содержание

[ВВЕДЕНИЕ 4](#_Toc438248399)

[1 АНАЛИЗ ИСХОДНЫХ ДАННЫХ И ПОСТАНОВКА ЗАДАЧ 5](#_Toc438248400)

[2 Проектирование программы 6](#_Toc438248401)

[3 РЕАЛИЗАЦИЯ ПРОГРАММЫ 8](#_Toc438248402)

[4 ТЕСТИРОВАНИЕ ПРОГРАММЫ 20](#_Toc438248403)

[ЗАКЛЮЧЕНИЕ 24](#_Toc438248404)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 25](#_Toc438248405)

[ПРИЛОЖЕНИЕ А. МОДУЛЬНАЯ СХЕМА ПРОГРАММЫ 26](#_Toc438248406)

[ПРИЛОЖЕНИЕ Б. ФУНКЦИОНАЛЬНАЯ СХЕМА ПРОГРАММЫ 27](#_Toc438248407)

[ПРИЛОЖЕНИЕ В. ПРОТОТИП ГРАФИЧЕСКОГО ИНТЕРФЕЙСА 28](#_Toc438248408)

**ВВЕДЕНИЕ**

Java – объектно-ориентированный язык программирования общего назначения. Особенностью языка является компиляция программ в специальный промежуточный код (байт-код), который затем интерпретируется виртуальной машиной Java (JVM). Преимуществом данного подхода является отличная переносимость программ, так как программы могут быть запущены без изменений на любой системе, под которую написана виртуальная машина. Также достоинством является безопасность выполнения кода, так как JVM контролирует выполнение программы и прерывает ее выполнение при превышении допустимых полномочий.

Недостатками данного подхода является снижение производительности из-за интерпретации байт-кода виртуальной машиной во время выполнения программ. Однако данная проблема частично решена благодаря применению JIT-компиляции и платформенно-ориентированного кода в стандартной библиотеке.

Также особенностью языка Java является использование сборщика мусора. Он представляет собой подпрограмму JVM, который выполняет удаление из памяти неиспользуемых объектов. Такой подход облегчает разработку программ, так как разработчику не приходится самому следить за использованием памяти и вручную удалять неиспользуемые объекты.

Android – это операционная система предназначенная для смартфонов, планшетов, различных носимых устройств, нетбуков и телевизоров. Основным разработчиком является компания Google. Android позволяет создавать Java-приложения, взаимодействующие с устройством через разработанные компанией Google библиотеки. Данная операционная система основана на ядре Linux и реализации JVM от Google. Она является наиболее распространенной на мобильных платформах, и разработка приложений для нее очень актуальна.

Целью выполнения данной курсовой работы является систематизация и закрепление теоретических знаний, полученных за время обучения, а также приобретение практических навыков в таких сферах разработки как проектирование программы, поиск и реализация необходимых алгоритмов, тестирование программы и разработка документации.

1 АНАЛИЗ ИСХОДНЫХ ДАННЫХ

Темой данной курсовой работы является разработка программы для анализа статистики игроков в игре World of Tanks на языке программирования Java для операционной системы Android.

Данное приложение должно обладать следующим функционалом:

* Поиск игрока по имени
* Отображение общей статистики по аккаунту;
* Отображение статистики игрока по технике игрока с выводом средних значений;
* Возможность сортировки техники по различным показателям;
* Отображение специальных рейтингов РЭ, WN6, WN8;
* Отображение общей сессионной статистики;
* Отображение сессионной статистики отдельно по каждому танку;
* Построение графиков динамики изменения средних показателей по каждому танку;
* Построение графиков динамики изменения средних показателей по аккаунту;
* Графики статистики по каждому танку;
* Возможность вывода статистики по танкам с определенным количеством боев;
* Хранение истории статистики и поиска.

Данные получаются с серверов компании Wargaming, отдельно для каждого региона с помощью Wargaming API путем отправки HTTP-запросов. Результаты запроса возвращаются в формате JSON.

