МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего профессионального образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ

ОТЧЕТ

ЗАЩИЩЕН С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

Старший преподаватель Е.О.Шумова должность, уч. степень, звание подпись, дата инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ

№3

**«Наследование классов, базовый класс, производный класс»**

по дисциплине: Объектно-ориентированное программирование

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. Z7431 20.01.2020 М.Д.Семочкин

подпись, дата инициалы, фамилия

Студ. билет 2014/1054

Санкт-Петербург

2020

**Цель работы**

Изучить механизм создания нового класса на основе уже существующего, варианты доступа к элементам базового класса из производного.

Закрепить знания по теме: Классы, наследование классов, варианты доступа.

**Задание**

Вариант 4:

Создать класс Board для описания шахматной доски. В нём предусмотреть массив 8х8 элементов и метод для перевода цифр 1-8 в буквы A-H и обратно.

На основе класса Board создать класс Composition для составления шахматной композиции. В нём предусмотреть возможность добавления/удаления фигур на доску, распечатку композиции.

**Используемые инструменты разработки**

IDE – Microsoft Visual C++ 2010 Express

**Текст программы**

// Вариант 4

//

// Создать класс Board для описания шахматной доски.

// В нём предусмотреть массив 8х8 элементов и метод для перевода цифр 1-8 в буквы A-H и обратно.

// На основе класса Board создать класс Composition для составления шахматной композиции.

// В нём предусмотреть возможность добавления/удаления фигур на доску, распечатку композиции.

//

#include <iostream>

#include <stdlib.h>

using namespace std;

class Board {

protected:

int figures[8][8];

public:

int letterToNumber(char letter);

char numberToLetter(int number);

void initBoard();

};

void Board::initBoard() {

for (int i = 0; i < 8; i++) {

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

figures[i][j] = 0;

}

}

}

int Board::letterToNumber(char letter) {

switch (letter) {

case 'A': return 1;

case 'B': return 2;

case 'C': return 3;

case 'D': return 4;

case 'E': return 5;

case 'F': return 6;

case 'G': return 7;

case 'H': return 8;

}

}

char Board::numberToLetter(int number) {

switch (number) {

case 1: return 'A';

case 2: return 'B';

case 3: return 'C';

case 4: return 'D';

case 5: return 'E';

case 6: return 'F';

case 7: return 'G';

case 8: return 'H';

}

}

class Composition : public Board {

public:

void addFigure(int figureCode, int x, int y);

void removeFigure(int x, int y);

void printBoard();

void printInfo();

};

void Composition::printBoard() {

for (int i = 0; i < 8; i++) {

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

cout << figures[i][j] << " ";

}

cout << endl << endl;

}

cout << endl << endl;

}

void Composition::addFigure(int figureCode, int x, int y) {

figures[x][y] = figureCode;

}

void Composition::removeFigure(int x, int y) {

figures[x][y] = 0;

}

void Composition::printInfo() {

cout << "0 - пустая клетка" << endl;

cout << "1 - пешка" << endl;

cout << "2 - ладья" << endl;

cout << "3 - конь" << endl;

cout << "4 - слон" << endl;

cout << "5 - ферзь" << endl;

cout << "6 - король" << endl << endl;

}

void printMenu() {

cout << "Выберите команду:" << endl;

cout << "1 - вывести доску" << endl;

cout << "2 - очистить доску" << endl;

cout << "3 - добавить фигуру" << endl;

cout << "4 - удалить фигуру" << endl;

cout << "5 - выход" << endl << endl;

}

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

setlocale(LC\_ALL, "russian");

Composition myComposition;

myComposition.initBoard();

myComposition.printBoard();

myComposition.printInfo();

int x, y, figureCode, inputCommandCode;

int exit = 0;

while (exit == 0) {

printMenu();

inputCommandCode = -1;

while ((inputCommandCode < 1) || (inputCommandCode > 5)) {

cout << "Ваш выбор: ";

cin >> inputCommandCode;

}

switch (inputCommandCode) {

case 1:

myComposition.printBoard();

myComposition.printInfo();

break;

case 2:

myComposition.initBoard();

cout << "Доска очищена" << endl;

myComposition.printBoard();

myComposition.printInfo();

break;

case 3:

cout << "Добавление фигуры:" << endl;

myComposition.printInfo();

figureCode = -1;

while ((figureCode < 1) || (figureCode > 6)) {

cout << "Введите код фигуры: ";

cin >> figureCode;

}

x = -1;

while ((x < 1) || (x > 8)) {

cout << "Введите координату фигуры по горизонтали (от 1 до 8): ";

cin >> x;

}

y = -1;

while ((y < 1) || (y > 8)) {

cout << "Введите координату фигуры по вертикали (от 1 до 8): ";

cin >> y;

}

myComposition.addFigure(figureCode, x-1, y-1);

cout << "Фигура добавлена!" << endl << endl;

myComposition.printBoard();

myComposition.printInfo();

break;

case 4:

cout << "Удаление фигуры:" << endl;

x = -1;

while ((x < 1) || (x > 8)) {

cout << "Введите координату фигуры по горизонтали (от 1 до 8): ";

cin >> x;

}

y = -1;

while ((y < 1) || (y > 8)) {

cout << "Введите координату фигуры по вертикали (от 1 до 8): ";

cin >> y;

}

myComposition.removeFigure(x-1, y-1);

cout << "Фигура удалена!" << endl << endl;

myComposition.printBoard();

myComposition.printInfo();

break;

case 5:

exit = 1;

