АВТОНОМНАЯ НЕКОММЕРЧЕСКАЯ ОРГАНИЗАЦИЯ ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ "МЕЖДУНАРОДНАЯ АКАДЕМИЯ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ "ИТ ХАБ"

Локальная разработка. Линейные алгоритмы.

Выполнил: студент группы 3ИБ2

Минаев Д.

Москва 2024

Оглавление

[**1.** **Задача** 3](#_Toc157605937)

[**2.** **Вербальная модель решения** 3](#_Toc157605938)

[**3.** **Математическая модель решения** 3](#_Toc157605939)

[**4.** **Блок схема** 4](#_Toc157605940)

[**5.** **Программа** 5](#_Toc157605941)

[**6.** **Проверка** 5](#_Toc157605942)

[**7.** **Заключение** 5](#_Toc157605943)

[**8.** **Выводы** 6](#_Toc157605944)

[**9.** **Приложение** 6](#_Toc157605945)

# **Задача**

На шахматной доске стоят черный король и белые ладья и слон (ладья бьет по горизонтали и вертикали, слон – по диагоналям). Проверить, есть ли угроза королю и если есть, то от кого именно. Учесть возможность защиты (например, ладья не бьет через слона).

# **Вербальная модель решения**

Перенесем шахматную доску на двухмерную систему координат 'x,y’

Запрашиваем у пользователя расположение всех фигур в виде трех точек в системе координат

К(xK,yK), L(xL,yL), S(xS,yS).

Далее рассмотрим возможные сценарии шаха для короля:

1. Король находится в одной плоскости x или в одной плоскости с ладьей, король находится под угрозой ладьи.
2. Если слон перекрывает ладью на плоскости x или y то король не под угрозой
3. Разница значений координат плоскости x для короля и слона равна разницы координат плоскости y для короля и слона, король находится под угрозой.
4. Если слон перекрывает ладью на плоскости x или y то король не под угрозой
5. Пункт одни и два одновременной верны
6. Если координаты точки короля не подходят из выше описанный условий, то королю ничего не угрожает.

# **Математическая модель решения**

Дано:

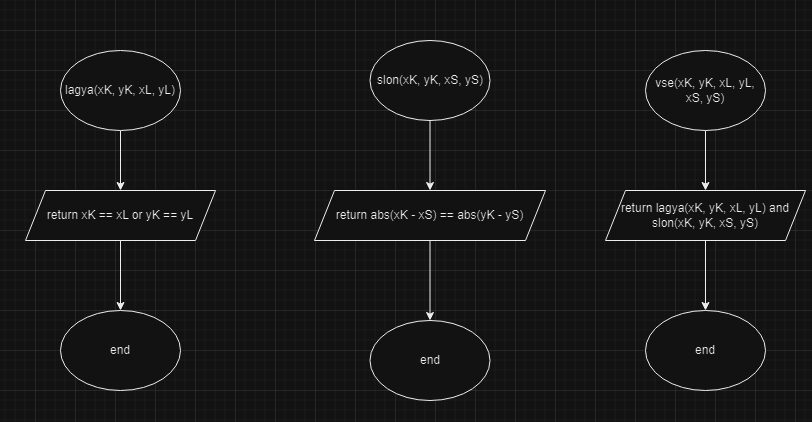
Координаты короля K(x,y)

