Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Системное программирование (СП)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

Игровое приложение «Сапер»

БГУИР КП 1-40 01 01  003  ПЗ

Студент: гр. 151003 Барановский Р.А.

Руководитель: Низовцов Д.В.

Минск 2023

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой ПОИТ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

Лапицкая Н.В. 2023г.

ЗАДАНИЕ

по курсовому проектированию

Студенту *Барановскому Роману Алексеевичу*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Тема работы *Игровое приложение «Сапер»*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Срок сдачи студентом законченной работы\_\_19.12.2023г.\_\_\_\_

3. Исходные данные к работе *Среда программирования Visual studio. Windows API для взаимодействия приложения с операционной системой Windows. Язык программирования c++. Наличие графической реализации интерфейса игры. Возможность игроку выбрать желаемую сложность.*

4. Содержание расчетно-пояснительной записки (перечень вопросов, которые подлежат разработке)

*Введение*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*1 Анализ литературных источников\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2 Постановка задачи\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*3 Разработка программного средства\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*4 Тестирование и проверка работоспособности программного средства\_\_\_\_*

*5 Руководство по использованию программного средства\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Заключение \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Список использованной литературы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

\_*Приложения* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

*Схема алгоритма в формате А1*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. Консультант по курсовой работе Низовцов Д.В.*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

7.Дата выдачи задания *16.09.2023г.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и процентом от общего обьема работы):

*Раздел 1. Введение к 25.09.2023г. – 10 % готовности работы;\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Раздел 2 к 15.10.2023г. – 30% готовности работы;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Раздел 3 к 15.11.2023г. – 60% готовности работы;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Разделы 4, 5, Заключение к 15.12.2023 – 90 % готовности работы;

Оформление пояснительной записки и графического материала к 17.12.2023 – 100 % готовности работы.

Защита курсового проекта с 17.12.2023г. по 20.12.2023г.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

РУКОВОДИТЕЛЬ *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* Низовцов Д.В.

*(подпись)*

Задание принял к исполнению *\_\_\_\_\_\_\_\_\_\_\_\_\_Барановский Р.А. 16.09.2023г.*

*(дата и подпись студента)*

**СОДЕРЖАНИЕ**

[Введение 5](#_Toc153306297)

[1 анализ литературных источников 6](#_Toc153306298)

[1.1 Анализ существующих аналогов 6](#_Toc153306299)

[1.2 Анализ методов и способов разработки 10](#_Toc153306300)

[2 Постановка задачи 11](#_Toc153306301)

[2.1 Назначение разработки 11](#_Toc153306302)

[2.2 Перечень функциональных требований 11](#_Toc153306303)

[2.3 Структура программы 12](#_Toc153306304)

[2.4 Состав и параметры технических и программных средств 12](#_Toc153306305)

[3 Разработка программного средства 13](#_Toc153306306)

[3.1 Описание алгоритмов решения задачи 13](#_Toc153306307)

[3.2 Структура типов 15](#_Toc153306308)

[3.3 Схема алгоритмов решения задач по ГОСТ 19.701-90 21](#_Toc153306309)

[4 тестирование и проверка работоспособности программного средства 27](#_Toc153306310)

[4.1 Взаимодействие с меню 27](#_Toc153306311)

[4.2 Игровой процесс 29](#_Toc153306312)

[5 Руководство по использованию программного средства 34](#_Toc153306313)

[5.1 Меню……………………………………………………………………….34](#_Toc153306314)

[5.2 Игра………………………………………………………………………...34](#_Toc153306315)

[5.3 Новая игра 34](#_Toc153306316)

[5.4 Завершение игры 34](#_Toc153306317)

[Заключение 35](#_Toc153306318)

[Список использованной литературы 36](#_Toc153306319)

[Приложение А 37](#_Toc153306320)

Введение

Данная работа посвящена созданию компьютерной игры «Сапер». Данная игра была создана в компании «Microsoft» Робертом Доннером и Куртом Джонсоном в 1992 году для операционной системы «Windows» с целью помочь пользователям отточить навыки использования компьютерной мыши. Поиск заминированных клеток призван был научить новичков свободно владеть как правой, так и левой кнопками мыши, и показать им, как соотносятся скорость и точность движения мышки. Принцип игры состоит в следующем: плоское или объёмное игровое поле разделено на смежные ячейки (квадраты, шестиугольники, кубы и т. п.), некоторые из которых «заминированы»; количество «заминированных» ячеек известно. Целью игры является открытие всех ячеек, не содержащих мины. Игрок открывает ячейки, стараясь не открыть ячейку с миной. Открыв ячейку с миной, он проигрывает. Для того, чтобы игрок открывал ячейки не «вслепую», используется следующий принцип: если под открытой ячейкой мины нет, то в ней появляется число, показывающее, сколько ячеек, соседствующих с только что открытой, «заминировано» (в каждом варианте игры соседство определяется по-своему); используя эти числа, игрок пытается рассчитать расположение мин, однако иногда даже в середине и в конце игры некоторые ячейки всё же приходится открывать наугад. Если под соседними ячейками тоже нет мин, то открывается некоторая «не заминированная» область до ячеек, в которых есть цифры. «Заминированные» ячейки игрок может пометить, чтобы случайно не открыть их. «Сапер» завоевал огромное популярность, существует множество различных версий игры и даже проводятся соревнования по скорости «разминирования». Особо примечательным рекордом является достижение Атто Юкисиро (Япония), который в 2016 году прошел поле «Сапера» размером 64х48 на ультрасложном уровне с 777 бомбами, на что у него ушло 12,5 часов, попытки пройти это поле он предпринимал в течение 10 лет.

# анализ литературных источников

## Анализ существующих аналогов

Современные разновидности «Сапера» отличаются огромным разнообразием. Они кардинально отличаются как графически, так и реализицией. Однако стандартные правила остаются везде практически без изменений.

### «Сапер» от google

«Сапер» от google – популярная версия игры «сапер» от компании google. Сыграть можно при наличии браузера и доступа к интернету. Игра обладает стандартными правилами, ведет статистику о времени в игре и количестве найденных мин, предоставляет возможность выбрать уровень сложности.

Достоинства игры:

* отличное графическое, музыкальное сопровождение;

В свою очередь к недостаткам можно отнести:

* доступность игры лишь в виде веб-приложения, что не позволяет установить ее на устройство и играть без доступа к интернету;
* отсутствие возможности выбрать желаемый размер поля.



Рисунок 1.1 – «Сапер» от google



Рисунок 1.2 – «Сапер» от google

### «Minesweeper» от Evkar games

«Minesweeper» от Evkar games – игра на мобильные устройства, позволяющая проникнуться духом классического «сапера».

Достоинства игры:

* отличное графическое сопровождение;
* наличие огромного количества различных настроек.

В свою очередь к недостаткам можно отнести:

* присутствие багов;
* большое количество рекламы в бесплатной версии.

