МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Кафедра информатики и систем управления

Рекурсия и головоломки
(наименование работы)

ОТЧЕТ

по лабораторной работе №4

по дисциплине

Технологии программирования
(наименование дисциплины)

РУКОВОДИТЕЛЬ:

_____ Капранов С.Н.
(подпись) (фамилия, и., о.)

СТУДЕНТ:

_____ Максимова Е.И.
(подпись) (фамилия, и., о.)

18-ИСТ-4
(шифр группы)

Работа защищена «___» _____

С оценкой _____

Нижний Новгород, 2020

Содержание

Задача

9 вариант:

Фонари ("Light Up", "Akari", "Bijutsukan") — это логическая головоломка. Игровое поле состоит из белых и черных клеток; в некоторых черных клетках расположены числа. Необходимо разместить "светильники" в белых клетках таким образом, чтобы все игровое поле было освещено, но фонари не "светили" бы друг на друга.

Свет фонаря распространяется по горизонтали и по вертикали, но может быть заблокирован черной клеткой. В черной клетке может находиться число от 0 до 4, указывая, сколько фонарей должно быть размещено рядом с ней (не учитываются фонари, помещенные по диагонали от этой черной клетки). Если клетка не содержит числа, около нее может быть размещено любое количество фонарей.

Основная часть отчета

Программа написана на языке c++ в среде разработки Visual Studio 2019.