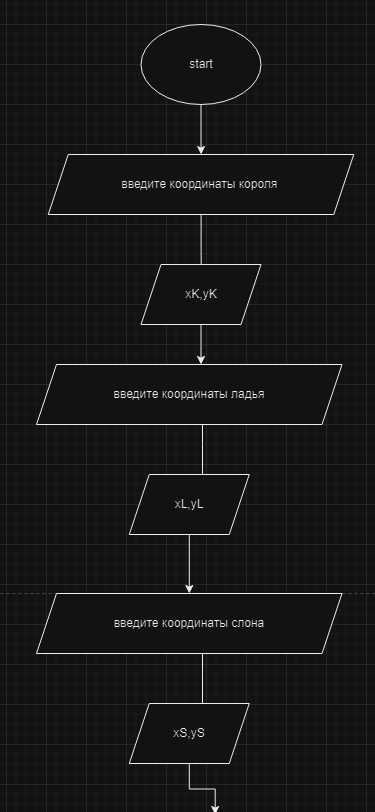
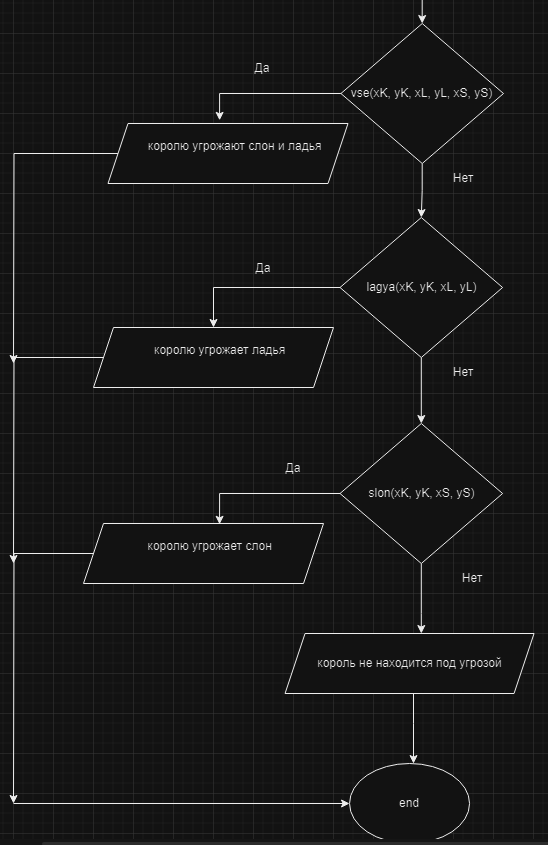
Координаты ладье Л(x,y)

Координаты слона С(x,y)

Решение:

# **Блок схема**



# **Программа**

**void** ChessBoard::inputCoordinates() {

**do** {

cout << "Введите координаты короля (1-8): ";

cin >> xK >> yK;

} **while** (xK < **1** || xK > **8** || yK < **1** || yK > **8**);

**do** {

cout << "Введите координаты ладьи (1-8): ";

cin >> xL >> yL;

} **while** (xL < **1** || xL > **8** || yL < **1** || yL > **8** || (xK==xL and yK==yL));

**do** {

cout << "Введите координаты слона (1-8): ";

cin >> xS >> yS;

} **while** (xS < **1** || xS > **8** || yS < **1** || yS > **8** || (xK==xS and yK==yS) || (xL==xS and yL==yS));

}

# **Проверка**

|  |  |  |
| --- | --- | --- |
| Ввод | Ожидание | Результат |
| К(3,3) Л(3,5) С(5,5) | Passed | Король под угрозой (Л,С) |
| К(3,2) Л(3,5) С(5,5) | Passed | Король под угрозой (Л) |
| К(2,2) Л(3,5) С(5,5) | Passed | Король под угрозой (С) |
| К(2,2) Л(2,5) С(2,4) | Passed | Король не под угрозой |
| К(2,6) Л(3,7) С(4,8) | Passed | Король не под угрозой |
| К(8,1) Л(8,7) С(1,8) | Passed | Король под угрозой (Л,С) |
| К(6,8) Л(6,3) С(1,8) | Passed | Король под угрозой (Л) |
| К(3,6) Л(6,3) С(6,6) | Passed | Король под угрозой (С) |
| К(2,4) Л(2,5) С(2,4) | Passed | Король не под угрозой |
| К(8,8) Л(6,6) С(1,1) | Passed | Король не под угрозой |