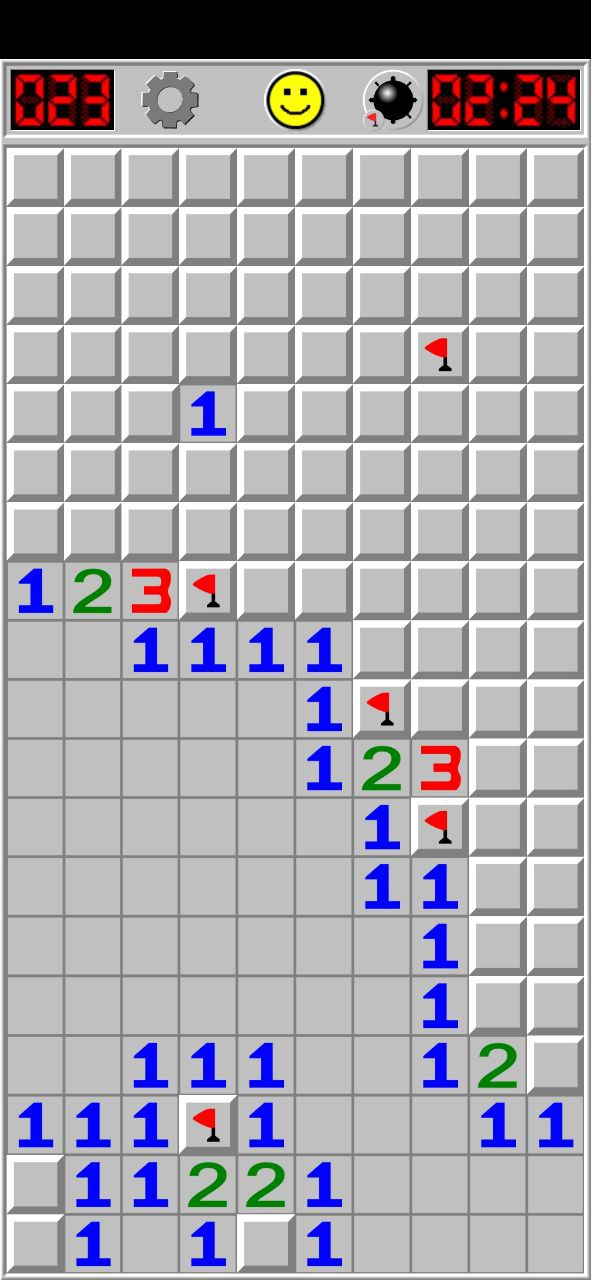


Рисунок 1.3 – «Minesweeper» от Evkar games

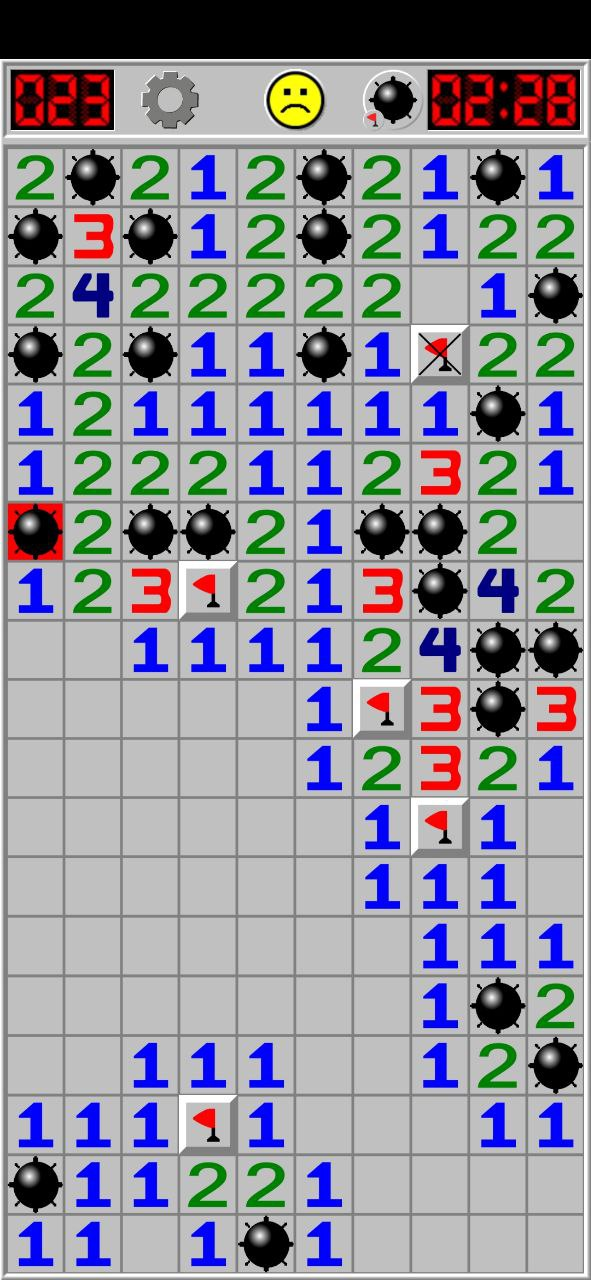


Рисунок 1.4 – «Minesweeper» от Evkar games

### «Microsoft Minesweeper»

«Microsoft Minesweeper» – «сапер» от компании Microsoft, разработчики которой придумали правила данной игры и впервые разработали ее.

Достоинства игры:

* присутствие различных разнообразных режимов игры;
* присутствие ежедневных заданий;
* отличное графическое, музыкальное сопровождение.

В свою очередь к недостаткам можно отнести:

* возрастное ограничение 16+ с пометкой «насилие».
* большое количество рекламы в бесплатной версии

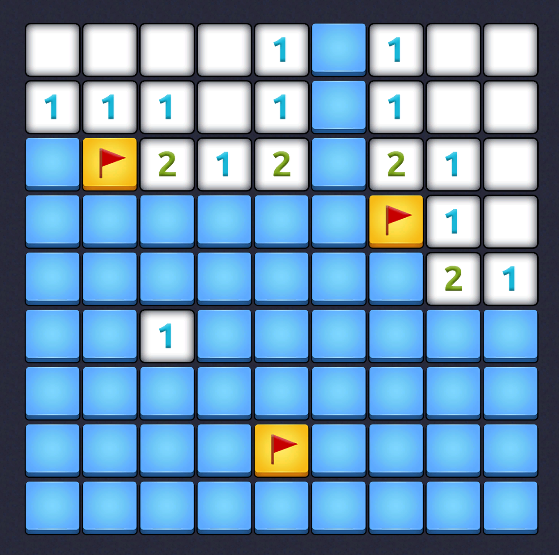


Рисунок 1.5 – «Microsoft Minesweeper»



Рисунок 1.6 – «Microsoft Minesweeper»

## Анализ методов и способов разработки

### Используемые библиотеки и технологии

Предполагается, что разрабатываемая игра будет обладать графикой с возможностью взаимодействия с пользователем. Для этих целей будет использован Windows API.

Windows API – общее наименование набора базовых функций интерфейсов программирования приложений операционных систем семейств Microsoft Windows корпорации Microsoft. Предоставляет прямой способ взаимодействия приложений пользователя с операционной системой Windows.

### Используемые структуры данных

В данном проекте в качестве основной структуры данных будет использован vector.

