.].
3. Шилдт Г. Самоучитель C++. 3-е изд. СПБ.: БХВ-Петербург, 2002. 688 с. [Schildt H. Teach Yourself C++. 3d ed. McGraw-Hill, 1998. 768 p.].
4. Волосатова Т.М., Родионов С.В. Объектно-ориентированное программирование на С++. Режим доступа: http://bigor.bmstu.ru/?cnt/?doc=VU/base.cou (дата обращения 12.04.2023).

**Приложение 1**

Текст программы

#include <iostream>

#include <cstdio>

#include <cstdlib>

class Square{

protected:

unsigned \_degree;

unsigned \*\*\_table;

public:

Square(unsigned degree){

\_degree = degree;

\_table = new unsigned\*[degree];

for (int i = 0; i < degree; i++) {

\_table[i] = new unsigned[degree];

for (int j = 0; j < degree; j++){

\_table[i][j] = 0;

}

}

}

~Square(){

for (int i = 0; i < \_degree; i++){

delete []\_table[i];

}

delete []\_table;

}

};

class Magic : public Square{

public:

Magic(unsigned);

void indian();

void print();

protected:

int width();

int reflect(int);

};

Magic::Magic(unsigned degree) : Square(degree){};

void Magic::print(){

int len = width();

for (int i = 0; i < \_degree; i++){

for (int j = 0; j < \_degree; j++){

printf("%0\*d ", len, \_table[i][j]);

}

putchar('\n');

}

putchar('\n');

}

int Magic::width(){

int max = \_degree \* \_degree;

int result = 0;

while (max > 0){

max/=10;

result++;

}

return result;

}

int Magic::reflect(int k){

if (k < 0) {return k + \_degree;}

return k;

}

void Magic::indian(){

int i, j;

int z = 1;

int row = 0, col = 0;

int max = \_degree \* \_degree;

col = \_degree/2;

\_table[row][col] = z;

while (z < max){

i = reflect(row - 1);

j = reflect(col - 1);

if (\_table[i][j] > 0){

i = reflect(row+1);

j = reflect(col);

}

row = i;

col = j;

z++;

\_table[row][col] = z;

}

}

int main(int argc, char \*\*argv){

if (argc < 2) return -1;

int n = atoi(argv[1]);

if ((n % 2) == 0) return -2;

Magic m(n);

m.indian();

m.print();

return 0;

}

**Приложение 2**

Результат работы программы