# **Заключение**

На основе проведенных тестов было показано, что вербальная, математическая модели составлены верно. Декомпозиция на процедуры и функции позволила запрограммировать задачу на языке С++

|  |  |
| --- | --- |
| Тесты | 10 из них верно 10 |
| Версия компилятора/интерпретатора | g++ (Debian 10.2.1-6) 10.2.1 20210110 |
| Объемы исполняемого кода | text data bss dec hex filename  4653 656 576 5885 16fd chess.out |
| Аппаратные ресурсы | 3723 0.0 0.0 6372 636 pts/2 S+ 14:59 0:00 grep chess.out |
| Зависимость от стандартных библиотек | linux-vdso.so.1 (0x00007ffeafbcd000)  libstdc++.so.6 => /lib/x86\_64-linux-gnu/libstdc++.so.6 (0x00007faf04e81000)  libc.so.6 => /lib/x86\_64-linux-gnu/libc.so.6 (0x00007faf04cad000)  libm.so.6 => /lib/x86\_64-linux-gnu/libm.so.6 (0x00007faf04b69000)  /lib64/ld-linux-x86-64.so.2 (0x00007faf0506d000)  libgcc\_s.so.1 => /lib/x86\_64-linux-gnu/libgcc\_s.so.1 (0x00007faf04b4f000) |
| Дополнительная информация | chess.out: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=af24693166adb79ad37a0828772e7568f81598b7, for GNU/Linux 3.2.0, not stripped |

1. **Выводы**

Данная программа позволяет определить, есть ли угроза Белому королю на шахматной доске со стоящими на ней черной ладьей и слоном. Ход слона возможен только по диагоналям, а ладья может перемещаться по горизонталям и вертикалям.

Для определения угрозы королю, программа проверяет все возможные положения, которые могут занять слон и ладья. Если одно из этих положений совпадает с координатами короля, то угроза существует. Вывод программы сообщит, есть ли угроза, и если да, то от кого именно – от слона или ладьи.

# **Приложение**

#include <iostream>

#include <cmath>

using namespace std;

class ChessBoard {

private:

int xK, yK; // Координаты короля

int xL, yL; // Координаты ладьи

int xS, yS; // Координаты слона

public:

ChessBoard();

void inputCoordinates();

bool isRookThreatening() const;

bool isBishopThreatening() const;

bool isKingUnderThreat() const;

void printThreatStatus() const;

};

ChessBoard::ChessBoard() {

}

void ChessBoard::inputCoordinates() {

do {

cout << "Введите координаты короля (1-8): ";

cin >> xK >> yK;

} while (xK < 1 || xK > 8 || yK < 1 || yK > 8);

do {

cout << "Введите координаты ладьи (1-8): ";

cin >> xL >> yL;

} while (xL < 1 || xL > 8 || yL < 1 || yL > 8 || (xK==xL and yK==yL));

do {

cout << "Введите координаты слона (1-8): ";

cin >> xS >> yS;

} while (xS < 1 || xS > 8 || yS < 1 || yS > 8 || (xK==xS and yK==yS) || (xL==xS and yL==yS));

}

bool ChessBoard::isRookThreatening() const {

if (xK == xL) {

if (xK==xS){

if (yS>min(yK,yL) and yS < max(yK,yL)){

return false;

}

else {return true;}

}

}

else if (yK == yL) {

if (yK==yS){

if (xS>min(xK,xL) and xS < max(xK,xL)){

return false;

}

else {return true;}

}

}

else {return false;}

return false;

}

bool ChessBoard::isBishopThreatening() const {

if (abs(xK - xS) == abs(yK - yS)){

if (xL <= min(xK,xS) || xL >= max(xK,xS) || yL <= min(yK,yS) || yL >= max(yK,yL)) {

return true;

}

else if (abs(xK - xL) == abs(yK - yL)) {

return false;

}

else { return true;}

}

else { return false;}

}

bool ChessBoard::isKingUnderThreat() const {

return isRookThreatening() && isBishopThreatening();

}

void ChessBoard::printThreatStatus() const {

if (isKingUnderThreat()) {

cout << "Королю угрожают слон и ладья" << endl;

} else if (isRookThreatening()) {

cout << "Королю угрожает ладья" << endl;

} else if (isBishopThreatening()) {

cout << "Королю угрожает слон" << endl;

} else {

cout << "Король не находится под угрозой" << endl;

}

}