|  |  |
| --- | --- |
|  | **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 ЛАБОРАТОРНОЙ РАБОТЫ**

**по курсу объектно-ориентированное программирование**

Студент Камалов Антон Павлович

Группа РК6-25Б

Тип задания Лабораторная работа №7

Вариант C05+

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

*подпись, дата фамилия, и.о.*

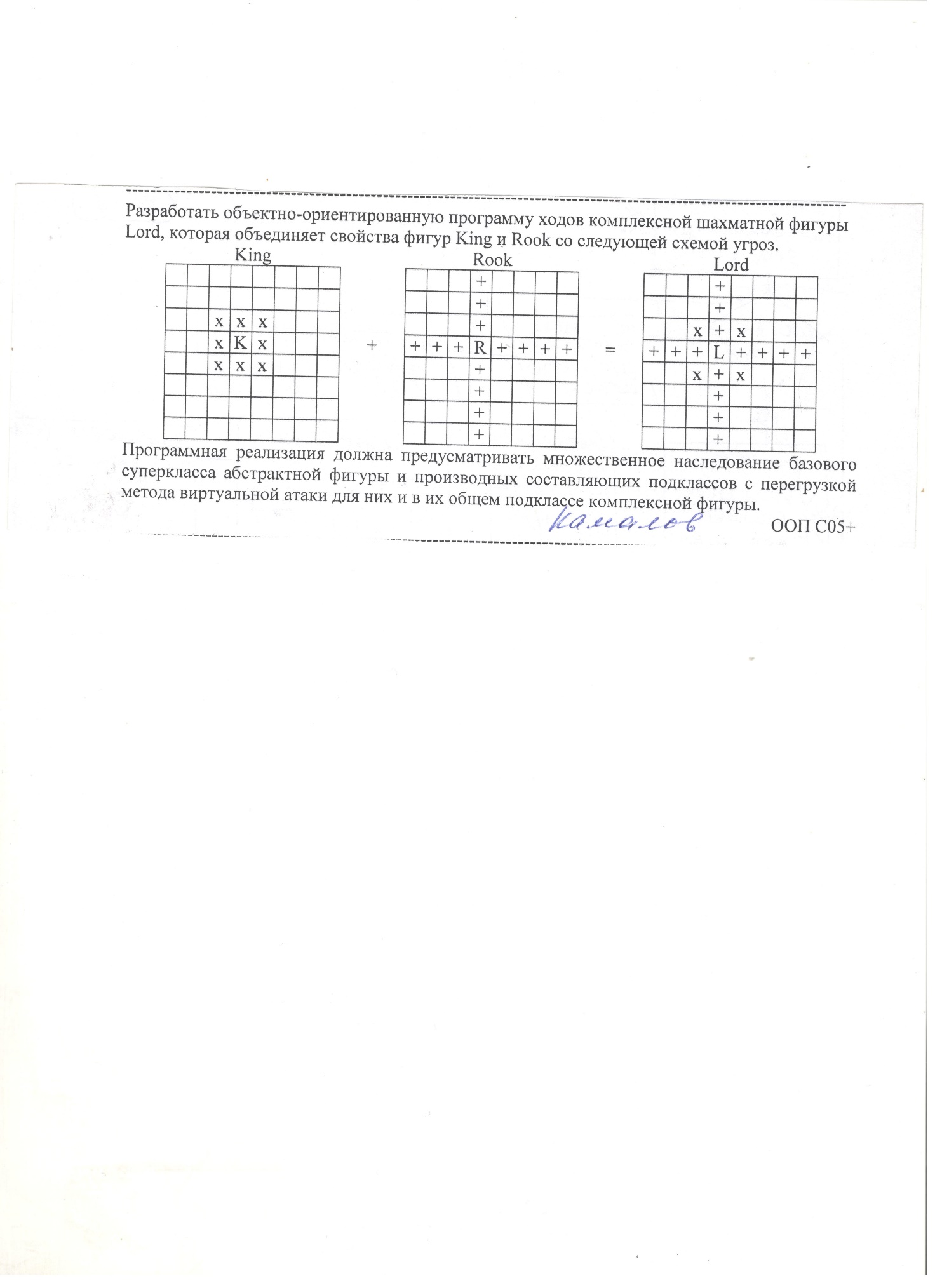
Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Волосатова Т. М.**

*подпись, дата фамилия, и.о.*

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

*Москва, 2023 г.*

**Задание**

****

**Описание программы**

**Входные данные:** символьная строка с координатами фигуры.

**Выходные данные:** символьное отображение шахматной доски с доступными для фигуры позициями для атаки.

**Класс Figure**

*Информационные поля:*

pos (char []) – координаты фигуры на шахматной доске;

*Конструкторы и деструкторы:*

Figure () – конструктор по умолчанию;

Figure (char \*) – конструктор инициализации;

*Методы:*

operator== (char \*) – публичный метод перегрузки оператора равенства;

attack (char \*) – публичный чистый виртуальный метод атаки фигуры;

isa () – публичный чистый виртуальный метод маркировки фигуры;

desk () – публичный метод отображения шахматной доски.

**Класс King**

*Конструкторы и деструкторы:*

King () – конструктор по умолчанию;

King (char \*) – конструктор инициализации;

*Методы:*

attack (char \*) – публичный метод атаки фигуры King;

isa () – публичный метод маркировки фигуры King.