Структура vector – структура данных в языке c++, предназначенная для хранения множества значений. Данная структура данных основана на массиве и является его усовершенствованной версией, так как динамична. Использование данной структуры облегчает работу за счет удаления необходимости вручную изменять размер массива, удалять элементы из него, вставлять элементы в массив. Структура vector представлена на рисунке 1.7.[1]

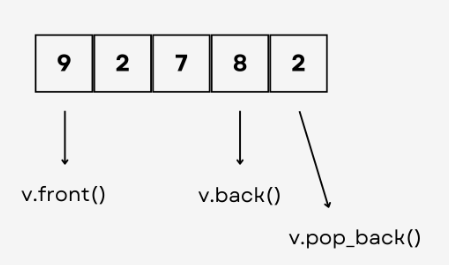


Рисунок 1.7 – vector

# Постановка задачи

## Назначение разработки

Назначением проектирования является разработка игры «Сапер». На основании произведенного обзора существующих аналогов, выявленных преимуществ и недостатков данных игр, сделан вывод, что для решения поставленной цели необходимо выполнить следующие задачи:

* проектирование архитектуры игры;
* проектирование графического сопровождения;
* разработка алгоритмов загрузки изображений из внешних файлов;
* разработка алгоритмов взаимодействия игрока с полем;
* разработка алгоритмов взаимодействия объектов между собой;
* разработка алгоритмов отображения сопровождающей информации;
* разработка меню для настроек игры;
* тестирование приложения.

## Перечень функциональных требований

Целью разработки игры «Сапер» является объединение основных достоинств рассмотренных существующих аналогов, а также компенсация недостатков этих игр. В результате разработки необходимо предоставить реализацию следующих функций:

* загрузка изображений из внешних файлов;
* настройка пользователем сложности и размера поля;
* взаимодействие между игроком и полем;
* отображение количества времени текущей игры;
* отображение количества оставшихся мин;
* просмотр информации о программе;
* завершение игры при попадании на мину.

## Структура программы

Основные модули программы:

* Program.cpp (Program.h) – модуль, обеспечивающий запуск игры и обработку сообщений;
* Type.cpp (Type.h) – модуль, содержащий информацию о возможных типах ячеек в игре(закрытая ячейка, бомба, флажок и т.д.);
* Cell.cpp (Cell.h) – модуль, обеспечивающий логику взаимодействия с ячейкой;
* Field.cpp (Field.h) – модуль, обеспечивающий логику взаимодействия с игровым полем;
* DrawableField.cpp (DrawableField.h) – модуль, обеспечивающий логику отрисовки игрового поля на экране пользователя.

## Состав и параметры технических и программных средств

Игра «Сапер» должна функционировать на персональных компьютерах со следующими характеристиками:

* Операционная система: Windows 10;
* RAM: 2 GB;
* Пространство на диске: 1 GB;
* Процессор: минимальное требование - Pentium 2 266 МГц;
* Монитор;
* Мышь;
* Клавиатура.

В данном разделе указаны минимальные технические требования для запуска игры. Для эксплуатации в реальных условиях могут потребоваться более мощные технические средства. Разработанная игра должна корректно функционировать на более мощном оборудовании.

# Разработка программного средства

## Описание алгоритмов решения задачи

Таблица 1 – Описание алгоритмов решения задачи

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №  п.п. | Наименование алгоритма | Назначение алгоритма | Формальные параметры | Предлагае-мый тип реализации |
| 1 | Program::  wWinMain(  \_In\_ HINSTANCE hInstance, \_In\_opt\_ HINSTANCE hPrevInstance, \_In\_ LPWSTR lpCmdLine, \_In\_ int nCmdShow) | Служит отправной точкой выполнения программы.  Создает главное окно программы. Запускает цикл обработки сообщений[2] | hInstance – дескриптор для текущего экземпляра приложения.  hPrevInstance – дескриптор предыдущего экземпляра приложения.  lpCmdLine – командная строка приложения, за исключением имени программы.  nCmdShow – управляет тем, как должно отображаться окно | Функция.  Возвраща-емое значение –  выходное значение из параметра сообщения wParam или 0 в случае заверше-  ния до входа в цикл обработки сообщений |
| 2 | Program::  WndProc(  HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam) | Обрабатывает сообщения, отправляемые в главное окно приложения[3] | hWnd – дескриптор окна  message – сообщение  wParam – дополнитель- ные сведения о сообщении  lParam – дополнитель- | Функция.  Возвраща-емое значение – результат обработки сообщения |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | ные сведения о сообщении |  |
| 3 | Field::  reveal(int x,  int y) | Открывает одну или несколько ячеек на игровом поле | x – координата ячейки на оси абсцисс.  y – координата ячейки на оси ординат | Процедура |
| 4 | Field::  revealAll() | Открывает все ячейки на игровом поле | Формальные параметры отсутствуют | Процедура |
| 5 | Cell::  changeFlag() | Меняет состояние флага на ячейке | Формальные параметры отсутствуют | Функция.  Возвраща-емое значение – тип клетки в результате смены состояния флага |
| 6 | Cell::open() | Открывает ячейку | Формальные параметры отсутствуют | Функция.  Возвраща-емое значение – true, если ячейка была открыта, false в противном случае |
| 7 | Field::  moveAway-  Bomb(int x,  int y) | Передвигает мину из ячейки с координатами (x, y) в случайную свободную ячейку на поле | x – координата ячейки на оси абсцисс.  y – координата ячейки на оси ординат | Процедура |

## Структура типов

Таблица 2 – Структура типов программы

|  |  |  |
| --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение |
| Type | class Type  {  private:  static std::map<Types, std::string> typeImagePath;  Types m\_type;  std::string m\_imagePath;  public:  enum class Types;  Type(Types type);  operator std::string() const;  operator Types() const;  bool operator == (const Type& type);  bool operator == (const Types& type);  bool operator != (const Type& type);  bool operator != (const Types& type);  Type& operator = (const Types& type);  static std::map<Types, std::string> getTypesAndPaths();  }; | Класс, содержащий информацию о типе ячейки;  Поля данных:  typeImagePath – ассоциативный массив с ключами-типами и значениями-логическими путями типов;[4]  m\_type – тип ячейки;  m\_imagePath – логический путь к изображению типа;  Types – перечисление, содержащее все возможные типы ячейки;  Методы класса:  Type(Types type) – конструктор класса;  operator std::string() – оператор приведения к строке(логическому пути к изображению типа);[5]  operator Types() – оператор приведения к типу-перечислению;  operator == (const Type& type) – оператор сравнения с типом;  operator == (const Types& type) – оператор сравнения с типом-перечислением; |

Продолжение таблицы 2 – Структура типов программы

|  |  |  |
| --- | --- | --- |
|  |  | operator != (const Type& type) – оператор сравнения с типом;  operator != (const Types& type) – оператор сравнения с типом-перечислением;  operator = (const Types& type) – оператор присваивания с типом-перечислением;  getTypesAndPaths() – возвращает typeImagePath; |
| Cell | class Cell  {  private:  Type m\_inner;  Type m\_cover;  bool m\_isOpened;  public:  Cell();  Type getInner();  void setInner(Type inner);  Type getCover();  void setCover(Type cover);  bool isOpened();  Type getType();  Type changeFlag();  bool open();  }; | Класс, представляющий собой ячейку;  Поля данных:  m\_inner – внутренняя часть ячейки;  m\_cover – внешняя часть ячейки;  m\_isOpened – true, если ячейка открыта, false в противном случае;  Методы класса:  Cell() – конструктор класса;  getInner() – возвращает m\_inner;  setInner(Type inner) – устанавливает m\_inner;  getCover() – возвращает m\_cover;  setCover(Type cover) – устанавливает m\_cover;  isOpened() – возвращает m\_isOpened;  getType() – возвращает тип ячейки |

