МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт информационных технологий
Кафедра вычислительной техники

КУРСОВАЯ РАБОТА

по дисциплине Объектно-ориентированное программирование

Тема курсового проекта (работы) Проектирование программной системы на базе сигналов и обработчиков

Студент группы ИКБО-07-19 Ле Д.К. _____
(подпись студента)

Руководитель курсового проекта (работы) ассистент Боронников А.С. _____
(подпись руководителя)

Работа представлена к защите « 4 » июня 2020 г.

Допущен к защите « 1 » июня 2020 г.

Москва 2020

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий
Кафедра вычислительной техники

Утверждаю
Заведующий кафедрой _____
(подпись)
Платонова О.В.
«21» февраля 2020 г.

ЗАДАНИЕ
на выполнение курсовой работы по дисциплине

« Объектно-ориентированное программирование »

Студент Ле Динь Кыонг Группа ИКБО-0719
Тема работы: Проектирование программной системы на базе сигналов и обработчиков

Исходные данные: АСО «Аврора», язык программирования C++
Перечень вопросов, подлежащих разработке, и обязательного графического материала:
1) Разработка базового класса на языке C++;
2) Разработка методов вывода иерархии объектов, поиска объектов по указателю;
3) Разработка методов сигналов и обработчиков;
4) Написание программы форматирования файла на базе сигналов и обработчиков.

Срок представления к защите курсовой работы: **до «** 30 **»** мая 2020 г**.**

Задание на курсовой проект (работу) выдал _____ (Боронников А.С.)
Подпись руководителя *Ф.И.О. Руководителя*
«21» февраля 2020 г.

Задание на курсовой проект (работу) получил _____ (Ле Д.К)
Подпись обучающегося *Ф.И.О. Обучающегося*

Москва 2020

Постановка задачи

Дано поле 10 x 10 позиций. Нумерация позиций по горизонтали и по вертикали начинается с 1. Поле 10 x 10 формируется в файле field.txt и первоначально все позиции заполняются 8.
На вход подаются последовательно данные: координата позиции и символ для отметки. Координата позиции задается посредством пары целых чисел от 1 до 10. Первое число номер строки, второе номер столбца. Начало координат верхний левый угол поля. Символ для отметки принадлежит латинскому алфавиту. Таких троек данных может быть множество. После получения очередной тройки **выдается** **сигнал** с текстом, содержащим координату и символ латинского алфавита. Если координата равна (0, 0), программа завершает работу. Первая строка входных данных не содержит (0, 0).

Использовать объекты:
1. Для установки позиции в файле согласно координате. Объект **выдает сигнал** об установке позиции или выводит в конце файла сообщение об ошибке.
2. Для записи символа в файле в установленную позицию. Объект пишет символ в установленной позиции или выводит в конце файла сообщение об ошибке.
3. Для вывода результата из файла на консоль. Все сообщения об ошибках пишутся построчно, с новой строки. Первое сообщение выводится с 11 строки (после поля 10 x 10).

Написать программу, реализующую следующий алгоритм:
1. Создание файла и формирование исходного содержания.
2. Ввод первой тройки данных.
3. Начало цикла.
   3.1. Выдача сигнала о вводе данных.
   3.2. Ввод очередной тройки данных и выдача сигнала.
4. Завершение цикла, если введена координата (0, 0).
5. Вывод на консоль содержимого файла.

Описание входных данных

Построчно множество координат и символ: «целое число»«целое число»«символ»

Описание выходных данных

Десять строк по десять символов в каждой, согласно сформированному в файле полю.

Если какая-либо координата не принадлежит интервалу [1, 10] то выводится сообщение:
Coordinate is wrong («значение», «значение»)

Если символ не принадлежит латинскому алфавиту:
Not a letter of the Latin alphabet: «символ»

Метод решения

Используя потоки Ввода/Вывода - cin/cout

Используя void buildFile для строить фаил

Используя void createNewFile() для создать новый фаил

Используя void scanPosition() для обрабатывать координаты

Используя int exec_app() для применять

Используя void printToScreen() для показать фаил

Используя void setSignal SIGNALING для передать координаты x, y,символов k и обрабатывать события, которые происходят

Используя void handling1 HANDLING1 и void handling2 HANDLING2 для записи символа в файле в установленную позицию.

Используя void showFile() для вывода результата из файла на консоль.

Описание алгоритма

int main()

№ шага	Предикат	Действие	№ перехода

1		cl_application file_application;
		file_application.buildFile();
		return file_application.exec_app();
			Ø

void cl_ base::setNameFile(string nameFile)

№ шага	Предикат	Действие	
1		this->nameFile = nameFile;	Ø

string cl_base::getNameFile()

1		return this->nameFile;	Ø

#define SIGNALING (cl_base* p_parent, int x, int y, char k)
void cl_base::setSignal (int x, int y, char k)

№ шага	Предикат	Действие	№ перехода
1	if (!((1 <= x && x <= 10)&&(1 <= y && y <= 10)))	addDataFile.open(nameFile, ios::app); handling1(this, x, y);	Ø
	else		2
2	if (!(('a' <= k && k <= 'z')||('A' <= k && k <= 'Z')))	addDataFile.open(nameFile, ios::app); handling2(this,k, false);	Ø

	else	addDataFile.open(nameFile, ios::in);
		int position = (x-1)*11 + y -1;
		addDataFile.seekp(position, ios::beg);
		handling(this, k, true);
			Ø

