Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет)

Факультет «Робототехника и комплексная автоматизация»

Кафедра «Системы автоматизированного проектирования»

**Отчет по практической работу**

По курсу «Проектно-технологическая практика»

Выполнил: Студент Сергеева Д.К.

Группа РК6-36Б

Проверил: \_\_\_КузьминаИ.А.\_\_\_\_\_

Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2020 г.

**Вариант G18**

**Задание:**

Разработать ООП для компьютерной реализации игры с переходами одностороннего короля на шахматной доске по следующим правилам. Сначала король стоит в поле a1. Два игрока должны по очереди переставлять его по горизонтали, вертикали или диагонали на 1 клетку в направлении поля h8. Проигрывает тот, кто не может сделать свой очередной ход. Эта игра должна быть реализована в варианте, когда противниками являются компьютер и человек. Человек может делать произвольные ходы по запросу стандартного ввода. При этом любое нарушение правил должно приводить к аварийному прерыванию игры. Компьютер должен отвечать ходами короля по полям с четным смещением по горизонтали и по вертикали от a1, со стандартным выводом его положения на шахматной доске. Игра должна завершаться диагностикой ее итогового результата.

**Алгоритм:**

На экран при каждом ходе игроков (компьютера и человека) выводится шахматная доска, где поле, на которой стоит король, помечается литерой “K”, и где все доступные ходы отмечены знаком “+”. Игра будет продолжаться до тех пор, пока фигура не окажется на клетке “h8”. Игроки ходят по очереди.

**Входные данные:**

Отсутствуют входные данные.

**Выходные данные:**

Шахматная доска с указанной буквой “K” - позицией короля; ходы игкроков

**Текст программы:**

**Main.cpp**

#include "Gambler.h"

#include <iostream>

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

{

//Initialization start position

char\* startPos = (char\*)"a1";

char\* endPos = (char\*)"h8";

//Init Game

Figure f(startPos);

f.printBoard();

Gambler\* players[] = {

new Man(f, "Man"),

new Pen(f, "Computer")

};

//Start Game

int i = 1;

while (f != endPos)

{

players[i%2]->query();

if (!players[i%2]->move())

{

std::cout << "Invalid input\n";

return 0;

}

f.printBoard();

i++;

}

std::cout << std::endl << players[(i+1) % 2]->getName()<<" Wins"<<std::endl;

delete players[0];

delete players[1];

return 0;

}

**Gambler.cpp**

#include "Gambler.h"

const char\* Gambler::getName()

{

return \_name;

}

Gambler::Gambler(Figure& f)

{

\_figure = &f;

}

void Gambler::query()

{

std::cout << \_name <<": "<< (\*\_figure)<<"-"<<\_figure->isA();

return;

}

Man::Man(Figure& f, const char\* n) :Gambler(f)

{

\_name = n;

}

bool Man::move()

{

char s[3];

std::cin.unsetf(std::ios::skipws);

std::cin >> s[0] >> s[1];

s[2] = '\0';

std::cin.ignore(64, '\n');

if (Figure::deskOut(s) || (\_figure->attack(s) == 0) || (\*\_figure) == s)

return false;

(\*\_figure) = s;

return true;

}

Pen::Pen(Figure& f, const char\* n) :Gambler(f)

{

\_name = n;

}

bool Pen::move()

{

static char s[3];

s[0] = \_figure->getAlp();

s[1] = \_figure->getNum();

s[2] = '\0';

s[0] = (s[0] - 'a') % 2 != 0 ? s[0] : (s[0] + 1);

s[1] = (s[1] - '1') % 2 != 0 ? s[1] : (s[1] + 1);

(\*\_figure) = s;

std::cout << s;

std::cin.ignore(64, '\n');

return true;

}

**Figure.cpp**

#include "Figure.h"

Figure::Figure(const char\* s)

{

\_position[0] = s[0];

\_position[1] = s[1];

}

Figure::Figure()

{

\_position[0] = 'a';

\_position[1] = '1';

}

char Figure::getAlp()

{

return \_position[0];

}

char Figure::getNum()

{

return \_position[1];

}

char Figure::isA()

{

return 'K';

}

int Figure::attack(char\* p)

{

if (deskOut(p) > 0)

return 0;

int x = (int)(\_position[0] - p[0]);

int y = \_position[1] - p[1];

if ((x > 0) || (y > 0))

return 0;

x \*= -1;

y \*= -1;

if ((x > 1) || (y > 1))

return 0;

return 1;

}

int Figure::deskOut(char\* pos)

{

if (strlen(pos) == 2)

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

return 1;

}

void Figure::printBoard()

{

char s[3];

s[2] = '\0';

const char\* mark = ".+";

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

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

std::cout << i << ' ';

s[1] = '0' + i;

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

s[0] = 'a' + j;

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

std::cout << m << ' ';

}

std::cout << i << "\n";

}

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

}

Figure& Figure::operator=(char\* p)

{

\_position[0] = p[0];

\_position[1] = p[1];

return \*this;

}

int Figure::operator!=(char\* p)

{

return (\_position[0] != p[0]) || (\_position[1] != p[1]);

}

int Figure::operator==(char\* p)

{

return((\_position[0] == p[0]) && (\_position[1] == p[1]));

}

std::ostream& operator<<(std::ostream& out, Figure& f)

{

return out << f.isA() << f.\_position[0] << f.\_position[1];

}

**Figure.h**

#ifndef FIGUREH

#define FIGUREH

#include <iostream>

#include <string.h>

class Figure

{

protected:

char \_position[2];

public:

Figure(const char\*);

Figure();

char getAlp();

char getNum();

char isA();

int attack(char\*);

static int deskOut(char\*);

void printBoard();

Figure& operator=(char\*);

int operator!=(char\*);

int operator==(char\*);

friend std::ostream& operator<<(std::ostream&, Figure&);

};

#endif

**Gambler.h**

#ifndef GAMBLERH

#define GAMBLERH

#include "Figure.h"

#include <iostream>

class Gambler

{

protected:

Figure\* \_figure;

const char\* \_name;

public:

const char\* getName();

Gambler(Figure&);

void query();

virtual bool move() = 0;

};

class Man : public Gambler

{

public:

Man(Figure&, const char\*);

virtual bool move();

};

class Pen : public Gambler

{

public:

Pen(Figure&, const char\*);

virtual bool move();

};

#endif

**Тесты:**

|  |  |
| --- | --- |
| **Номер** | **Выход** |
| 1 |  |

**Список использованной литературы:**

* Волосатова Т.М., Родионов С.В. Лекции по курсу «Объектно-ориентированное программирование»
* bigor.bmstu.ru