Продолжение таблицы 2 – Структура типов программы

|  |  |  |
| --- | --- | --- |
|  |  | в зависимости от ее состояния.  changeFlag() – устанавливает или убирает флаг на ячейке  open() – открывает ячейку |
| Field | class Field  {  private:  struct Coordinates;  static std::random\_device rd;  static std::mt19937 mersenne;  std::vector<std::vector  <Cell>> m\_field;  bool m\_isDeminingStarted;  bool m\_isGameOver;  int m\_numMines;  int m\_cellsRevealed;  int m\_width;  int m\_height;  bool outBounds(int x, int y);  public:  Field(int width, int height, int numMines);  bool isGameEnded();  bool isDemined();  bool isDeminingStarted();  int generateInt(int min, int max);  int calcBombsNear(int x, int y);  void countBombsNearEachCell();  void moveAwayBomb(int x, int y); | Класс, представляющий собой игровое поле;  Поля данных:  Coordinates – структура с координатами x, y;  rd – объект для инициализации mersenne;  mersenne – объект для генерации случайных целочисленных значений;  m\_field – вектор ячеек, представляющий поле;  m\_isDeminingStarted – true, если началось разминирование, false в противном случае;  m\_isGameOver – true, если игра завершилась, иначе false;  m\_numMines – количество мин;  m\_cellsRevealed – количество открытых ячеек;  m\_width – ширина поля;  m\_height – высота поля;  Методы класса: |

Продолжение таблицы 2 – Структура типов программы

|  |  |  |
| --- | --- | --- |
|  | void reveal(int x, int y);  void revealAll();  void generate(int width,  int height, int numMines);  std::vector<Cell>& operator[](const int index);  int getHeight() const;  int getWidth() const;  int getNumMines() const;  }; | outBounds(int x, int y) – возвращает true, если x или y за границами поля, false в противном случае;  Field(int width, int height, int numMines) – конструктор класса;  isGameEnded() – возвращает true, если игра завершилась, false в противном случае;  isDemined() – возвращает true, если всё поле разминировано(игрок победил), false в противном случае;  isDeminingStarted() – возвращает true, если разминирование началось(открыта хотя бы 1 клетка), false в противном случае;  generateInt(int min, int max) – генерирует случайное целое число от min до max включительно;  calcBombsNear(int x,  int y) – подсчитывает количество бомб рядом с ячейкой с координатами x, y;  countBombsNearEach-  Cell() – подсчитвает количество бомб рядом с каждой свободной клеткой.  moveAwayBomb(int x, int y) – убирает бомбу с ячейки с координатами x, y и переносит в |

Продолжение таблицы 2 – Структура типов программы

|  |  |  |
| --- | --- | --- |
|  |  | случайную свободную ячейку на поле;  reveal(int x, int y) – раскрывает ячейки, начиная с ячейки с координатами x, y в соответствии с правилами;  revealAll() – раскрывает все ячейки на поле;  generate(int width,  int height, int numMines)  – генерирует случайное поле в соответствии с переданными размерами и количеством мин;  operator[](const int index) – оператор индексации элементов массива для получения ячеек поля.  getHeight() – получение высоты поля;  getWidth() – получение ширины поля;  getNumMines() – получение количества мин на поле; |
| DrawableField | class DrawableField  {  private:  HWND& m\_hWnd;  HINSTANCE& m\_hInst;  Field& m\_field;  std::vector<HWND> m\_buttons; | Класс, служащий для отрисовки поля:  Поля данных:  m\_hWnd – дескриптор окна, в котором происходит отрисовка;  m\_hInst – сущность окна, в котором происходит отрисовка;  m\_field – ссылка на поле для отрисовки; |

Продолжение таблицы 2 – Структура типов программы

|  |  |  |
| --- | --- | --- |
|  | std::vector<HBITMAP> m\_images;  public:  DrawableField(Field& field, HWND& hWnd, HINSTANCE& hInst);  void update(bool isRecreateButtons, SUBCLASSPROC fieldActions);  }; | m\_buttons – кнопки, представляющие ячейки поля;  m\_images – изображения на кнопках(необходимо для избежания обновления изображений на кнопках, которые не изменились);[6]    Методы класса:  DrawableField(Field& field, HWND& hWnd, HINSTANCE& hInst) – конструктор класса;  update(bool isRecreateButtons, SUBCLASSPROC fieldActions) – перерисовывает поле с пересозданием кнопок или без; |

## Схема алгоритмов решения задач по ГОСТ 19.701-90

### Схема алгоритма Field::reveal

Схема алгоритма открытия одной или нескольких ячеек на игровом поле в соотетствии с правилами.