#define HANDLING1 (cl_base* p_parent, int x, int y)
void cl_base::handling1 (cl_base* p_parent, int x, int y)

№ шага	Предикат	Действие	№ перехода
1		addDataFile << endl << "Coordinate is wrong (" << x << ", " << y << ")"; addDataFile.close();	Ø

#define HANDLING2 (cl_base* p_parent, char k, bool letter)
void cl_base::handling2 (cl_base* p_parent, char k, bool letter)

№ шага	Предикат	Действие	№ перехода

1	if (letter)	addDataFile.put(k); addDataFile.close();	Ø
	else	addDataFile << endl << "Not a letter of the Latin alphabet: " << k; addDataFile.close();	Ø

void cl_base::showFile()

№ шага	Предикат	Действие	№ перехода
1		readFile.open(nameFile, ios::in);	2
2	if (!readFile)	return;	Ø
	else		3
3		string line;	4
4	while(getline(readFile, line))	cout << line;	5
	0		6
5	if (!readFile.eof())	cout << endl;	4
	else		6
6		readFile.close();	Ø

cl_application::cl_application()

№ шага	Предикат	Действие	№ перехода
1		string name ="file.txt"; toDo.setNameFile(name);	Ø

void cl_application::buildFile()

№ шага	Предикат	Действие	№ перехода

1		createNewFile();
		
		scanPosition();
			Ø

void cl_application::createNewFile()

№ шага	Предикат	Действие	№ перехода
1		ofstream newFile(toDo.getNameFile(),	2

		ios::out);	
2	if (!newFile)	return;	Ø
	else		3
3	for (int indexR = 1; indexR <= 10; indexR++)		4
	indexR > 10;		6
4	if (indexR != 1)	newFile << endl;	5
	else		5
5	for (int indexC = 1; indexC <= 10; indexC++)	int number = 8; newFile << number;	5
	indexC > 10		3
6		newFile.close();	Ø

void cl_application::scanPosition()

№ шага	Предикат	Действие	№ перехода
1	while(true)	cin >> x >> y;	2
2	if (x == 0 && y == 0)	break;	Ø
	else	char k; cin >>k; toDo.setSignal(this, x, y, k);	1

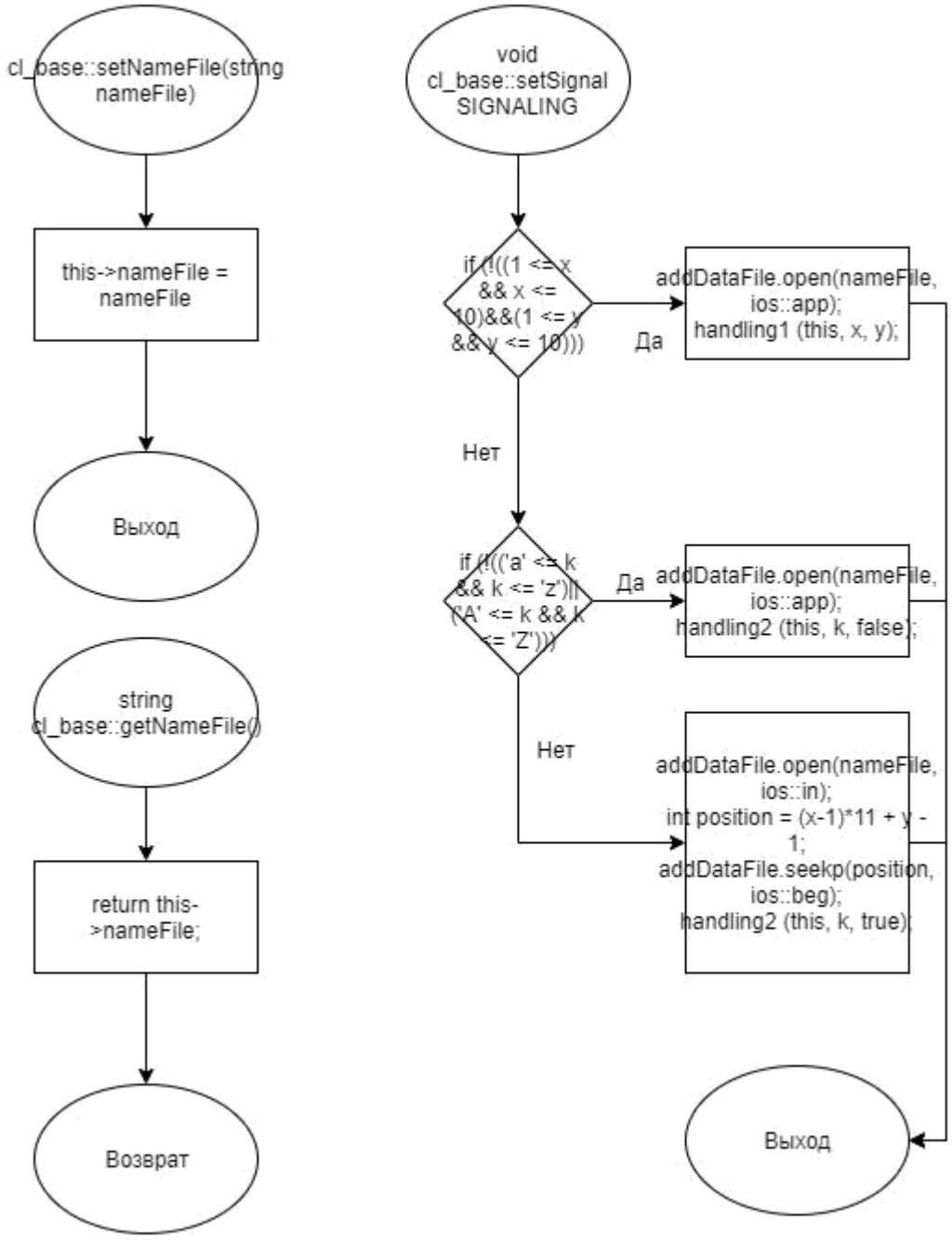
int cl_application::exec_app()