Листинг программы

```
#include <iostream>
#include <fstream>
#include <cmath>
using namespace std;
bool relay = false;   //переменная для проверки, нашлось ли решение

//структура клетки поля
struct cage {
	char color = 'w';				//b-чёрный, w-белый, L-стоит фонарь
	short int value;				//значение в черной клетке;
		//переменная для проверки можно ли ставить
фонарь
	bool glim = false;				//есть свет в клетке или нет
	short int intersections = 0;		//количество пересечений света в клетке
	bool lantern = true;				//нужно ставить рядом с чёрной клеткой
фонари или нет
};

//функция, определяющая стоят ли данные клетки рядом (НЕ по диагонали)
//принимает координаты i и j 1-й и 2-й клеток
//возвращает 1 если стоят рядом, 0 если не рядом
int proxCells(short int i1, short int j1, short int i2, short int j2) {
	if ((i1 - i2 == 0 && abs(j1 - j2) == 1) || (j1 - j2 == 0 && abs(i1 - i2) == 1))
		return 1;
	else
		return 0;
}

//считает кол-во фонарей, которое можно поставить около данной клетки
//принимает массив-поле, координаты клетки около которой считаем кол-во фонарей
//которое можно поставить, размер поля
//возвращает число, соответствующее кол-ву фонарей, которые можно поставить около клетки
int byLamp(cage** mas, short int i, short int j, int size) {
	short int lamp = 0;
	if ((i - 1) >= 0)
		if (mas[i - 1][j].id)
			lamp++;
	if ((i + 1) < size)
		if (mas[i + 1][j].id)
			lamp++;
	if ((j - 1) >= 0)
		if (mas[i][j - 1].id)
			lamp++;
	if ((j + 1) < size)
		if (mas[i][j + 1].id)
			lamp++;
	return lamp;
}

//распределяет свет от фонарей на поле
```

```
//принимает массив-поле, координаты фонаря от которого распространяется свет, размер поля
void lightSpreads(cage** mas, short int i, short int j, int size) {
	int x = i;
	int y = j;
	//распространяем свет пока не встретили черный квадрат
	//распространение света ВНИЗ
	while ((x >= 0) && (mas[x][y].color != 'b'))
	{
		mas[x][y].intersections++;
		mas[x][y].id = false;
		mas[x][y].glim = true;
		x--;
	}
	x = i;
	y = j;
	//распространение света ВПРАВО
	while ((y < size) && (mas[x][y].color != 'b'))
	{
		mas[x][y].intersections++;
		mas[x][y].id = false;
		mas[x][y].glim = true;
		y++;
	}
	x = i;
	y = j;
	//распространение света ВВЕРХ
	while ((x < size) && (mas[x][y].color != 'b'))
	{
		mas[x][y].intersections++;
		mas[x][y].id = false;
		mas[x][y].glim = true;
		x++;
	}
	x = i;
	y = j;
	//распространение света ВЛЕВО
	while ((y >= 0) && (mas[x][y].color != 'b'))
	{
		mas[x][y].intersections++;
		mas[x][y].id = false;
		mas[x][y].glim = true;
		y--;
	}
}

//расстановка фонарей, которые будут 100% стоять на заданном месте
//принимает массив-поле и размер этого поля size (поле size x size)
void surelyLight(cage** mas, int size) {
	int light_nearby = 0;	//кол-во ламп вокруг черной клетки
	bool round = false;	//переменная чтобы узнать нужно ли еще вызывать ф-ю
	//проводим следующие операции около черных клеток
	for (int i = 0; i < size; i++)
	{
		for (int j = 0; j < size; j++)
		{
```

```
if (mas[i][j].color == 'b' && mas[i][j].value != 5 && mas[i][j].value != 0)
{
	light_nearby = 0;		//кол-во ламп около черной клетки
	//считаем сколько ламп стоит около черной клетки
	if ((i - 1) >= 0)//данные проверки нужны чтобы узнать, что мы не
выходим за пределы поля
		if (mas[i - 1][j].color == 'L')
		{
			light_nearby++;
		}
	if ((i + 1) <= (size - 1))
		if (mas[i + 1][j].color == 'L')
		{
			light_nearby++;
		}
	if ((j - 1) >= 0)
		if (mas[i][j - 1].color == 'L')
		{
			light_nearby++;
		}
	if ((j + 1) <= (size - 1))
		if (mas[i][j + 1].color == 'L')
		{
			light_nearby++;
		}
	//запрещаем ставить фонари у тех черных клеток у которых
	//уже поставлено нужное кол-во фонарей
	if (light_nearby == mas[i][j].value)
	{
		if ((i - 1) >= 0)
			if (mas[i - 1][j].color != 'L')
			{
				mas[i - 1][j].id = false;
			}
		if ((i + 1) <= (size - 1))
			if (mas[i + 1][j].color != 'L')
			{
				mas[i + 1][j].id = false;
			}
		if ((j - 1) >= 0)
			if (mas[i][j - 1].color != 'L')
			{
				mas[i][j - 1].id = false;
			}
		if ((j + 1) <= (size - 1))
			if (mas[i][j + 1].color != 'L')
			{
				mas[i][j + 1].id = false;
			}
	}

	//если кол-во фонарей, которое можно поставить равно цифре
внутри чёрного квадрата
	// то ставим в клетки рядом фонари (mas.color = 'L')
```

```
                                        if (byLamp(mas, i, j, size) + light_nearby == mas[i][j].value &&
mas[i][j].lantern)
                                        {
                                                if ((i - 1) >= 0)
                                                        if (mas[i - 1][j].id)
                                                        {
                                                                mas[i - 1][j].color = 'L';
                                                                lightSpreads(mas, i - 1, j, size);
                                                        }
                                                if ((i + 1) <= (size - 1))
                                                        if (mas[i + 1][j].id)
                                                        {
                                                                mas[i + 1][j].color = 'L';
                                                                lightSpreads(mas, i + 1, j, size);
                                                        }
                                                if ((j - 1) >= 0)
                                                        if (mas[i][j - 1].id)
                                                        {
                                                                mas[i][j - 1].color = 'L';
                                                                lightSpreads(mas, i, j - 1, size);
                                                        }
                                                if ((j + 1) <= (size - 1))
                                                        if (mas[i][j + 1].id)
                                                        {
                                                                mas[i][j + 1].color = 'L';
                                                                lightSpreads(mas, i, j + 1, size);
                                                        }
                                                mas[i][j].lantern = false;
                                                round = true;
                                        }