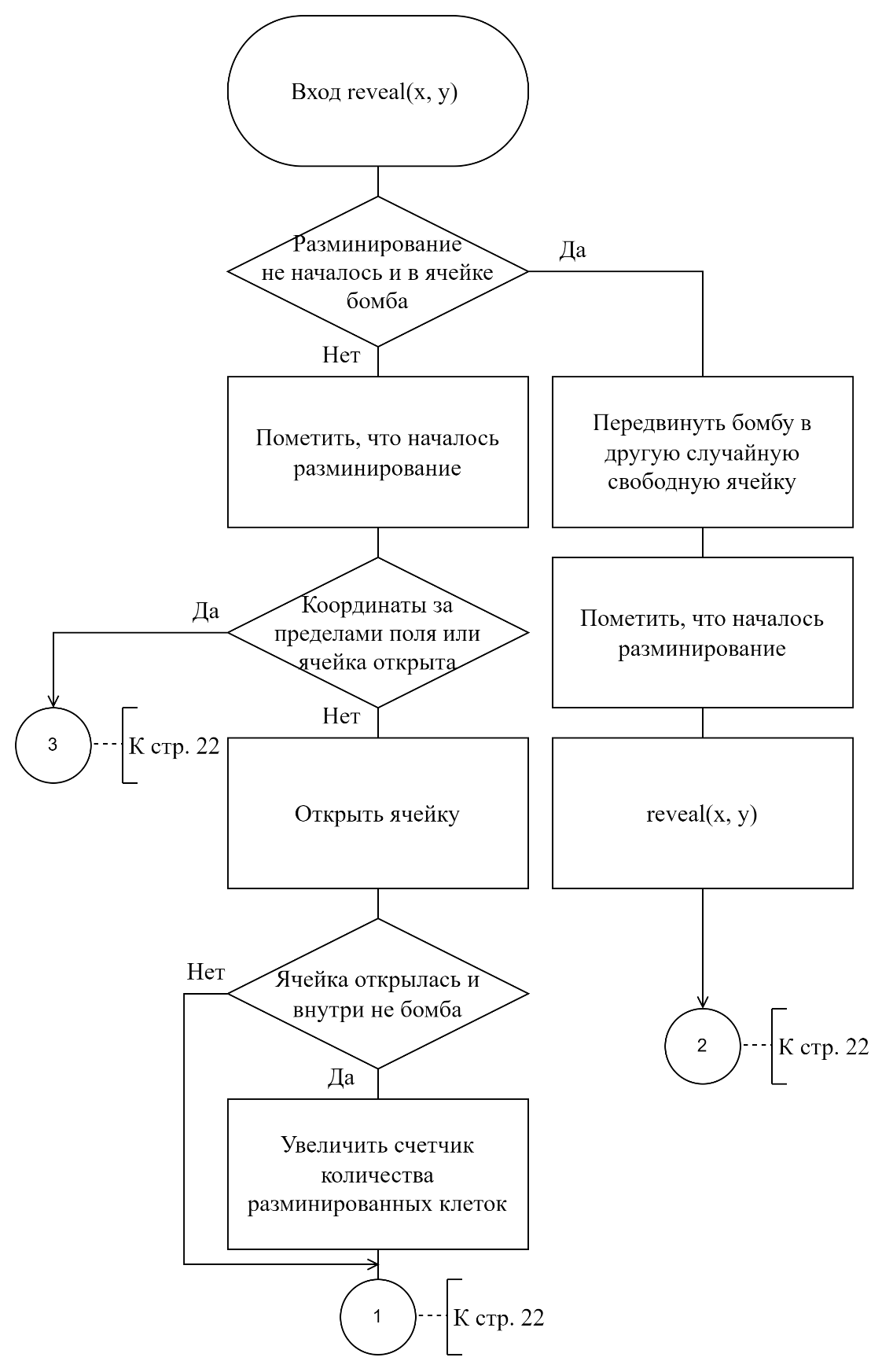
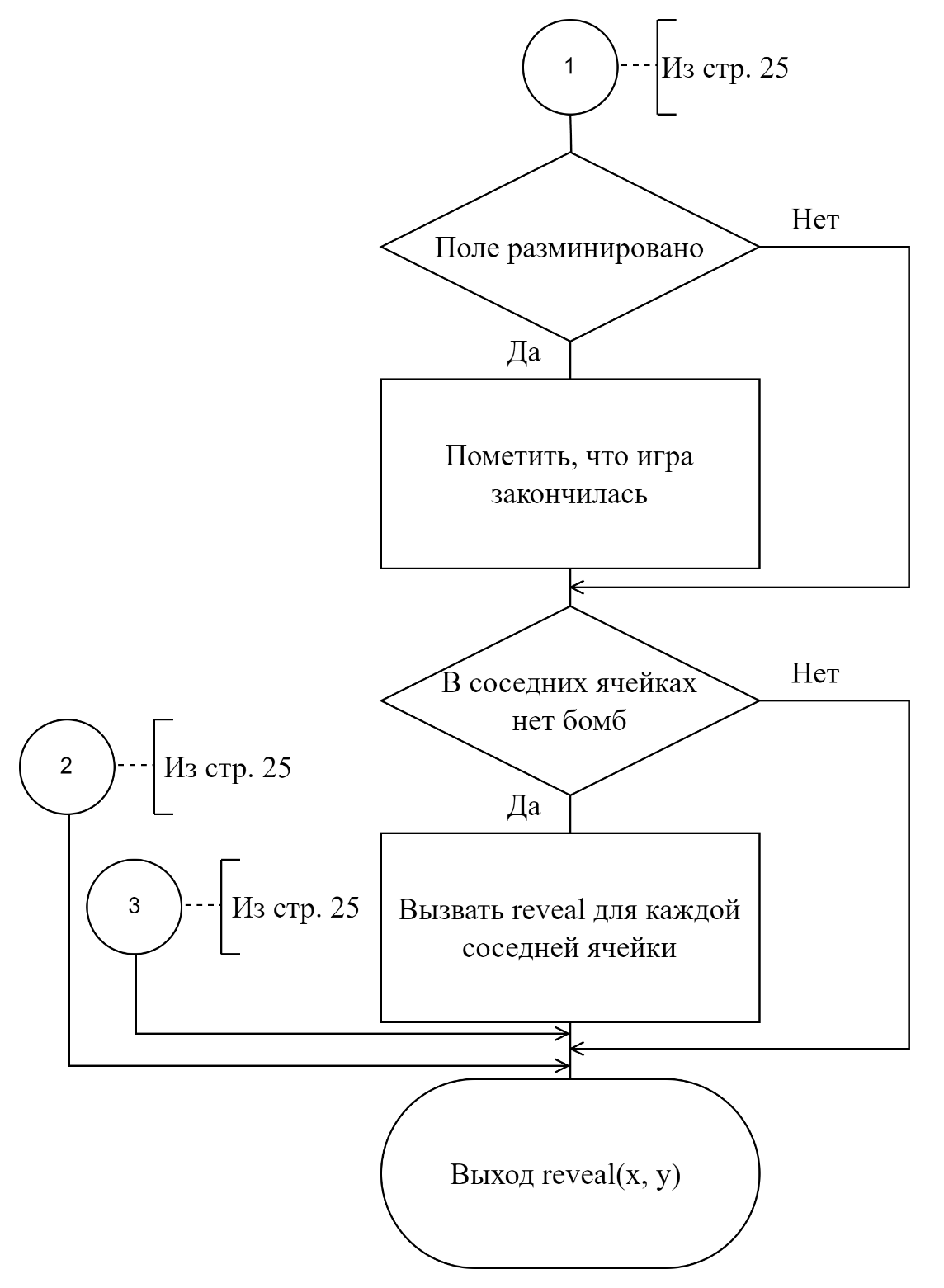


Рисунок 3.1 – Схема алгоритма reveal (часть 1)



21

21

21

Рисунок 3.1 – Схема алгоритма reveal (часть 2)

### Схема алгоритма Field::revealAll()

Схема алгоритма открытия всех ячеек на поле.