Для разработки приложения нам понадобится:

* Среда разработки IntelliJ IDEA 15.0.3;
* Enterprise Architect для построения диаграммы вариантов использования.
* Смартфон с установленной системой Android 5.0 и выше либо аналогичный эмулятор для тестирования работы приложения.

**2 программНОЕ ПРОЕКТИРОВАНИЕ**

Для начала реализации проекта нам необходимо создать и настроить проект Android приложения в среде разработки IntelliJ IDEA. При разработке будут использованы библиотеки android-support-v4 и android-support-v13, которые содержат классы, облегчающие реализацию интерфейса с использованием вкладок. Также нам необходимо скачать библиотеку MPAndroidChartLibrary. Она содержит богатый функционал для построения графиков и диаграмм различных типов. Для обработки ответов от сервисов Wargaming будет использоваться библиотека com.parse, которая позволяет работать с JSON объектами. Интерфейс программы для операционной системы Android будет построен на основе XML разметки.

При первом запуске должно будет выводиться окно, с предложением выбрать игрока. Для этого необходимо будет щелкнуть на кнопку в правом верхнем углу экрана. Затем после выбора игрока отобразится окно «Статистика игрока», являющееся главным окном программы. Данное окно будет содержать две вкладки:

* Общая статистика;
* Техника игрока.

Переключение между различными меню будет осуществляется с помощью выдвигающегося бокового меню. Там будет содержаться шесть пунктов:

* Статистика игрока
* Динамика статистики
* Сессии
* История статистики
* Инфографика
* Графики по технике

Если был осуществлен первый запуск, то при выборе пунктов меню «Динамика статистики» и «Сессии» будет выведено предложение сыграть несколько боев для построения графиков. Иначе на вкладке «Динамика статистики» будут выведены графики, показывающие динамику изменения статистики игрока. Различные показатели будут вынесены в отдельные вкладки. А во вкладке «Сессии» будут отображаться подробные показатели игрока за сессии, сыгранные в определенный период времени.

В пункте меню «История статистики» будет отображаться упрощенный список сессий игрока, который позволит быстро проследить количественное изменение ключевых показателей статистики.

На вкладке «Инфографика» будут представлены диаграммы, показывающие распределение различных показателей статистики по различным критериям.

Пункт меню «Графики по технике» будет содержать графики, которые будут визуально представлять статистику игрока на каждом отдельном танке.

Для обновления статистики будет необходимо открыть пункт меню «Статистика игрока».

* Вся информация будет храниться в базе данных SQLite, которая изначально поддерживается системой Android. Для работы с базой данных будет создан отдельный класс. База данных будет содержать таблицы для хранения данных об аккаунтах, игровых сессиях, общей статистике игрока на отдельных танках, последней сессии игрока на танках. Также там будут храниться информация о танках и средние показатели игроков на всех танках.

Для каждого пункта меню и окна выбора игрока будут созданы отдельные классы и соответствующие им XML документы, содержащие разметку графического интерфейса.

Также необходимо создать класс, который будет взаимодействовать с сервисами Wargaming. Он будет отвечать за отправку запросов на получение данных и принимать ответы в формате JSON, которые затем будут обработаны с помощью библиотеки com.parse и добавлены в базу данных.

Также будут созданы вспомогательные классы, представляющие различные составляющие элементы интерфейса приложения.

3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

**3.1 Детальная реализация функциональных частей ПО**

Ключевым классом является класс MainActivity. Он отвечает за первоначальную инициализацию приложения, создание меню и управляет переключением между различными пунктами меню.

Основные методы данного класса:

* onCreate
* onSectionAttached
* onActivityResult
* onOptionsItemSelected

Также внутри класса MainActivity содержится класс PlaceHolderFragment, который отвечает за передачу сообщения о том, что в меню был какой-либо пункт.