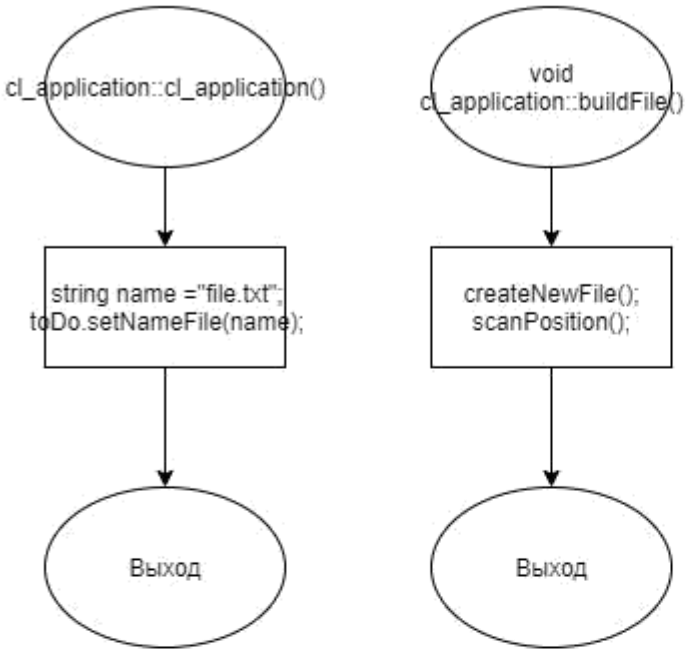
№ шага	Предикат	Действие	№ перехода
1		printToScreen(); return 0;	Ø

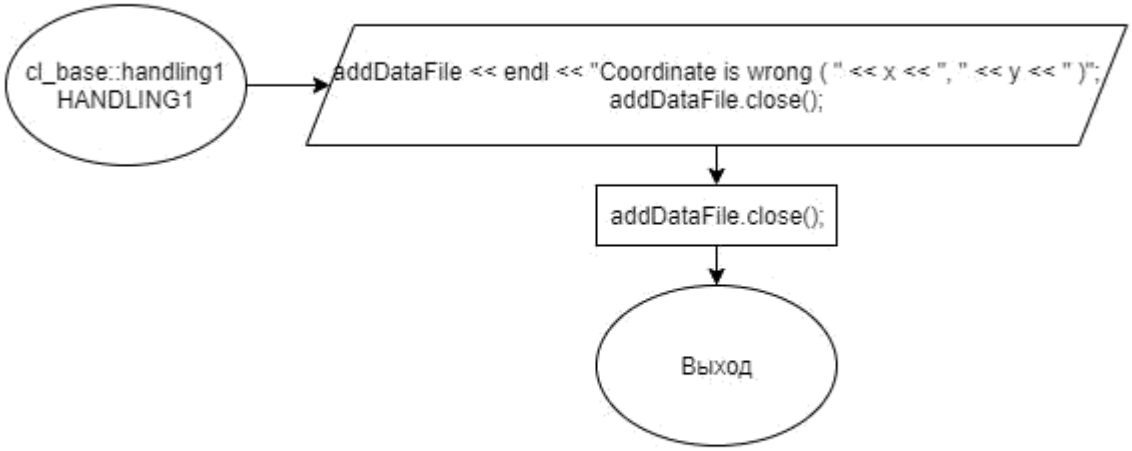
void cl_application::printToScreen()

