**Постановка задачи**

Исполнитель умеет перемещаться по клетчатому полю 10 х 10. Он знает команды: ВЛЕВО, ВПРАВО, ВВЕРХ и ВНИЗ, смещаясь по каждой из них на одну клетку поля в указанном направлении (в случае выхода за границы поля выполнение алгоритма прекращается выдачей диагностического сообщения). Кроме того, исполнитель понимает команду цикла, имеющую вид

ПОВТОРИТЬ N

..........

КЦ

где после слова "повторить" стоит ровно один пробел и N - целое число в интервале от 1 до 9. Внутри цикла может быть любое произвольное количество команд, в том числе и новый цикл.

Написать программу, которая транслирует введенный текст алгоритма для исполнителя и выполняет его, выдавая в качестве ответа координаты конечного поля. Считать, что исходное положение исполнителя - левая нижняя клетка с координатами (1,1). Предусмотреть синтаксический контроль вводимых команд.

**Определение идеи алгоритма**

**Выбор методов решения и структуры данных**

Для решения задачи была выбрана система программирования Microsoft Visual Studio.

Исходя из условия задачи на вход будет подаваться строка, т. к. команда подается текстовым сообщением. Кроме этого строка подается символами русского алфавита так что подключаем кодировку Windows-1251.

* Подключаем кодировку Windows-1251
* Добавляем инструменты для работы со строками (функции) char\_to\_digit, digit\_to\_char
* Создаем массивы для ввода команд, для хранения N, номера конца цикла
* Создаем цикл с помощью которого будем считывать в массив и выводить команды в консоли
* Создаем функцию для проверки написания команды
* Создаем функцию для ввода циклов, для дальнейшей работы с ним
* Создаем функции для перемещения исполнителя(относительно осей x и y)
* Создаем алгоритм завершения программы, то-есть при вводе пустой строки программа выдает финальный результат
* Создаем функцию для считывания команд из цикла
* Создаем функцию реализации цикла
* Создаем функцию проверки разрушения Исполнителя после перемещения
* Если Исполнитель не разрушен, то выводим конечные координаты
* Если Исполнитель разрушен, то выводим команду, на которой он разрушится

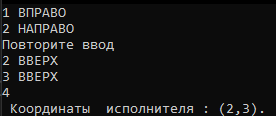
Далее если Исполнитель разрушен, то выводим ошибочную строку, но если Исполнитель не разрушен выводим Координаты исполнителя.

**Тестирование программы**

Проведя анализ возможного ввода данных можно выделить 5 классов входных комбинаций команд Исполнителя

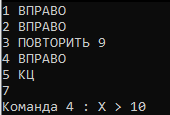
Далее приведу их пример:

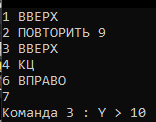
1. Введена не корректная команда, из-за чего программа просит повторить ввод, но продолжает свою работу:

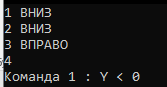


1. Исполнитель разрушается на одной из команд, программа выводит строку на которой именно происходит разрушение:

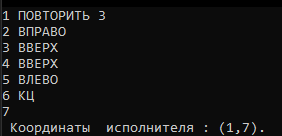




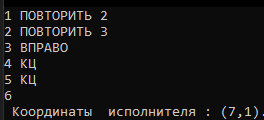




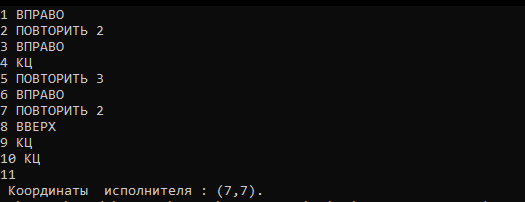
1. На вход подается цикл, алгоритм должен выводить корректные данные и завершаться при команде КЦ