Метод onCreate вызывается при запуске приложения и выполняет начальную инициализацию. Он представлен в листинге 1:

Листинг 1 – метод onCreate

1. protected void onCreate(Bundle savedInstanceState) {
2. super.onCreate(savedInstanceState);
3. setContentView(R.layout.activity\_main);
4. Utils.init(this);
5. mNavigationDrawerFragment = (NavigationDrawerFragment) getFragmentManager().findFragmentById(R.id.navigation\_drawer);
6. mTitle = getTitle();
7. dbHelper = DatabaseHelper.createDatabaseHelper(this);
8. SQLiteDatabase sdb = dbHelper.getWritableDatabase();
9. mainPlayer = getString(R.string.ttl\_search\_name);
10. Cursor cursor = sdb.rawQuery("SELECT \* FROM " + DatabaseHelper.DATABASE\_TABLE\_MAIN\_PLAYER + " WHERE active = 1", null);
11. if (cursor.getCount() > 0) {
12. cursor.moveToFirst();
13. mainPlayer = cursor.getString(1);
14. }
15. cursor.close();
16. this.mNavigationDrawerFragment.setUp(R.id.navigation\_drawer, (DrawerLayout) findViewById(R.id.drawer\_layout));
17. }

Метод onSectionAttached отвечает за смену отображаемого окна при выборе какого-либо пункта меню. Код данного метода представлен в листинге 2:

Листинг 2 – Метод onSectionAttached

1. public void onSectionAttached(int number) {
2. switch (number) {
3. case 1:
4. tmp = new MainStatFragment();
5. mTitle = getString(R.string.title\_section1);
6. break;
7. case 2:
8. mTitle = getString(R.string.title\_section2);
9. tmp = new StatGraphRoot();
10. break;
11. case 3:
12. mTitle = getString(R.string.title\_section3);
13. tmp = new SessionHistoryRoot();
14. break;
15. case 4:
16. tmp = new StatHistory();
17. mTitle = getString(R.string.title\_section4);
18. break;
19. case 5:
20. tmp = new Infographics();
21. mTitle = getString(R.string.title\_section5);
22. break;
23. case 6:
24. tmp = TanksGraph.newInstance("1", "2");
25. mTitle = getString(R.string.title\_section6);
26. break; }
27. getFragmentManager().beginTransaction().replace(R.id.container, tmp).commit(); }

Метод OnOptionsItemSelected отвечает за обработку кнопки меню в верхнем правом углу. Если будет выбран пункт выбрать игрока, то будет создано новое окно для поиска нового игрока либо выбора из списка существующего. Код данного метода представлен в листинге 3:

Листинг 3 – Метод onOptionsItemSelected

1. public boolean onOptionsItemSelected(MenuItem item) {
2. int id = item.getItemId();
3. if (id == R.id.action\_settings) {
4. return true;}
5. if (id != R.id.active\_player) {
6. return super.onOptionsItemSelected(item);}
7. startActivityForResult(new Intent(this, SelectPlayer.class), CHOOSE\_PLAYER);
8. return true;
9. }

Метод onActivityResult отвечает за обработку события выбора нового игрока. В нем из окна выбора игрока получается имя игрока, проверяется, есть ли сведения о данном игроке в системе, и в случае присутствия ставится метка о том, что данный игрок является активным в системе. В случае же отсутствия, в базу данных добавляется новая запись и ставится выше упомянутая метка.

За отрисовку и обработку бокового меню отвечает класс NavigationDrawerFragment.

Основные методы класса:

* onCreateView
* setUp
* selectItem

Метод onCreateView вызывается при открытии бокового меню, и отвечает за создание элементов меню. Его код представлен в листинге 4:

Листинг 4 – Метод onCreateView

1. public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
2. mDrawerListView = (ListView) inflater.inflate (R.layout.fragment\_navigation\_drawer, container, false);
3. mDrawerListView.setOnItemClickListener(new OnItemClickListener() {
4. @Override
5. public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
6. selectItem(position);
7. }
8. });
9. listValueName = new ArrayList<>();
10. listValueName.add(getString(R.string.title\_section1));
11. listValueName.add(getString(R.string.title\_section2));
12. listValueName.add(getString(R.string.title\_section3));
13. listValueName.add(getString(R.string.title\_section4));
14. listValueName.add(getString(R.string.title\_section5));
15. listValueName.add(getString(R.string.title\_section6));
16. mDrawerListView.setAdapter(new MyArrayAdapter (mDrawerListView.getContext(), mDrawerListView, listValueName));
17. mDrawerListView.setItemChecked(mCurrentSelectedPosition, true);
18. return mDrawerListView;
19. }

Метод setUp выполняет инициализацию полей класса.