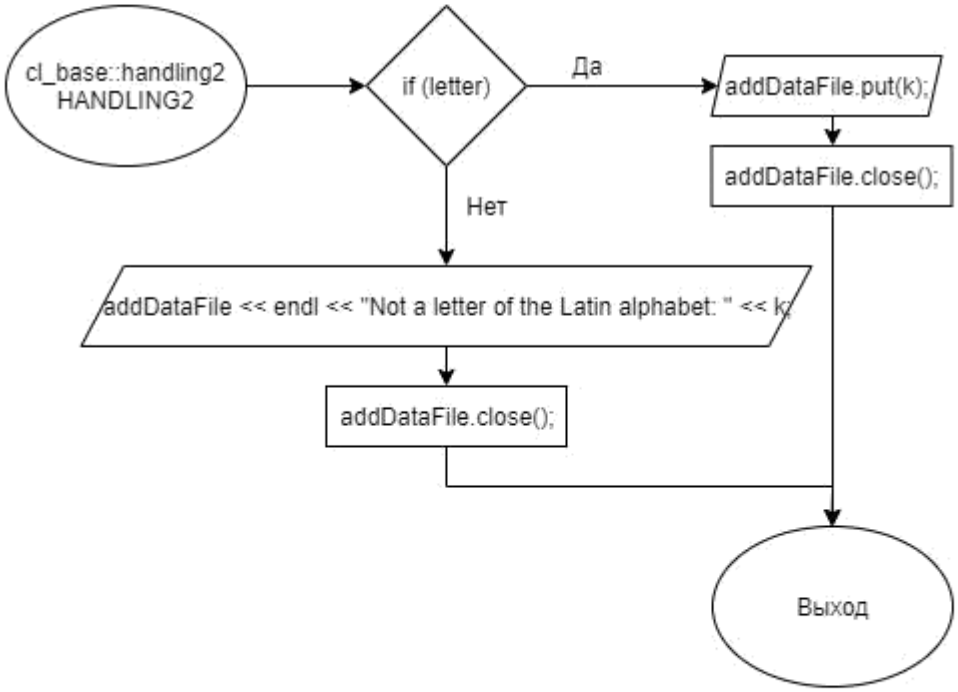
№ шага	Предикат	Действие	№ перехода
1		toDo.showFile();	Ø