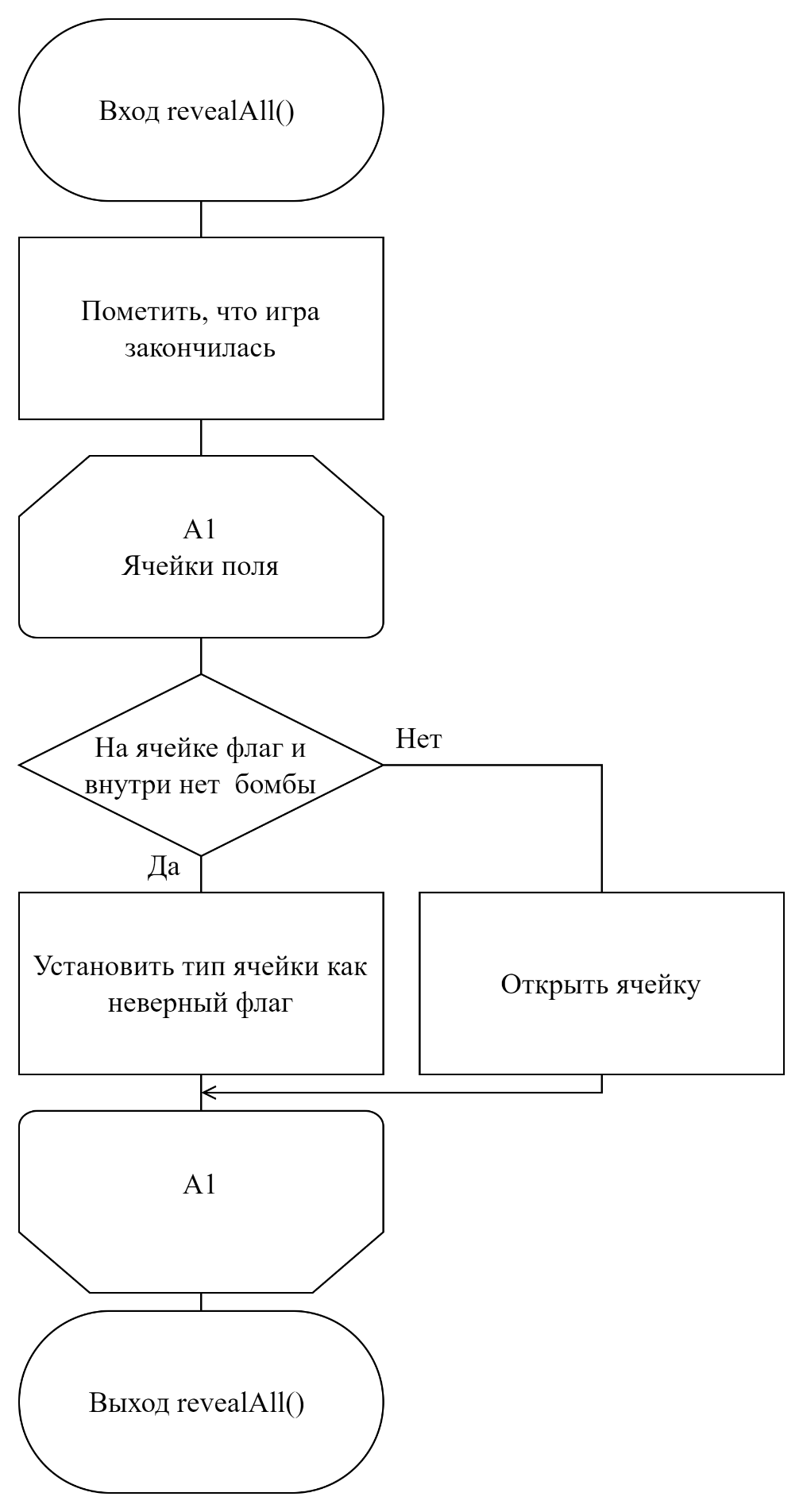


Рисунок 3.2 – Схема алгоритма revealAll

### Схема алгоритма Cell::changeFlag

Схема алгоритма установления или удаления флага с ячейки.

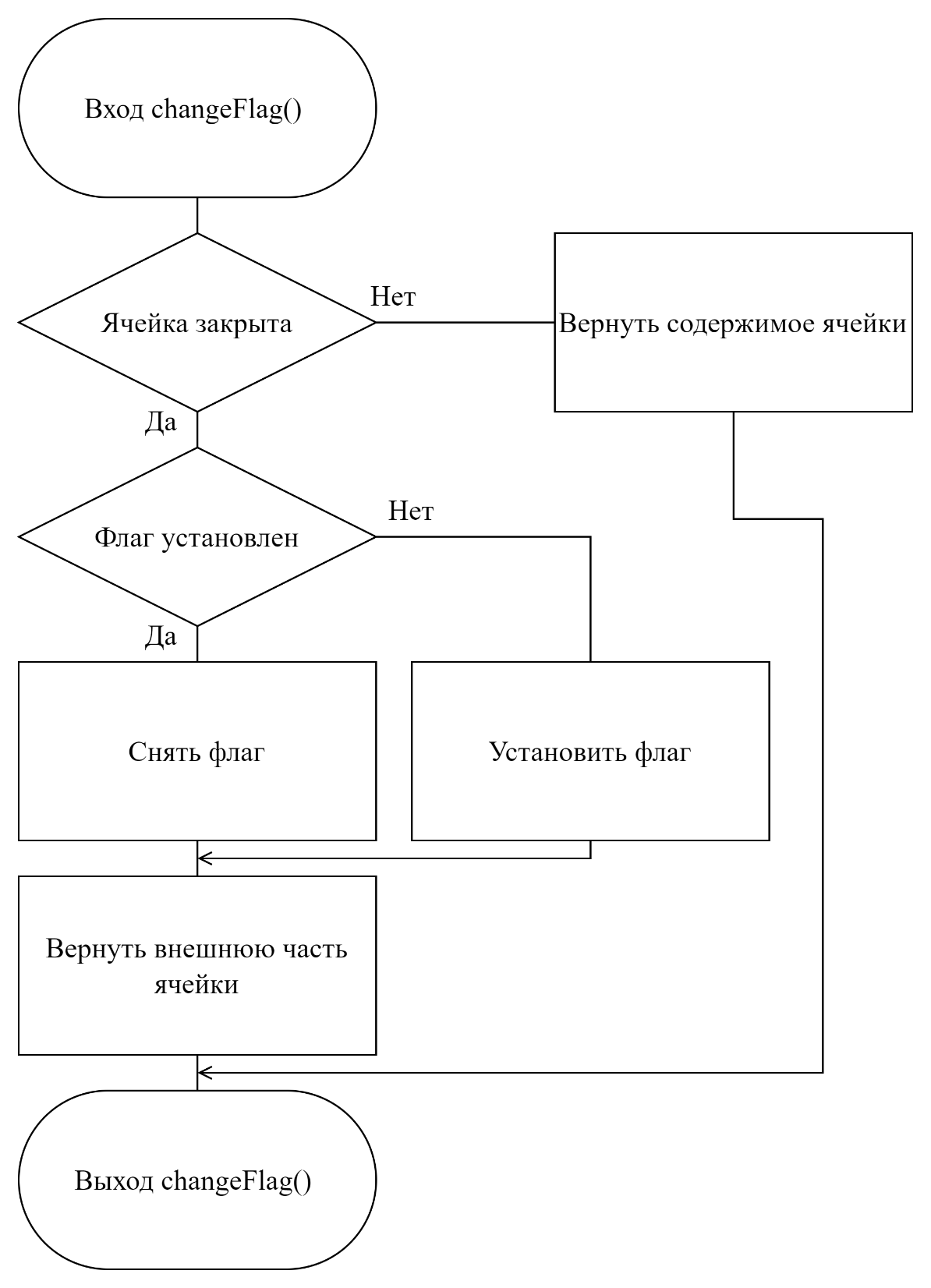


Рисунок 3.3 – Схема алгоритма changeFlag

### Схема алгоритма Cell::open

Схема алгоритма открытия ячейки.