**Класс Rook**

*Конструкторы и деструкторы:*

Rook () – конструктор по умолчанию;

Rook (char \*) – конструктор инициализации;

*Методы:*

attack (char \*) – публичный метод атаки фигуры Rook;

isa () – публичный метод маркировки фигуры Rook.

**Класс Lord**

*Конструкторы и деструкторы:*

Rook (char \*) – конструктор инициализации;

*Методы:*

attack (char \*) – публичный метод атаки фигуры Lord;

isa () – публичный метод маркировки фигуры Lord.

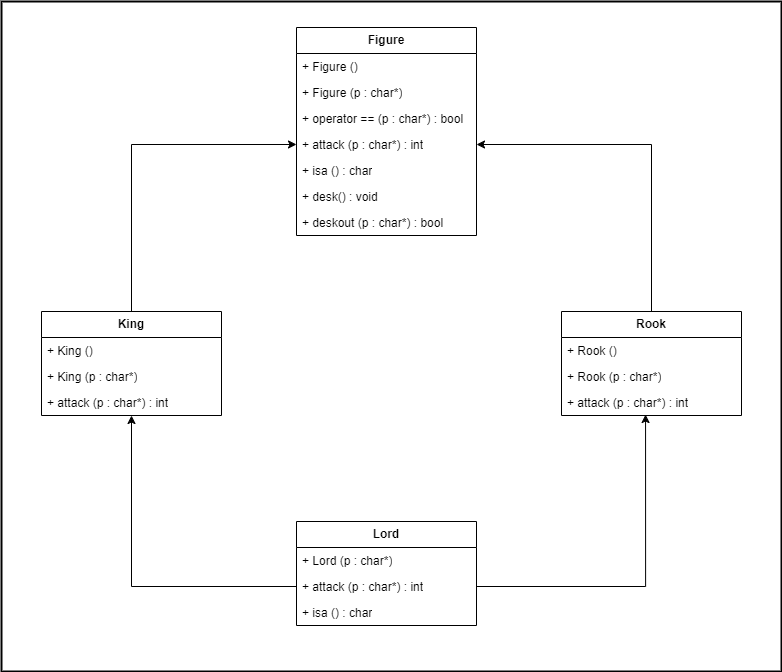


Рисунок 1. Диаграмма классов

**Алгоритм решения:**

1. Проверка на количество аргументов. Если их меньше 2, то программа завершает свою работу.
2. Создаётся объект класса **Lord** с позицией, переданной в аргументе командной строки.
3. Вызов функции desk() для отображения шахматной доски с расположением фигуры Lord.
   1. Проходя все клетки шахматной доски, проверяем, не совпадают ли координаты текущей с координатами самой фигуры Lord. Если да, то ставим на месте клетки символ ‘L’. Иначе производим проверку позиции на возможность в ней атаковать.
   2. Если текущая позиция является атакующей, то мы помечаем её соответствующим символом.
   3. Выводим всю шахматную доску на экран

**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 21.05.2023).

**Приложение 1**

Текст программы

#include <iostream>

using std::cout;

class Figure {

protected:

char pos[2];

public:

Figure() {}; // нужен для реализации мн. наследования

Figure(char \* p) {

pos[0] = p[0];

pos[1] = p[1];

};

bool operator==(char \* p) { return (pos[0] == p[0]) && (pos[1] == p[1]); };

virtual int attack(char \* p) = 0;

virtual char isa() = 0;

void desk();

static bool deskout(char \* p);

};

class King : virtual public Figure {

public:

King() {};

King(char \* p) : Figure(p) {};

int attack(char \* p);

};

class Rook : virtual public Figure {

public:

Rook() {};

Rook(char \* p) : Figure(p) {};

int attack(char \* p);

};

class Lord : public King, public Rook {

public:

Lord(char \* p) : Figure(p) {};

char isa() {return 'L';};

int attack(char\*);

};

bool Figure::deskout(char \* p) {

return ((p[0] > 'h') || (p[0] < 'a') || (p[1] < '1') || (p[1] > '8'));

}

void Figure::desk() {

int i, j;

char s[2];

const char \* mark = ".+x";

char m;

cout << " a b c d e f g h\n";

for (i = 8; i > 0; i--) {

cout << i << ' ';

s[1] = '0' + i;

for (j=0; j < 8; j++) {

s[0] = 'a' + j;

m = (!(\*this == s)) ? mark[attack(s)] : isa();

cout << m << " ";

}

cout << i << '\n';

}

cout << " a b c d e f g h\n";

return;

}

int King::attack(char \* p) {

if (deskout(p))

return 0;

int x = abs(p[0] - pos[0]);

int y = abs(p[1] - pos[1]);

if (x + y < 2 || (x == y && x + y == 2))

return 1;

return 0;

}

int Rook::attack(char \* p) {

if (deskout(p))

return 0;

int x = abs(p[0] - pos[0]);

int y = abs(p[1] - pos[1]);

if ((x + y > 0 && y == 0) || (x + y > 0 && x == 0))

return 1;

return 0;

}

int Lord::attack(char \* p) {

if (Rook::attack(p) > 0)

return 1;

if (King::attack(p) > 0)

return 2;

return 0;

}

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

if (argc < 2)

return -1;

Lord k(argv[1]);

k.desk();

}

**Приложение 2**

Результат работы программы