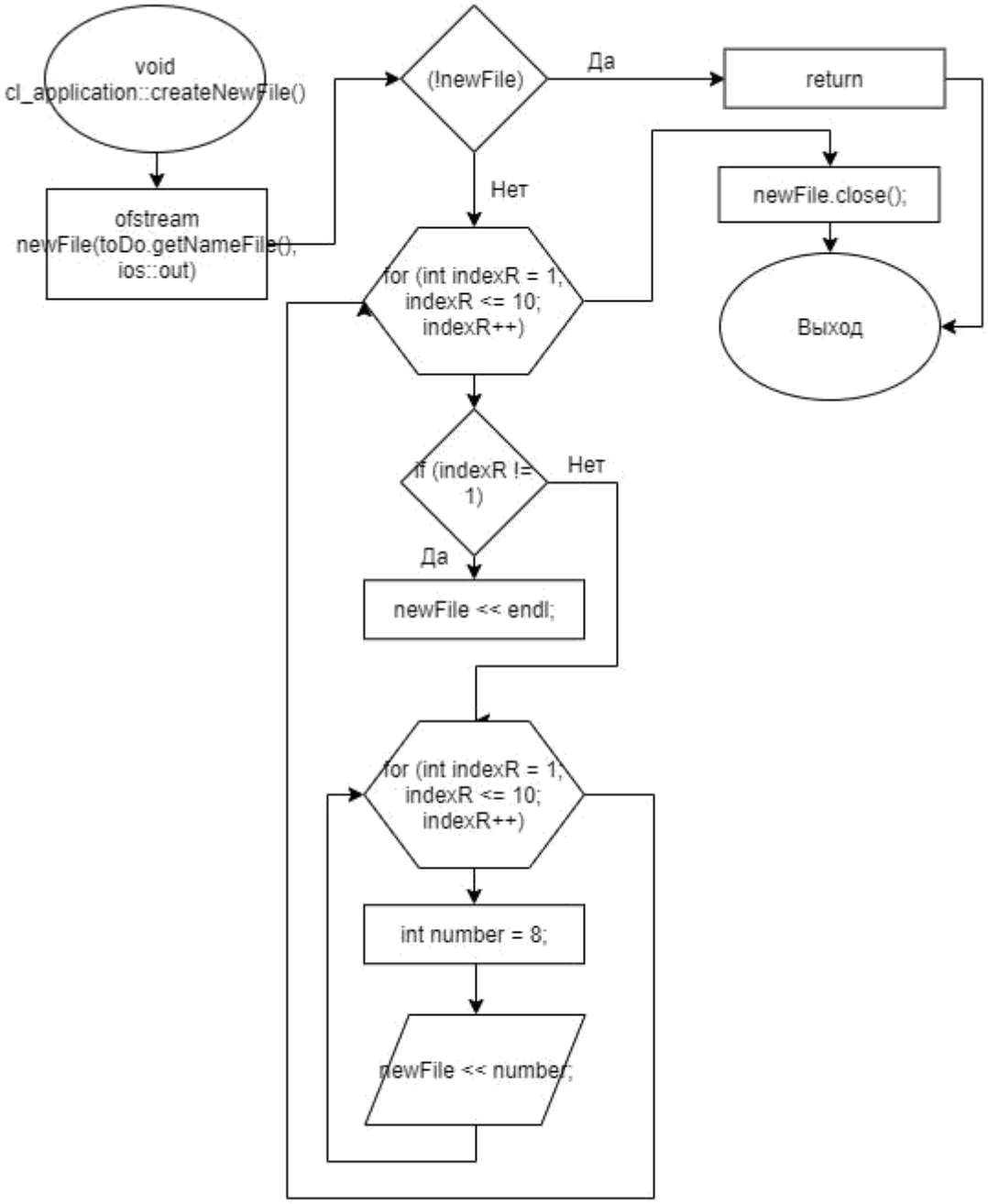
Блок-схема алгоритма

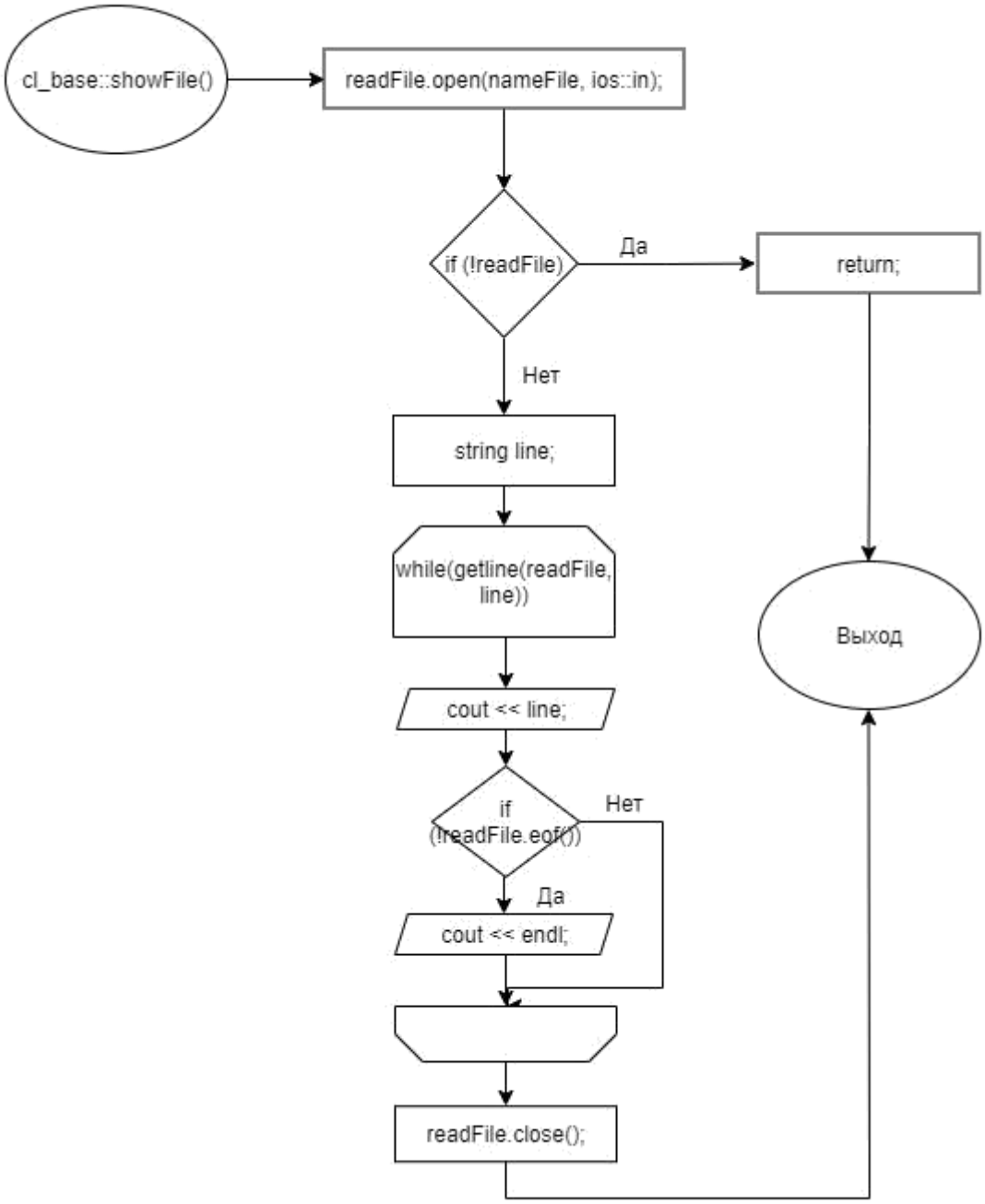

cl_base::setNameFile(string nameFile)
this->nameFile = nameFile
Выход
string cl_base::getNameFile()
return this->nameFile;
Возврат
void cl_base::setSignal SIGNALING
if (!((1 <= x && x <= 10)&&(1 <= y && y <= 10)))
Да
addDataFile.open(nameFile, ios::app); handling1 (this, x, y);
Нет
if (!(('a' <= k && k <= 'z')|| ('A' <= k && k <= 'Z')))
Да
addDataFile.open(nameFile, ios::app); handling2 (this, k, false);
Нет
addDataFile.open(nameFile, ios::in); int position = (x-1)*11 + y - 1; addDataFile.seekp(position, ios::beg); handling2 (this, k, true);
Выход