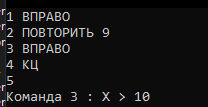


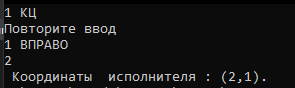
1. Проверить работу работу вложенного цикла



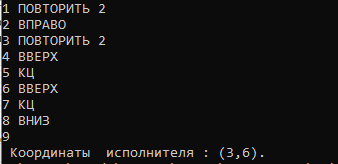
1. На вход подаются команды обычного перемещения, цикла, вложенного цикла



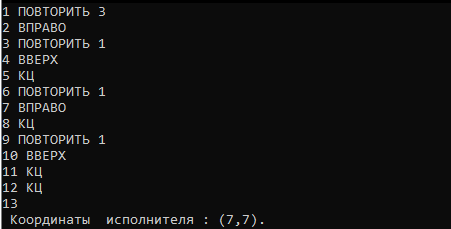
1. Программа обнаружила выход за границу при выполнении цикла 
2. Программа просит повторить ввод, т к цикл не запущен, но подано КЦ



1. Программа содержит команды вне вложенного цикла



1. Программа выдерживает до 4 вложенных циклов



**Код программы**

#include <iostream>

#include <string>

#include <sstream>

#include <windows.h>

#include <fstream>

using namespace std;

int char\_to\_digit(char ch) {

return ch - '0';

}

char digit\_to\_char(int x) {

return x + '0';

};

bool inputpr(string comand);

void left(int& x, int& y, bool& erorr, int i, string& xyeror);

void right(int& x, int& y, bool& erorr, int i, string& xyeror);

void up(int& x, int& y, bool& erorr, int i, string& xyeror);

void down(int& x, int& y, bool& erorr, int i, string& xyeror);

void vvodcircle(string mas\_comand[1000], int N[1000], int& i, string comand, int KC[1000]);

void circle(int& x, int& y, int n, bool& erorr, int i0, string& xyeror, string mas\_comand[1000], int N[1000], int KC[1000]);

void eror(int x, int y, bool& erorr, string& xyeror, int i);

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int y = 1, x = 1, i = 1, N[1000],KC[1000];

bool erorr = false;

string comand, xyeror,mas\_comand[1000];

for (cout << i << " ", i++; getline(cin, comand) && (comand != ""); cout << i << " ", i++) {

if (inputpr(comand) == true) {

if (comand == "ПОВТОРИТЬ 1" || comand == "ПОВТОРИТЬ 2" || comand == "ПОВТОРИТЬ 3" || comand == "ПОВТОРИТЬ 4" || comand == "ПОВТОРИТЬ 5" || comand == "ПОВТОРИТЬ 6" || comand == "ПОВТОРИТЬ 7" || comand == "ПОВТОРИТЬ 8" || comand == "ПОВТОРИТЬ 9") {

vvodcircle(mas\_comand, N, i, comand, KC);

}

else {

mas\_comand[i] = comand;

N[i] = 0;

}

}

else {

cout << "Повторите ввод" << endl;

i--;

};

};

for (int i1 = 1; i1 < i && erorr == false; i1++) {

if (mas\_comand[i1] == "ВЛЕВО") left(x, y, erorr, i1, xyeror);

else

if (mas\_comand[i1] == "ВПРАВО") right(x, y, erorr, i1, xyeror);

else

if (mas\_comand[i1] == "ВВЕРХ") up(x, y, erorr, i1, xyeror);

else

if (mas\_comand[i1] == "ВНИЗ") down(x, y, erorr, i1, xyeror);

else

if (mas\_comand[i1] == "ЦИКЛ") circle(x, y, N[i1], erorr, i1, xyeror, mas\_comand, N,KC);

}

if (erorr == false) cout << " Координаты исполнителя : (" << x << "," << y << ").";

else

cout << xyeror;

}

void vvodcircle(string mas\_comand[1000], int N[1000], int& i, string comand,int KC[1000]){

mas\_comand[i] = "ЦИКЛ";

int i0 = i;