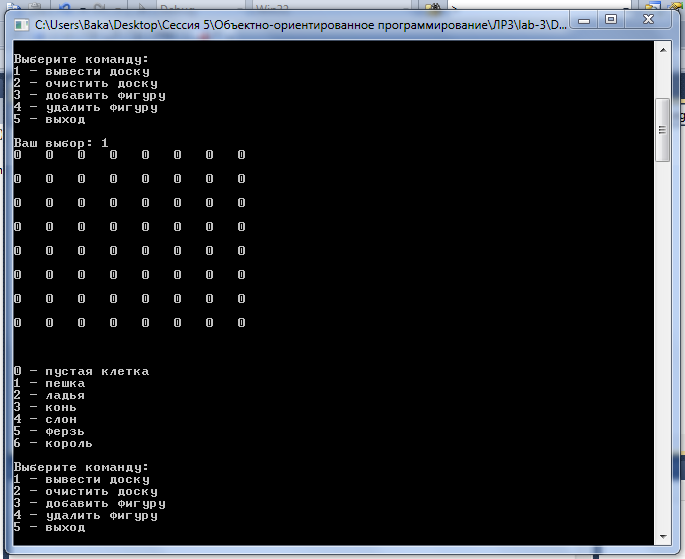
}

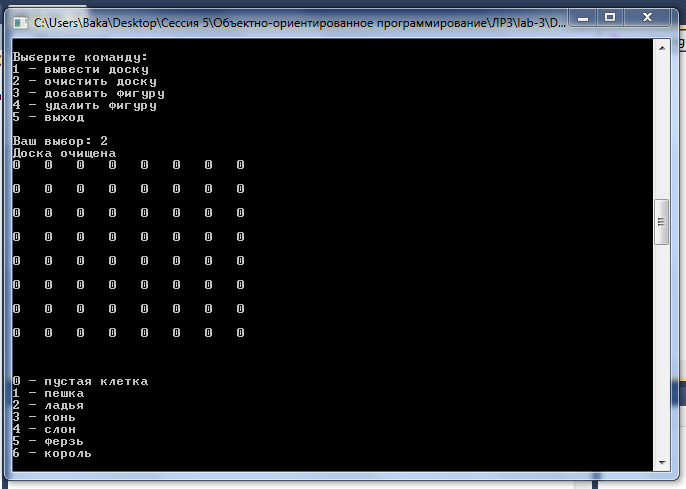
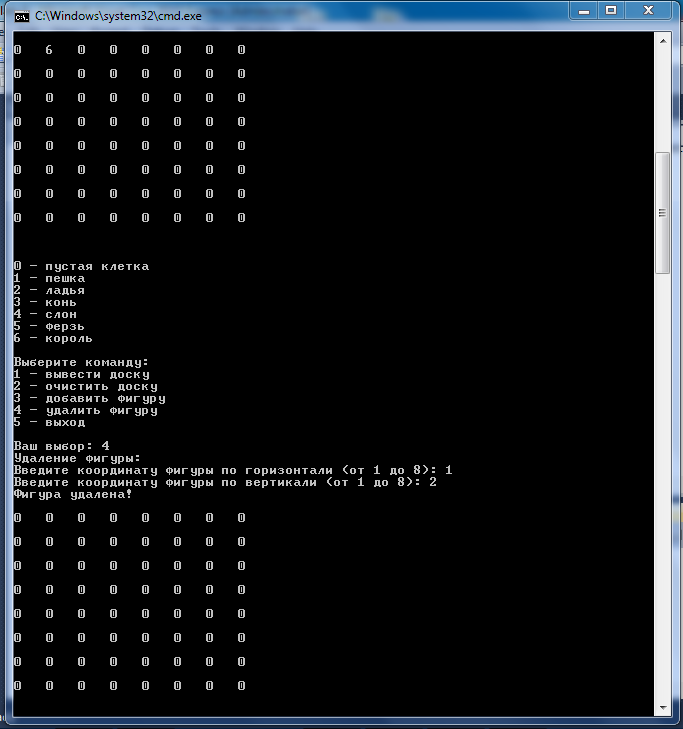
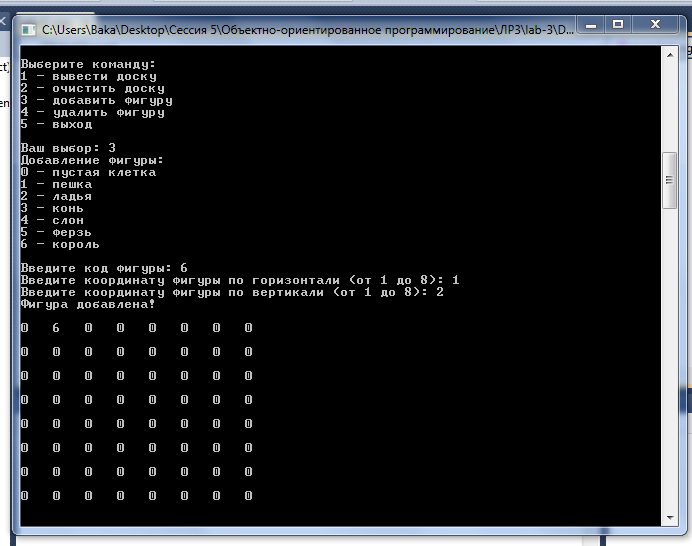
}

return 0;

}

**Результат работы программы**





**Вывод**

Во время выполнения лабораторной работы были изучены механизмы создания нового класса на основе уже существующего, варианты доступа к элементам базового класса из производного при программировании на языке C++.