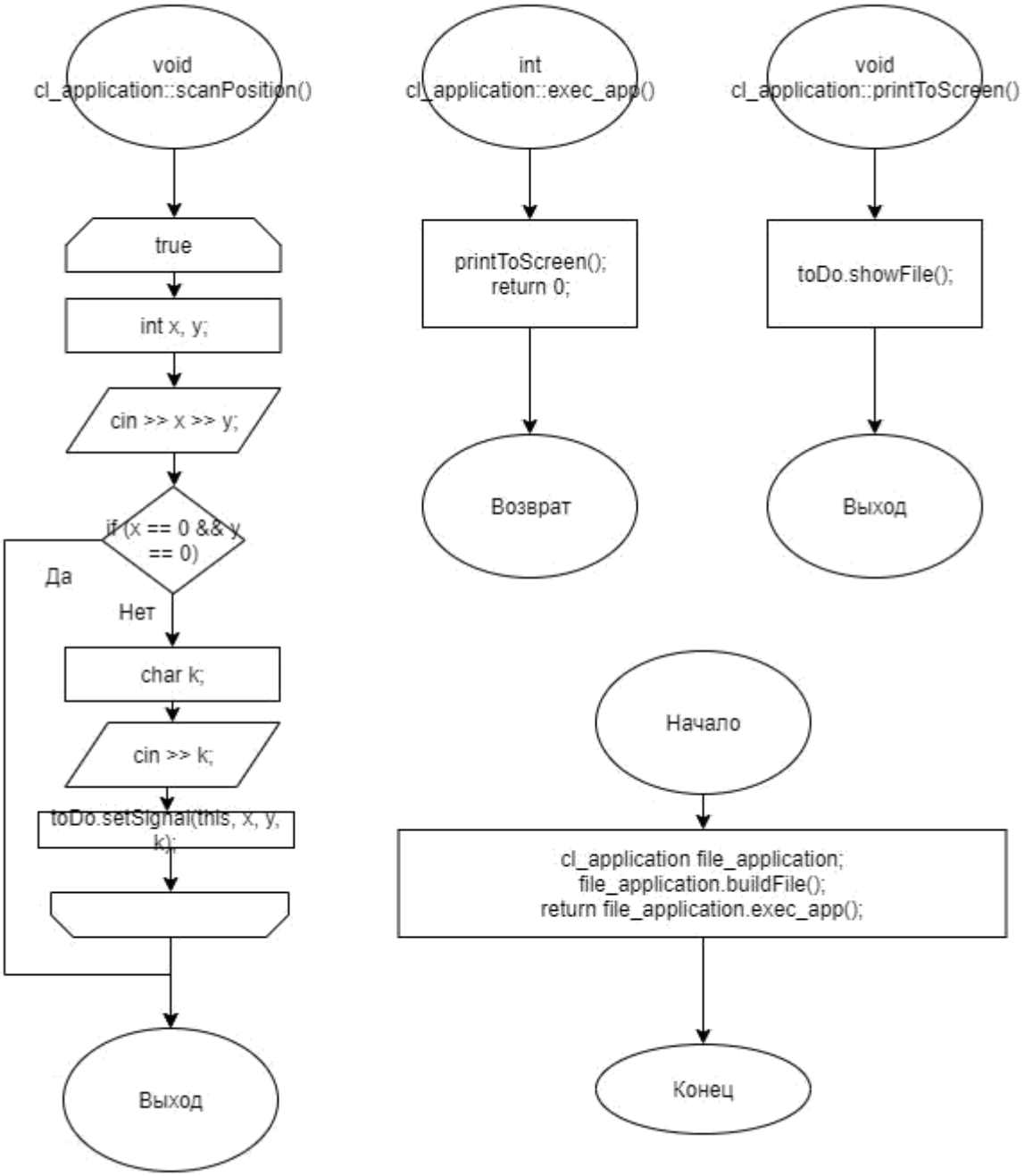

cl_application::cl_application()
string name ="file.txt";
toDo.setNameFile(name);
Выход
void
cl_application::buildFile()
createNewFile();
scanPosition();
Выход


cl_base::handling1
HANDLING1
addDataFile << endl << "Coordinate is wrong (" << x << ", " << y << ")";
addDataFile.close();
addDataFile.close();
Выход


cl_base::handling2
HANDLING2
if (letter)
Да
addDataFile.put(k);
addDataFile.close();
Нет
addDataFile << endl << "Not a letter of the Latin alphabet: " << k;
addDataFile.close();
Выход


void
cl_application::createNewFile()
ofstream
newFile(toDo.getNameFile(),
ios::out)
(!newFile)
Да
return
Нет
newFile.close();
for (int indexR = 1;
indexR <= 10;
indexR++)
Выход
if (indexR !=
1)
Нет
Да
newFile << endl;
for (int indexR = 1;
indexR <= 10;
indexR++)
int number = 8;
newFile << number;


cl_base::showFile()
readFile.open(nameFile, ios::in);
if (!readFile)
Да
return;
Нет
string line;
while(getline(readFile, line))
cout << line;
if (!readFile.eof())
Нет
Да
cout << endl;
readFile.close();
Выход