Метод selectItem выполняет смену выбранной позиции меню. Он представлен в листинге 5:

Листинг 5 – Метод selectItem

1. private void selectItem(int position) {
2. mCurrentSelectedPosition = position;
3. if (mDrawerListView != null) {
4. mDrawerListView.setItemChecked(position, true);}
5. if (mDrawerLayout != null) {
6. mDrawerLayout.closeDrawer(mFragmentContainerView);}
7. if (mCallbacks != null) {
8. mCallbacks.onNavigationDrawerItemSelected(position);}
9. }

Также внутри данного класса содержится класс MyArrayAdapter, который отвечает за отображение каждого пункта меню.

За поиск и выбор аккаунта игрока отвечает класс SelectPlayer. Основные методы данного класса:

* onCreate
* registerClickCallback
* buttonSearchPlayerClick

А также он содержит классы MyArrayAdapter и PrefetchDataCurrency.

Метод onCreate отвечает за создание окна выбора игрока и выводит в данном окне историю поиска аккаунтов.

Метод registerClickCallball выполняет регистрацию обработчика нажатия на элемент списка игроков, который возвращает результат в класс MainActivity.

Метод buttonSearchPlayerClick используется в качестве обработчика нажатия для кнопки «Поиск». Он запускает процесс поиска игрока.

Класс MyArrayAdapter выполняет вывод каждого элемента списка найденных игроков.

Класс PrefetchDataCurrency отвечает за отправку запроса о поиске игроков и обработку результатов.

За отправку запросов на сервер отвечает класс JSONParser. Он содержит три метода: getJSON, getJSONArrayFromUrl и getJSONObjFromUrl.

Метод getJSON выполняет подключение к серверу, отправляет HTTP запрос и получает ответ в виде строки JSON. Метод представлен в листинге 6:

Листинг 6 – Метод getJSON

1. private String getJSON(String requestUrl) {
2. StringBuilder builder = new StringBuilder();
3. try {
4. URL url = new URL(requestUrl);
5. HttpURLConnection connection = (HttpURLConnection) url.openConnection();

Листинг 6 - Продолжение

1. connection.setReadTimeout(10000);
2. connection.setConnectTimeout(10000);
3. connection.setRequestMethod("GET");
4. connection.setDoInput(true);
5. if (connection.getResponseCode() == HttpURLConnection.HTTP\_OK) {
6. BufferedReader reader = new BufferedReader(new InputStreamReader(connection.getInputStream()), 8);
7. while (true) {
8. String line = reader.readLine();
9. if (line != null) {
10. builder.append(line).append("\n");
11. } else {
12. connection.disconnect();
13. return builder.toString();}}}
14. Log.e("Error....", "Failed to download file");
15. return null;
16. } catch (IOException e3) {
17. e3.printStackTrace();
18. return null;}
19. }

Метод getJSONArrayFromUrl берет результат метода getJSON и преобразует его в объект JSONArray. Метод getJSONObjFromUrl действует аналогично, однако возвращает объект JSONObj.

Для облегчения работы с базой данных был создан класс DatabaseHelper. Он выполняет создание всех таблиц при первом запуске, а также содержит строковые константы для упрощения доступа к таблицам и их столбцам.