N[i] = char\_to\_digit(comand[10]);

for (cout << i << " ",i++; getline(cin, comand) && (comand != "КЦ"); cout << i << " ", i++) {

if (comand == "") {

cout << "Повторите ввод" << endl;

i--;

}else {

if (inputpr(comand) == true) {

if (comand == "ПОВТОРИТЬ 1" || comand == "ПОВТОРИТЬ 2" || comand == "ПОВТОРИТЬ 3" || comand == "ПОВТОРИТЬ 4" || comand == "ПОВТОРИТЬ 5" || comand == "ПОВТОРИТЬ 6" || comand == "ПОВТОРИТЬ 7" || comand == "ПОВТОРИТЬ 8" || comand == "ПОВТОРИТЬ 9") {

vvodcircle(mas\_comand, N, i, comand,KC);

}

else {

mas\_comand[i] = comand;

N[i] = 0;

}

}

else {

cout << "Повторите ввод" << endl;

i--;

};

};

};

KC[i0] = i;

mas\_comand[i] = "КЦ";

}

void circle(int& x, int& y, int n, bool& erorr, int i0, string& xyeror, string mas\_comand[1000], int N[1000],int KC[1000]) {

for (int n0 = 1; n0 < n && erorr == false; n0++) {

for (int i = i0 + 1; i<KC[i0]; i++) {

if (mas\_comand[i] == "ВЛЕВО") left(x, y, erorr, i, xyeror);

else

if (mas\_comand[i] == "ВПРАВО") right(x, y, erorr, i, xyeror);

else

if (mas\_comand[i] == "ВВЕРХ") up(x, y, erorr, i, xyeror);

else

if (mas\_comand[i] == "ВНИЗ") down(x, y, erorr, i, xyeror);

else

if (mas\_comand[i] == "ЦИКЛ") circle(x, y, N[i], erorr, i, xyeror, mas\_comand, N, KC);

}

}

}

void eror(int x, int y, bool& erorr, string& xyeror, int i) {

if (x < 1) {

xyeror.append("Команда ");

xyeror += digit\_to\_char(i - 1);

xyeror.append(" : X < 0");

erorr = true;

};

if (y < 1) {

xyeror.append("Команда ");

xyeror += digit\_to\_char(i - 1);

xyeror.append(" : Y < 0");

erorr = true;

};

if (x > 10) {

xyeror.append("Команда ");

xyeror += digit\_to\_char(i - 1);

xyeror.append(" : X > 10");

erorr = true;

};

if (y > 10) {

xyeror.append("Команда ");

xyeror += digit\_to\_char(i - 1);

xyeror.append(" : Y > 10");

erorr = true;

};

}

bool inputpr(string comand) {

if (comand == "ВЛЕВО" || comand == "ВПРАВО" || comand == "ВНИЗ" || comand == "ВВЕРХ" || comand == "ПОВТОРИТЬ 1" || comand == "ПОВТОРИТЬ 2" || comand == "ПОВТОРИТЬ 3" || comand == "ПОВТОРИТЬ 4" || comand == "ПОВТОРИТЬ 5" || comand == "ПОВТОРИТЬ 6" || comand == "ПОВТОРИТЬ 7" || comand == "ПОВТОРИТЬ 8" || comand == "ПОВТОРИТЬ 9") return true;

else return false;

}

void left(int& x, int& y, bool& erorr, int i, string& xyeror) {

x--;

eror(x, y, erorr, xyeror, i);

}

void right(int& x, int& y, bool& erorr, int i, string& xyeror) {

x++;

eror(x, y, erorr, xyeror, i);

}

void up(int& x, int& y, bool& erorr, int i, string& xyeror) {

y++;

eror(x, y, erorr, xyeror, i);

}

void down(int& x, int& y, bool& erorr, int i, string& xyeror) {

y--;

eror(x, y, erorr, xyeror, i);

}