Код программы

Файл cl_application.cpp

```
#include "cl_application.h"
#include <iostream>
using namespace std;

cl_application::cl_application() {
	string name ="file.txt";
	toDo.setNameFile(name);
}

//Строить фаил
void cl_application::buildFile() {
	createNewFile();
	scanPosition();
}

//Создать новый файл
void cl_application::createNewFile() {
	ofstream newFile(toDo.getNameFile(), ios::out); if (!newFile) {
		return;
	}
	for (int indexR = 1; indexR <= 10; indexR++){ if (indexR != 1){
				newFile << endl;
		}
		for (int indexC = 1; indexC <= 10; indexC++){ int number = 8;
				newFile << number;
		}
	}
	newFile.close();
}

//Обрабатывать координаты
void cl_application::scanPosition() {
	while(true){
		int x, y;
		cin >> x >> y;
		if (x == 0 && y == 0) {
				break;
		}
		else {
				char k;
				cin >>k;
				toDo.setSignal(this, x, y, k);
		}
	}
}
//Применять
int cl_application::exec_app(){
	printToScreen();
	return 0;
}
```

```
//Показать фаил
void cl_application::printToScreen() {
	toDo.showFile();
}
```

Файл cl_application.h

```
#ifndef CL_APPLICATION_H
#define CL_APPLICATION_H

#include "cl_base.h"

class cl_application : public cl_base {

public:
	cl_application();
	//Строить фаил
	void buildFile();
	//Создать новый файл
	void createNewFile();
	//Обрабатывать координаты
	void scanPosition();
	//Применять
	int exec_app();
private:
	cl_base toDo;
	//Показать фаил
	void printToScreen();
};


#endif //!CL_APPLICATION_H
```

Файл cl_base.cpp

```
#include "cl_base.h"

#include <iostream>
using namespace std;
#define SIGNALING (cl_base* p_parent, int x, int y, char k)
#define HANDLING1 (cl_base* p_parent, int x, int y)
#define HANDLING2 (cl_base* p_parent, char k, bool letter)

	//установить имя для файла
void cl_base::setNameFile(string nameFile){ this->nameFile =
	nameFile;
}
	//получить имя файла
```

```
string cl_base::getNameFile() {
	return this->nameFile;
}
	//передать координаты x, y,символов k и обрабатывать события, которые происходят
void cl_base::setSignal SIGNALING {
	if (!((1 <= x && x <= 10)&&(1 <= y && y <= 10))){
		//Открыть файл и дополнительная запись в конце файла
		addDataFile.open(nameFile, ios::app);
		handling1(this, x, y);
	}
	else {
		if (!(('a' <= k && k <= 'z')||('A' <= k && k <= 'Z'))) { //Открыть файл и
			дополнительная запись в конце файла addDataFile.open(nameFile,
			ios::app); handling2(this, k, false);
		}
		else {
			//Открыть файл и открыть для операций ввода
			addDataFile.open(nameFile, ios::in); //Координатый расчет
			int position = (x-1)*11 + y - 1;
			//Переместить курсор к координатам
			addDataFile.seekp(position, ios::beg);
			handling2(this, k, true);
		}
	}
}

//Если координаты неверны
void cl_base::handling1 HANDLING1{
					addDataFile << endl << "Coordinate is wrong ( " << x << ", " << y << "
)";
	addDataFile.close();
}

//Записи символа в файле в установленную позицию void
cl_base::handling2 HANDLING2{
	//Если координаты верны и k является символом if (letter) {
		addDataFile.put(k);
		addDataFile.close();
	}
	//Если k не является символом
	else {
		addDataFile << endl << "Not a letter of the Latin alphabet: "
<< k;
		addDataFile.close();
	}
}

//Вывода результата из файла на консоль
void cl_base::showFile(){
	//Открыть файл и открыть для операций ввода
	readFile.open(nameFile, ios::in);
	if (!readFile){
```

```
                return;
        }
        string line;
        while(getline(readFile, line)) {
                cout << line;
                if (!readFile.eof()) {
                        cout << endl;
                }
        }
        readFile.close();
}

//Из Лаб 3.4
cl_base::cl_base(cl_base* p_parent){
        set_object_name("cl_base");
        if (p_parent) {
                this->p_parent = p_parent;
                p_parent->add_child(this);
        } else {
                this->p_parent = 0;
        }
}

cl_base::cl_base(cl_base* p_parent, bool infoSender, bool storageMess){ if (infoSender) {
                set_object_name("cl_base");
                if (p_parent) {
                        this->p_parent = p_parent;
                        p_parent->addChildInfoSender(this);
                } else {
                        this->p_parent = 0;
                }
        }
        if (storageMess) {
                set_object_name("cl_base");
                if (p_parent) {
                        this->p_parent = p_parent;
                        p_parent->addChildStorageMess(this);
                } else {
                        this->p_parent = 0;
                }
        }
}

cl_base::cl_base(cl_base* p_parent, bool id){
                set_object_name("cl_base"); if (p_parent) {
                        this->p_parent = p_parent;
                        p_parent->addChildInfoSender(this);
                } else {
                        this->p_parent = 0;
                }
}
        void cl_base::set_object_name(string object_name){ this->object_name =
                object_name;
        }
        string cl_base::get_object_name(cl_base* p_parent){ return p_parent-
                >object_name;
        }
        void cl_base::set_parent(cl_base* p_parent){
```

```
		if(p_parent) {
			this->p_parent = p_parent;
			p_parent->add_child(this);
		}
}
void cl_base::add_child(cl_base* p_child) {
		children.push_back(p_child);
}
cl_base* cl_base::cl_base::get_child(string object_name) { if (children.size()== 0)
		return 0;
		it_child = children.begin();
		while(it_child != children.end()) {
			if (get_object_name((*it_child)) == object_name) { return (*it_child);
			}
			it_child++;
		}
		return 0;
}
void cl_base::set_state(int c_state) {
		this->c_state = c_state;
}
int cl_base::get_state(cl_base* p_parent) { return p_parent-
		>c_state;
}
void cl_base::setConnect(int id, string nameSender, string nameReceiver){
		setID(id);
		setNameSender(nameSender);
		setNameReceiver(nameReceiver);
}
void cl_base::deleteConnect(){}
void cl_base::signaling(string message, string nameSender){
		setNameSender(nameSender);
		setMessageText(message);
}
void cl_base::setNameSender(string nameSender){ this->nameSender =
		nameSender;
}
string cl_base::getNameSender(cl_base* p_parent){ return p_parent-
		>nameSender;
}
void cl_base::setNameReceiver(string nameReceiver) { this->nameReceiver
		= nameReceiver;
}
string cl_base::getNameReceiver(cl_base* p_parent){ return p_parent-
		>nameReceiver;
}
void cl_base::setMessageText(string messageText){ this->messageText =
		messageText;
}
string cl_base::getMessageText(cl_base* p_parent){ return p_parent-
		>messageText;
}
void cl_base::setID(int id){
		this->id = id;
```

```
}
int cl_base::getID(cl_base* p_parent){
	return p_parent->id;
}

void cl_base::addChildInfoSender(cl_base* p_child){
	infoSender.push_back(p_child);
}
void cl_base::addChildStorageMess(cl_base* p_child){
	storageMess.push_back(p_child);
}
```