Класс MainStatFragment представляет пункт меню «Статистика игрока». Его ключевые методы:

* refreshPlayerStats
* onCreateView
* getColorByWins
* getColorByEFF
* getColorByWN6
* getColorByWN8
* viewPlayerTank
* setLabelColor

Метод refreshPlayerStats запускает поток, который выполняет обновление статистики игрока. Он запускается при выборе игрока либо пункта меню «Статистика игрока».

Метод onCreateView вызывается при открытии пункта меню «Статистика игрока» и отвечает за вывод общей статистики игрока и средних показателей по технике.

Методы getColorByWins, getColorByEFF, getColorByWN6, getColorByWN8 возвращают значение цвета, в зависимости от значения рейтинга.

Метод viewPlayerTank получает данные из базы данных о статистике на всех танках и представляет их в виде списка.

Метод setLabelColor задает значение цвета для текстовых полей.

Назначение классов MyArrayAdapter и PrefetchDataCurrency совпадает с описаннымы выше аналогичными классами.

Пункт меню «Сессии» реализован при помощи классов SessionHistoryRoot, SessionHistoryMainFragment и SessionHistoryItem. Главным здесь является класс SessionHistoryRoot, который управляет построением интерфейса, например, таких элементов как вкладки. SessionHistoryMainFragment включается в класс SessionHistoryRoot и управляет переключением вкладок. SessionHistoryItem представляет каждую отдельную вкладку-сессию, на которой выводятся все показатели за сессию и изменение общей статистики.

Пункт меню «Динамика статистики» реализован с помощью классов StatGraphRoot и StatGraphMainFragment и TanksGraphItem. Назначение классов StatGraphRoot и StatGraphMainFragment аналогично назначению классов SessionHistoryRoot и SessionHistoryMainFragment соответственно. Класс TanksGraphitem представляет линейный график, показывающий динамику изменения статистики игрока по разным показателям.

Пункт меню «История статистики» реализован единственным классом StatHistory. Он получает список сессий из базы данных и выводит их в виде списка на экран.

Аналогично пункт меню «Инфографика» реализован единственным классом Infographics. Он получает данные из базы данных, выполняет их обработку, разбивая на определенные группы по различным критериям. Затем по каждому критерию строятся сравнительные диаграммы.

Пункт меню «Графики по технике» реализован в классе TanksGraph. Он содержит 4 вкладки с графиками, построенными на основе различных параметров для каждого конкретного танка.

Также в ходе реализации программы были созданы дополнительные классы SlidingTabLayout и SlidingTabStrip. Они обеспечивают работу со вкладками в интерфейсе приложения. А класс ServersChangeHelper предназначен для работы приложения с различными регионами.

**3.2 Сопроводительная документация ПО**

**3.2.1 Минимальные системные требования**

Для работы приложения необходимо:

* Устройство либо эмулятор с установленной системой Android 5.0
* 10 МБ свободного места во внутреннем хранилище
* Доступ в сеть интернет

**3.2.2 Описание установки и запуска приложения**

Перед установкой убедитесь, что в настройках включена галочка ставить приложения из неизвестных источников. Для установки приложения необходимо скопировать apk файл на устройство либо эмулятор, запустить его и проследовать инструкциям по установке.

Для запуска необходимо открыть главное меню и щелкнуть на иконку приложения.

**3.2.3 Руководство пользователя**

При запуске приложения в первый раз будет выведено страница с предложением найти игрока. Для поиска игрока необходимо щелкнуть на значок настроек в правом верхнем углу и нажать на пункт «Найти игрока». Будет открыто окно поиска, в котором необходимо ввести имя аккаунта, выбрать регион и нажать на кнопку поиска. Если поиск будет выполнен успешно, то будет выведен список найденных игроков. Для выбора игрока необходимо щелкнуть на его имя в списке. Если ранее были загружены данные каких-либо игроков, то они будут сохранены в системе, а в окне поиска будет отображен список с историей поиска.

Для навигации по приложению необходимо использовать боковое меню. После первого запуска при открытии пунктов меню «Динамика статистики» и «Сессии» будет выведено окно с предложением сыграть несколько боев, т.к. в системе отсутствуют данные для отслеживания динамики.