                                }
                        }
                }
}

//убираем свет от фонарей на поле (обратная функции lightSpreads)
//принимает массив-поле, координаты фонаря от которого убираем распр. света, размер поля
void spreadsDark(cage** mas, short int i, short int j, int size) {
        int x = i;
        int y = j;
        while ((x >= 0) && (mas[x][y].color != 'b'))
        {
                mas[x][y].intersections--;
                if (mas[x][y].intersections <= 0)
                {
                        mas[x][y].id = true;
                        mas[x][y].glim = false;
                }
                x--;
        }
        x = i;
        y = j;
        while ((y < size) && (mas[x][y].color != 'b'))
        {
                mas[x][y].intersections--;
```

```
                if (mas[x][y].intersections <= 0)
                {
                        mas[x][y].id = true;
                        mas[x][y].glim = false;
                }
                y++;
        }
        x = i;
        y = j;
        while ((x < size) && (mas[x][y].color != 'b'))
        {
                mas[x][y].intersections--;
                if (mas[x][y].intersections <= 0)
                {
                        mas[x][y].id = true;
                        mas[x][y].glim = false;
                }
                x++;
        }
        x = i;
        y = j;
        while ((y >= 0) && (mas[x][y].color != 'b'))
        {
                mas[x][y].intersections--;
                if (mas[x][y].intersections <= 0)
                {
                        mas[x][y].id = true;
                        mas[x][y].glim = false;
                }
                y--;
        }
}

//проверяем в каком состоянии находится поле
//принимается массив-поле и его размер
//возвращает 0-если можно еще поставить фонари
//1 - если найдено решение
//2 - если решения с данной конфигурацией быть не может
int test(cage** mas, int size) {
        short int lamp = 0;
        int for_1 = 0;
        int count = 0;
        //проверяем правильно ли стоят фонари около черных клеток
        for (int i = 0; i < size; i++)
                for (int j = 0; j < size; j++)
                {
                        if (mas[i][j].color == 'b' && mas[i][j].value != 5)
                        {
                                count++;
                                lamp = 0;
                                if ((i - 1) >= 0)
                                        if (mas[i - 1][j].color == 'L')
                                                lamp++;
                                if ((i + 1) < size)
                                        if (mas[i + 1][j].color == 'L')
```

```
                                        lamp++;
                                if ((j - 1) >= 0)
                                        if (mas[i][j - 1].color == 'L')
                                                lamp++;
                                if ((j + 1) < size)
                                        if (mas[i][j + 1].color == 'L')
                                                lamp++;
                                if (lamp > mas[i][j].value)
                                        return 2;
                                if (lamp == mas[i][j].value)
                                        for_1++;
                        }
                }
        int glim = 0;
        int id = 0;
        //проверяем во всех ли белых клетках горит свет
        for (int i = 0; i < size; i++)
                for (int j = 0; j < size; j++)
                {
                        if (mas[i][j].glim == true)
                                glim++;
                        if (mas[i][j].id == true)
                                id++;
                }
        if ((glim == size * size) && (count == for_1))
                return 1;
        if (id == 0)
                return 2;
        return 0;
}

//ищем места, где фонари 100% НЕ МОГУТ стоять
//принимает массив-поле и его размер
void noLight(cage** mas, int size) {
        //ищем белые клетки, на которых можно запретить ставить фонари
        for (int i = 0; i < size; i++)
                for (int j = 0; j < size; j++)
                {
                        //если мы встретили белую клетку мы распространяем от нее свет
                        //если этот свет сделает невозможным растановку других фонарей то
                        //запрещаем на этом месте ставить фонарь (mas.id = false)
                        if (mas[i][j].color == 'w')
                        {
                                lightSpreads(mas, i, j, size);

                                //обход вниз и вверх
                                for (int y = i; y >= 0; y--)
                                {
                                        if (j - 1 >= 0)
                                                if (mas[y][j - 1].color == 'b' && mas[y][j - 1].value != 5 &&
mas[y][j - 1].value != 0)
                                                {
                                                        if (mas[y][j - 1].value > byLamp(mas, y, j - 1, size)
+ proxCells(i, j, y, j - 1))
                                                        {
```

```
                                                //принимает q значит в эту клетку
фонарь НЕ МОЖЕМ поставить
                                                mas[i][j].color = 'q';
                                                break;
                                        }
                                }
                        if (j + 1 < size)
                                if (mas[y][j + 1].color == 'b' && mas[y][j + 1].value != 5
&& mas[y][j + 1].value != 0)
                                {
                                        if (mas[y][j + 1].value > byLamp(mas, y, j + 1,
size) + proxCells(i, j, y, j + 1))
                                        {
                                                mas[i][j].color = 'q';
                                                break;
                                        }
                                }
                        if (mas[y][j].color == 'b' && mas[y][j].value != 5 &&
mas[y][j].value != 0)
                                if (mas[y][j].value > byLamp(mas, y, j, size) + proxCells(i,
j, y, j))
                                {
                                        mas[i][j].color = 'q';
                                        break;