Файл cl_base.h

```
#ifndef CL_BASE_H
#define CL_BASE_H

#include <string>
#include <fstream>
#include <vector>

#define SIGNALING (cl_base* p_parent, int x, int y, char k)
#define HANDLING1 (cl_base* p_parent, int x, int y)
#define HANDLING2 (cl_base* p_parent, char k, bool letter)

using namespace std;
class cl_base {
private:
	string nameFile; ofstream		//Имя файла
	addDataFile; ifstream readFile;		//Обявить файл для чтения
						//Обявить файл для записи


	//Из лаб 3.4
	string object_name;
	cl_base* p_parent;
	int c_state;
	int id;
	string nameSender;
	string nameReceiver;
	string messageText;

public:
	//установить имя для файла
	void setNameFile(string nameFile);
	//получить имя файла
	string getNameFile();
	//передать координаты x, y,символов k и обрабатывать события, которые происходят
	void setSignal SIGNALING;
	//Если координаты неверны
	void handling1 HANDLING1;
```

```
//Записи символа в файле в установленную позицию void handling2
HANDLING2;
//Вывода результата из файла на консоль void
showFile();


//Из Лаб 3.4
cl_base(cl_base* p_parent = 0);
cl_base(cl_base* p_parent, bool infoSender, bool storageMess); cl_base(cl_base* p_parent,
bool id);
void set_object_name(string object_name); string
get_object_name(cl_base* p_parent); void set_parent(cl_base*
p_parent);
void add_child(cl_base* p_child);
cl_base* get_child(string object_name); void set_state(int
c_state);
int get_state(cl_base* p_parent);
void setConnect(int id, string nameSender, string nameReceiver); void deleteConnect();
void signaling(string message, string nameSender); void
setNameSender(string nameSender);
string getNameSender(cl_base* p_parent); void
setNameReceiver(string nameReceiver); string
getNameReceiver(cl_base* p_parent); void
setMessageText(string messageText); string
getMessageText(cl_base* p_parent); void setID(int id);
int getID(cl_base* p_parent);

void addChildInfoSender(cl_base* p_child); void
addChildStorageMess(cl_base* p_child);

vector <cl_base*> infoSender;
vector <cl_base*> :: iterator it_iS;

vector <cl_base*> storageMess;
vector <cl_base*> :: iterator it_sM;

vector <cl_base*> children;
vector <cl_base*> :: iterator it_child;

string text_finish = "endtree";
//

};


#endif //!CL_BASE_H
```

Файл main.cpp

```
#include <iostream>
using namespace std;
#include "cl_application.h"


int main()
{
	cl_application file_application;
	//Строить фаил
	file_application.buildFile();
	//Применять
	return file_application.exec_app();
}
```

Тестирование

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
1 1 h 6 6 g 1 6 a 6 1 b 9 5 C 5 9 D -1 3 k -6 3 g 3 4) 1 10 A 10 10B00	h8888a888A 8888888888 8888888888 8888888888 88888888D8 b8888g8888 8888888888 8888888888 8888C88888 888888888B Coordinate is wrong (-1, 3) Coordinate is wrong (-6, 3) Not a letter of the Latin alphabet:)	h8888a888A 8888888888 8888888888 8888888888 88888888D8 b8888g8888 8888888888 8888888888 8888C88888 888888888B Coordinate is wrong (-1, 3) Coordinate is wrong (-6, 3) Not a letter of the Latin alphabet:)