Для просмотра подробной статистики по танку в пункте меню Статистика игрока –> Техника игрока необходимо щелкнуть по записи в списке. При щелчке по выпадающему списку можно выбрать различные поля для сортировки списка техники. Галочка изменения оставляет в списке только те танки, для которых есть сохраненные сессии. Также для таких танков будет подсвечена кнопка с изображением графика. Это значит, что для танка доступен просмотр динамики статистики. Если сохраненные сессии отсутствуют, то при клике на данную кнопку будет доступна только вкладка «Средние значения».

На вкладке «Динамика статистики» есть возможность просмотра динамики за определенное число последних сессий либо за все сохраненные сессии. Также есть возможность отобразить границы рейтингов на графиках.

На вкладке «Сессии» почти каждый пункт меню кликабелен и позволяет просмотреть график изменения того или иного показателя статистики.

На вкладке «Графики по технике» есть возможность отфильтровать танки по количеству проведенных боев.

Для обновления статистики необходимо перейти на вкладку «Статистика игрока».

**3.3 Анализ ПО**

**3.3.1 Анализ исходного кода приложения**

Для анализа исходного кода приложения была использована функция Анализ кода в программе IntelliJ IDEA.

3.4 Тестирование Программы

При запуске программы пользователю показывается основное окно программы.