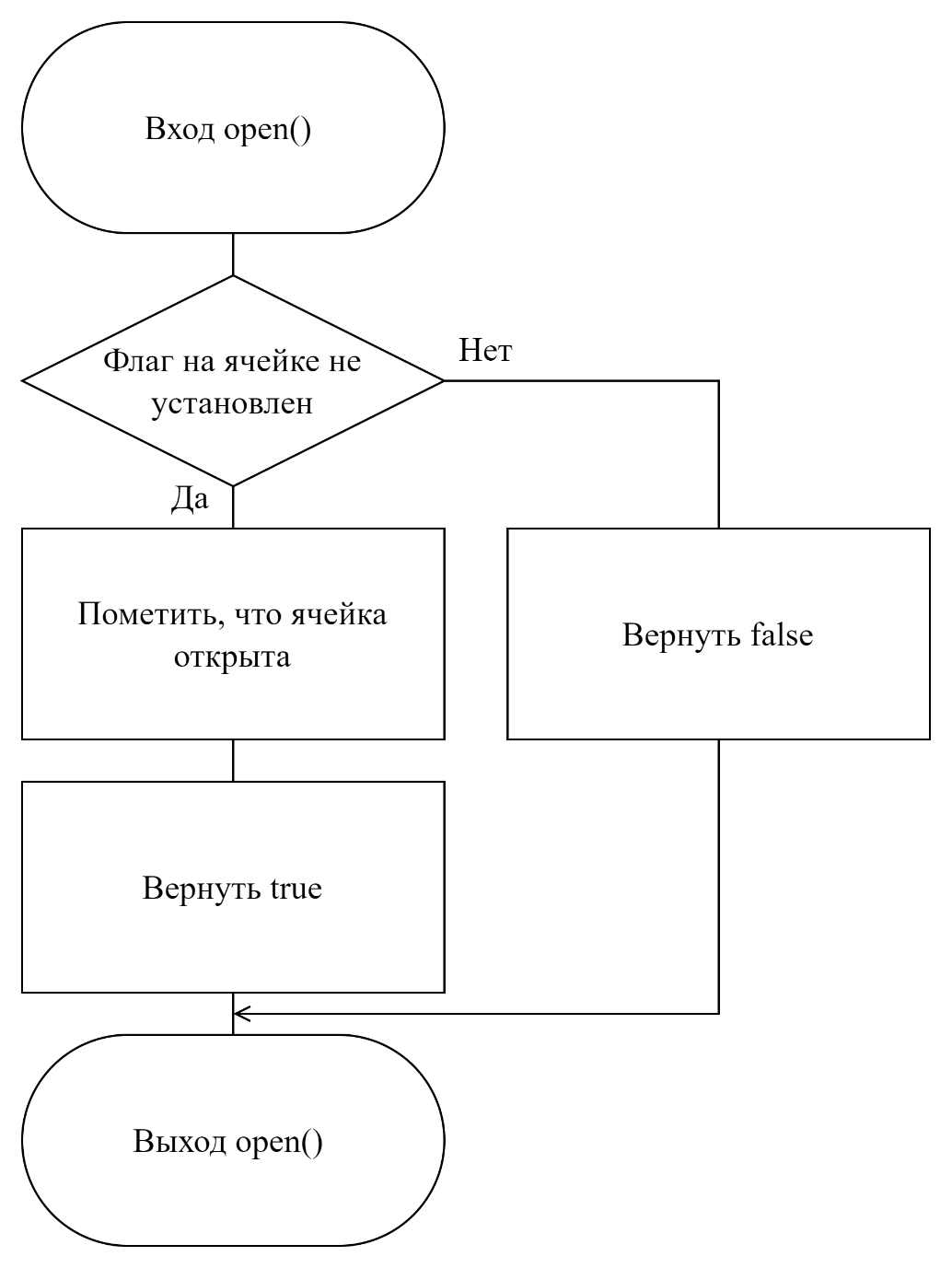


Рисунок 3.4 – Схема алгоритма open

### Схема алгоритма Field::moveAwayBomb

Схема алгоритма перемещения бомбы из указанной ячейки в случайную свободную ячейку на игровом поле. Применяется исключительно на 1-ом ходу, если игрок попал на бомбу.

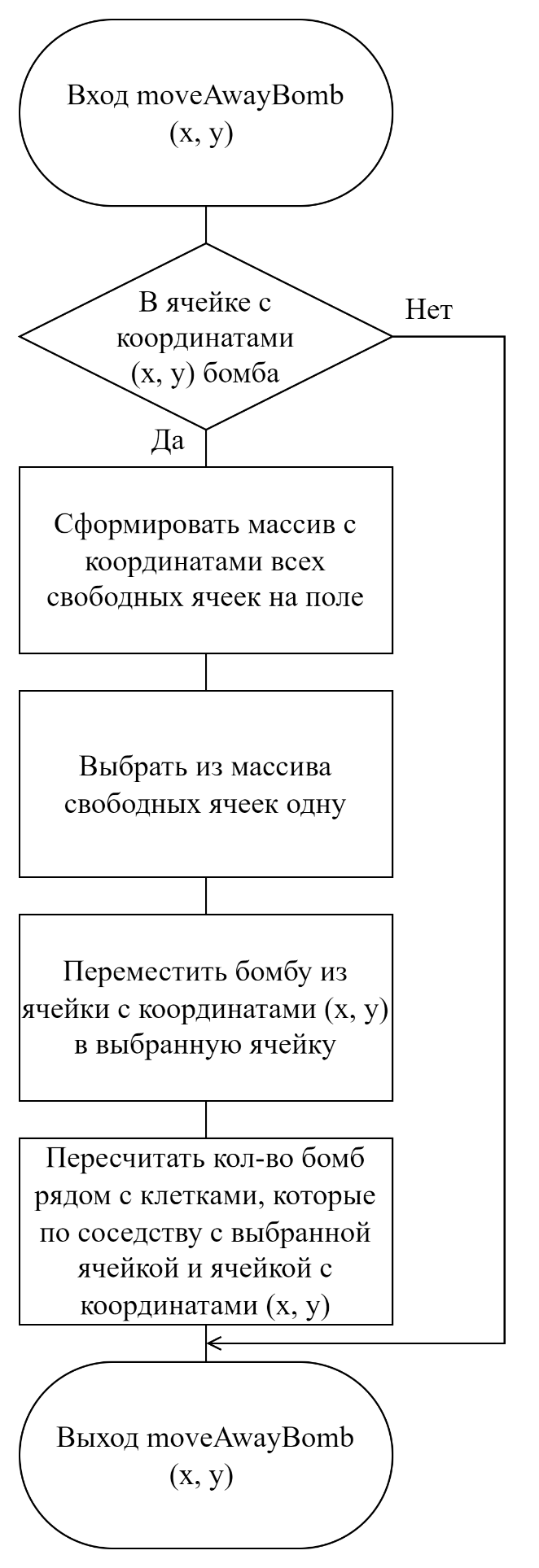


Рисунок 3.5 – Схема алгоритма moveAwayBomb

# тестирование и проверка работоспособности программного средства

## Взаимодействие с меню

### Тест 1

Таблица 3 – Тест1

|  |  |
| --- | --- |
| Тестовая ситуация: | Проверка корректности отображения меню и игры при запуске программы |
| Исходный набор данных: | Запуск программы |
| Ожидаемый результат: | Корректное открытие меню и игры |
| Полученный результат: |  |

### Тест 2

Таблица 4 – Тест 2

|  |  |
| --- | --- |
| Тестовая ситуация: | Проверка корректности открытия основного подменю |
| Исходный набор данных: | Открытие основного подменю |
| Ожидаемый результат: | Корректное открытие основного подменю |