                                }
                }
                //обход влево и вправо
                for (int x = j; x >= 0; x--)
                {
                        if (i - 1 >= 0)
                                if (mas[i - 1][x].color == 'b' && mas[i - 1][x].value != 5 &&
mas[i - 1][x].value != 0)
                                {
                                        if (mas[i - 1][x].value > byLamp(mas, i - 1, x, size)
+ proxCells(i, j, i - 1, x))
                                        {
                                                mas[i][j].color = 'q';
                                                break;
                                        }
                                }
                        if (i + 1 < size)
                                if (mas[i + 1][x].color == 'b' && mas[i + 1][x].value != 5
&& mas[i + 1][x].value != 0)
                                {
                                        if (mas[i + 1][x].value > byLamp(mas, i + 1, x,
size) + proxCells(i, j, i + 1, x))
                                        {
                                                mas[i][j].color = 'q';
                                                break;
                                        }
                                }
                        if (mas[i][x].color == 'b' && mas[i][x].value != 5 &&
mas[i][x].value != 0)
                                if (mas[i][x].value > byLamp(mas, i, x, size) + proxCells(i,
j, i, x))
```

```
                                        {
                                                mas[i][j].color = 'q';
                                                break;
                                        }
                                }
                                spreadsDark(mas, i, j, size);
                        }
                }
        //перепишем поле после обхода в нормальный вид
        for (int i = 0; i < size; i++)
                for (int j = 0; j < size; j++)
                {
                        if (mas[i][j].color == 'q')
                        {
                                mas[i][j].color = 'w';
                                mas[i][j].id = false;
                        }
                }
        //если встретили 0
        //запрещаем ставить рядом фонари
        for (int i = 0; i < size; i++)
                for (int j = 0; j < size; j++)
                        if (mas[i][j].value == 0 && mas[i][j].color == 'b')
                        {
                                if ((i - 1) >= 0)
                                {
                                        mas[i - 1][j].id = false;
                                }
                                if ((i + 1) < size)
                                {
                                        mas[i + 1][j].id = false;
                                }
                                if ((j - 1) >= 0)
                                {
                                        mas[i][j - 1].id = false;
                                }
                                if ((j + 1) < size)
                                {
                                        mas[i][j + 1].id = false;
                                }
                        }
}

//расставляем перебором на оставшихся местах
//принимает массив-поле и его размер
//возвращает 0-если ф-я завершила работу
int putOtherLamps(cage** mas, int size) {
        //если решение найдено то все вызванные ф-и прекращают работу
        if (relay)
        {
                return 0;
        }
        short int test_ = 0; //переменная для хранения результата теста
        //сделаем копию массива и будем работать с копией
        cage** mas1 = new cage * [size];
```

```
for (int i = 0; i < size; i++)
	mas1[i] = new cage[size];
for (int i = 0; i < size; i++)
	for (int j = 0; j < size; j++)
		mas1[i][j] = mas[i][j];

test_ = test(mas1, size);
//если решение найдено то выведем его на экран
//и завершим все другие ф-и (relay = true)
if (test_ == 1)
{
	for (int i = 0; i < 2 * size; i++)
		cout << char(196);
	cout << "\n";
	for (int i = 0; i < size; i++)
	{
		for (int j = 0; j < size; j++)
		{
			if (mas1[i][j].color == 'b' && mas1[i][j].value != 5) {
				cout << mas1[i][j].value << char(179);
			}
			else
				if (mas1[i][j].value == 5) {
					cout << char(219) << char(179);
				}
				else
					if (mas1[i][j].color == 'L') {

						cout << '*' << char(179);
					}
					else
						if (mas1[i][j].glim)
							cout << char(176) << char(179);
						else
							cout << char(179);
		}
		cout << '\n';
	}
	for (int i = 0; i < 2 * size; i++)
		cout << char(196);
	relay = true;

	for (int i = 0; i < size; i++)
		delete mas1[i];
	delete mas1;
	return 0;
}
else
	//если решения у данной расстановки поля нет, то завершим работу ф-и
	if (test_ == 2)
	{
		for (int i = 0; i < size; i++)
			delete mas1[i];
		delete mas1;
		return 0;
```

```
		}
	//если решения нет и расстановка поля нормальная то продолжим поиск решения
		else
			for (int i = 0; i < size; i++)
				for (int j = 0; j < size; j++)
					if (mas1[i][j].id)
					{
						lightSpreads(mas1, i, j, size);
						mas1[i][j].color = 'L';
						putOtherLamps(mas1, size);
						spreadsDark(mas1, i, j, size);
						mas1[i][j].color = 'w';
					}
	for (int i = 0; i < size; i++)
		delete mas1[i];
	delete mas1;
	return 0;
}
int main()
{
	int size;
	char symbol;
	ifstream file("field.txt");
	//вычиясляем размер поля: считаем соклько всего клеток
	for (size = 0;; size++)
	{
		file >> symbol;
		if (file.eof())
			break;
	}
	size = sqrt(size);
	file.clear();//чистим ошибки потока
	file.seekg(0, ios::beg);//смещает указатель "get" для текущего потока на 0
	cage** mas = new cage * [size];//динамическое выделение массива указателей,элементами
являются указатели на тип cage
	for (int i = 0; i < size; i++)
		mas[i] = new cage[size];

	//перепишем поля из файла
	for (int i = 0; i < size; i++)
		for (int j = 0; j < size; j++)
		{
			if (file.eof())
				break;
			file >> symbol;
			if (symbol == '-')
				continue;
			else
			{
				mas[i][j].value = symbol - 48;	//разница по аски коду
				mas[i][j].color = 'b';
				mas[i][j].id = false;
				mas[i][j].glim = true;
			}
		}
```

```
        // решение головоломки и её вывод на экран
        noLight(mas, size);
        surelyLight(mas, size);
        putOtherLamps(mas, size);
        file.close();
        getchar();
}
```

Входные и выходные данные