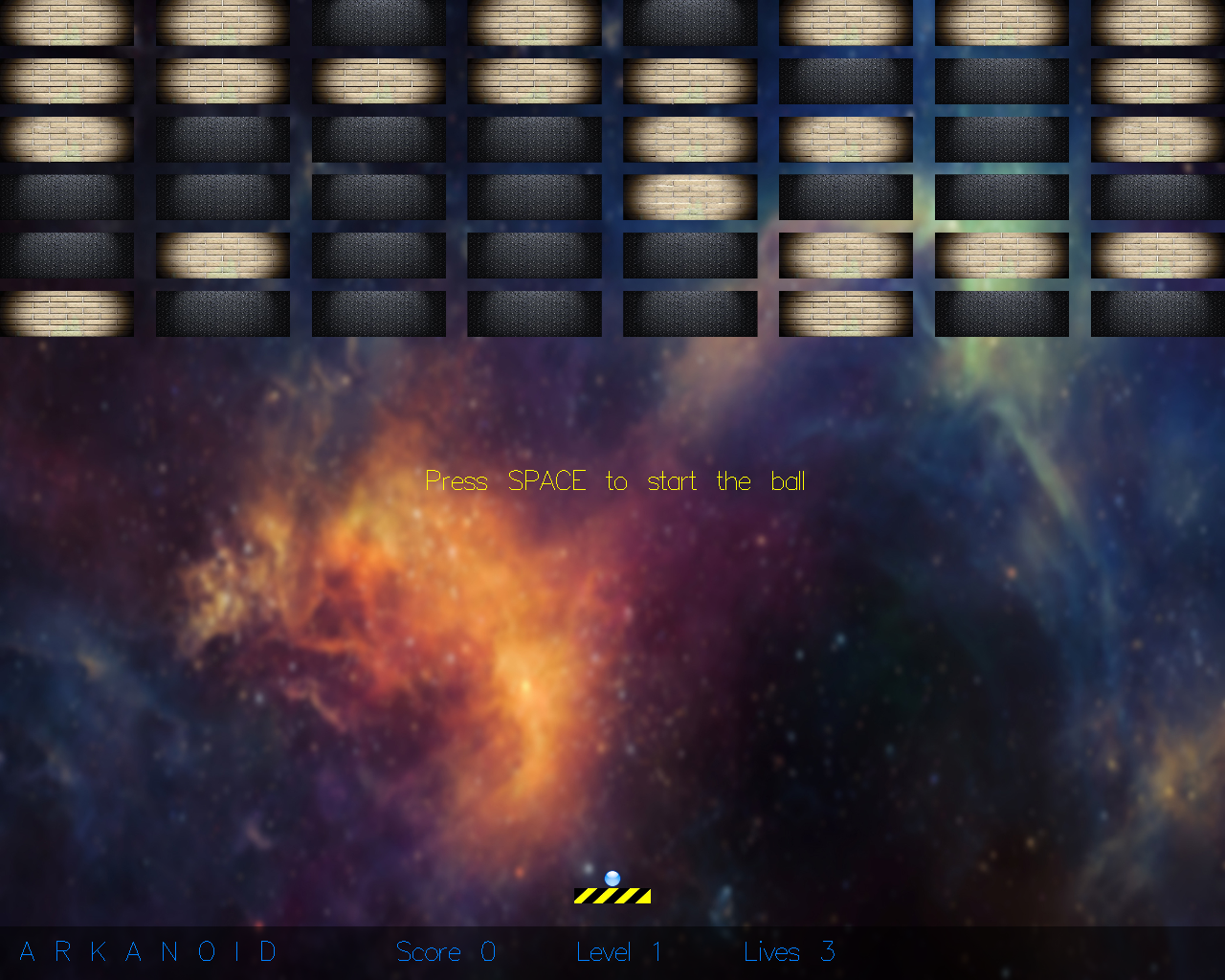


Рисунок 4.1 – Стартовое окно программы

При нажатии на клавишу пробел произойдет запуск шарика. При столкновении шарика с блоком, блок уничтожается, а шарик отскакивает в нужном направлении. Начисляется 25 очков.



Рисунок 4.2 – Запуск шарика и удар по блоку

При потере шарика количество жизней уменьшается на 1, а шарик восстанавливается на платформе.

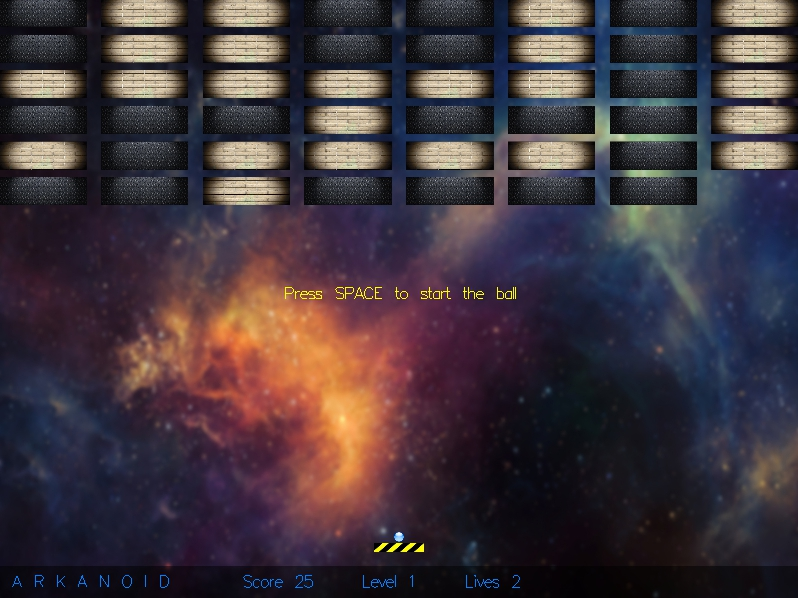


Рисунок 4.3 – Потеря шарика

При уничтожении блока есть шанс на выпадение приза.