Продолжение таблицы 4 – Тест 2

|  |  |
| --- | --- |
| Полученный результат: |  |

### Тест 3

Таблица 5 – Тест 3

|  |  |
| --- | --- |
| Тестовая ситуация: | Проверка корректности открытия модального окна с информацией о программе |
| Исходный набор данных: | Нажатие на кнопку «Справка» |
| Ожидаемый результат: | Корректное открытие модального окна с информацией о программе |
| Полученный результат: |  |

## Игровой процесс

### Тест 4

Таблица 6 – Тест 4

|  |  |
| --- | --- |
| Тестовая ситуация: | Проверка корректности открытия ячейки на поле |
| Исходный набор данных: | Нажатие левой кнопкой мыши на ячейку |
| Ожидаемый результат: | Открытие ячейки |
| Полученный результат: |  |

### Тест 5

Таблица 7 – Тест 5

|  |  |
| --- | --- |
| Тестовая ситуация: | Проверка корректности установки флага на ячейку |
| Исходный набор данных: | Нажатие правой кнопкой мыши на ячейку |
| Ожидаемый результат: | Установка флага на ячейку и уменьшение счетчика в левом верхнем углу |
| Полученный результат: |  |

### Тест 6

Таблица 8 – Тест 6

|  |  |
| --- | --- |
| Тестовая ситуация: | Проверка корректности реагирования программы при нажатии на ячейку с флагом |
| Исходный набор данных: | Нажатие на ячейку с флагом |
| Ожидаемый результат: | Ячейка не открывается |
| Полученный результат: |  |

### Тест 7

Таблица 9 – Тест 7

|  |  |
| --- | --- |
| Тестовая ситуация: | Проверка корректности открытия нескольких ячеек при нажатии на одну с определенным условием |
| Исходный набор данных: | Нажатие на ячейку |
| Ожидаемый результат: | Открытие нескольких ячеек |
| Полученный результат: |  |

### Тест 8

Таблица 10 – Тест 8

|  |  |
| --- | --- |
| Тестовая ситуация: | Проверка корректности смены сложности |
| Исходный набор данных: | Выбор уровня сложности «низкий» в меню |
| Ожидаемый результат: | Создание нового поля, обновление счетчиков времени и флагов |
| Полученный результат: |  |

### Тест 9

Таблица 11 – Тест 9

|  |  |
| --- | --- |
| Тестовая ситуация: | Проверка корректности ввода желаемого размера поля |
| Исходный набор данных: | Ввод поля размером 15x15 |
| Ожидаемый результат: | Создание нового поля, обновление счетчиков времени и флагов |

Продолжение таблицы 11 – Тест 9

|  |  |
| --- | --- |
| Полученный результат: |  |

### Тест 10

Таблица 12 – Тест 10

|  |  |
| --- | --- |
| Тестовая ситуация: | Проверка корректности завершения игры |
| Исходный набор данных: | Нажатие на мину |
| Ожидаемый результат: | Корректное завершение игры |
| Полученный результат: |  |

### Тест 11

Таблица 13 – Тест 11

|  |  |
| --- | --- |
| Тестовая ситуация: | Проверка корректности создания нового поля |
| Исходный набор данных: | Нажатие на смайлик |
| Ожидаемый результат: | Корректное создание нового поля |
| Полученный результат: |  |

# Руководство по использованию программного средства

## Меню

В меню у игрока есть возможность выбрать желаемый уровень сложности, от которого зависят размер поля и количество мин. Для выбора сложности нужно нажать соответствующую кнопку в меню. Также присутствует возможность создать поле с желаемыми размерами, для чего в меню присутствует пункт сложности «Пользовательская». После нажатия на соответствующую кнопку появится модальное окно, в котором есть возможность ввести характеристики поля.

## Игра

Для начала игры необходимо нажать левой кнопкой мыши на любую из ячеек поля, после чего одна или несколько ячеек откроются. Для нахождения местоположения мин необходимо смотреть на цифры в ячейках, которые обозначают количество мин, находящихся рядом. Для упрощения процесса можно обозначать мины флажками, нажимая на соответствующие ячейки правой кнопкой мыши. В левом верхнем углу окна находится счетчик поставленных флагов, в правом верхнем углу счетчик времени текущей игры в секундах.

## Новая игра

Для начала новой игры можно выбрать пункт «Новая игра» в меню или же нажать левой кнопкой мыши на смайлик, находящийся посередине в верхней части окна.

## Завершение игры

Игра завершается победой при открытии всех ячеек, не содержащих мин, или поражением при открытии ячейки, содержащей мину.

Заключение

В результате разработки игры «Сапер» было получено приложение, позволяющее полноценно испытать достоинства и недостатки данной игры.

Данная игра является простой в освоении и интуитивно понятной, так как все функции реализованы с использованием максимально понятных механик.

Для успешного выполнения всех поставленных целей потребовалось ознакомиться со средой разработки Visual Studio. Для создания графического интерфейса требовалось изучить различные аналоги игры в данном жанре. Также потребовалось изучить возможности Windows API.

Приложение прошло все этапы тестирования, в результате которых были устранены все неполадки. Приложение имеет относительно высокую скорость работы. Возможна дальнейшая доработка при выявлении ошибок.

Список использованной литературы

[1] Серебряная Л.В. Структуры и алгоритмы обработки данных: учеб.-метод. пособие / Л.В. Серебряная, И.М. Марина. – Минск: БГУИР, 2013. – 5 с.

[2] Дэвид Дж. Круглински. Программирование на Microsoft Visual c++ 6.0 / Дэвид Дж. Круглински, Скотт Уингоу, Джордж Шеферд. – Санкт-Петербург: ПИТЕР, 2001. – 819 с.

[3] Microsoft[Электронный ресурс]. – Режим доступа: [https://learn.microsoft.com/en-us/windows/win32/api/](https://learn.microsoft.com/en-us/windows/win32/api/%20) – Дата обращения: 07.12.2023.

[4] Ravesli[Электронный ресурс]. – Режим доступа: [https://ravesli.com/](https://ravesli.com/%20) – Дата обращения: 05.12.2023.

[5] StackOverflow[Электронный ресурс]. – Режим доступа: <https://stackoverflow.com/> – Дата обращения: 05.12.2023.

[6] Metanit[Электронный ресурс]. – Режим доступа: <https://metanit.com/cpp/> – Дата обращения: 24.11.2023.

Приложение А

(обязательное)

Исходный код программы

Модуль Cell

#include "Cell.h"

Cell::Cell() : m\_inner{ Type::Types::EMPTY }, m\_cover{ Type::Types::CLOSED }, m\_isOpened{ false }

{

}

Type Cell::getInner()

{

return m\_inner;

}

void Cell::setInner(Type inner)

{

m\_inner = inner;

}

Type Cell::getCover()

{

return m\_cover;

}

void Cell::setCover(Type cover)

{

m\_cover = cover;

}

bool Cell::isOpened()

{

return m\_isOpened;

}

Type Cell::getType()

{

return m\_isOpened ? m\_inner : m\_cover;

}

Type Cell::changeFlag()

{

if (!m\_isOpened)

{

if (m\_cover == Type::Types::FLAG)

{

m\_cover = Type::Types::CLOSED;

}

else

{

m\_cover = Type::Types::FLAG;

}

return m\_cover;

}

return m\_inner;

}

bool Cell::open()

{

if (m\_cover != Type::Types::FLAG)

{

m\_isOpened = true;

return true;

}

return false;

}