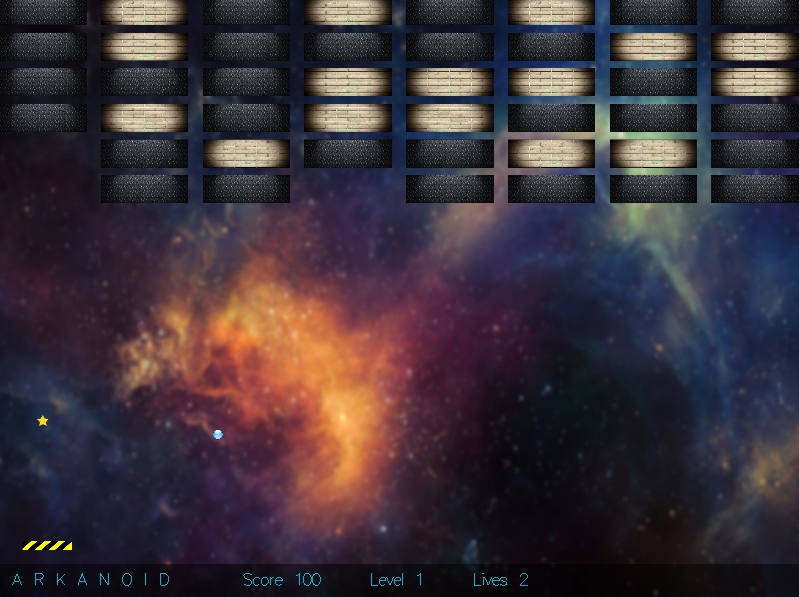


Рисунок 4.4 – Выпадение приза

Если поймать этот приз платформой, то он активируется и увеличит платформу.



Рисунок 4.5 – Активация приза

При нажатии на клавишу TAB вызывается таблица рекордов.



Рисунок 4.6 – Таблица рекордов при нажатии TAB

При исчерпании количества жизней игра заканчивается и на экран выводится таблица рекордов. Она отличается от таблицы, вызываемой клавишей TAB, наличием крестика в правом верхнем углу, при нажатии на который таблица закрывается и запускается новая игра.



Рисунок 4.7 – Окончание игры

При наведении на крестик он меняет цвет.



Рисунок 4.8 – Изменение цвета крестика

ЗАКЛЮЧЕНИЕ

В курсовом проекте была поставлена задача разработки игрового работоспособного ПО с использованием языка C++ без объектно-ориентированных возможностей. Программа должна представлять собой игру «Арканоид». Программа должна обладать интуитивно понятным интерфейсом.

Удалось реализовать программу, которая способна:

1. Обеспечивать движение и столкновение игровых объектов
2. Использовать звуки и изображения из файлов
3. Начислять очки, используя систему подсчета очков
4. Использовать систему выпадающих призов
5. Информировать об окончании игры
6. Отображать таблицу рекордов на 10 записей

Интерфейс программы интуитивно понятен любому пользователю, который будет использовать программу.

Был произведен разбор всех функций приложения и их тестирование с разбором основных возможностей.

Программа оснащена звуковыми эффектами и умеет загружать картинки игровых объектов из файлов.

В итоге реализовано приложение с интуитивно понятным интерфейсом, обладающее всеми необходимыми возможностями. Были реализованы все функции, задуманные вначале.

При выполнении курсового проекта были получены практические навыки проектирования графического интерфейса, разработки и реализации алгоритма работы программы, ее тестирования и разработки документации. Были изучены возможности языка программирования C++, программного интерфейса OpenGL и библиотеки GLUT(OpenGL Utility Toolkit).

Для использования OpenGL и GLUT произведена настройка Visual Studio и подключены все необходимые библиотеки и заголовочные файлы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Ламот А. Программирование игр для Windows. Советы профессионала, 2-е изд. : Пер. с англ. — М. : Издательский дом “Вильямс”, 2003. — 880 с. : ил. — Парал. тит. Англ
2. Лафоре Р. Объектно-ориентированное программирование в C++-СПБ.: Издательство «Питер», 2004. – 928 с.
3. Райт Р. Липчак Б. OpenGL. Суперкнига, 3-е издание.: Пер. с англ – М.: Издательский дом «Вильямс», 2006. – 1040 с.: ил. – Парал. тит. англ.
4. The OpenGL Utility Toolkit (GLUT) Programming Interface API Version 3 //[Электронный ресурс]. 1996 - https://www.opengl.org/documentation/specs/glut

/spec3/spec3.html – Дата доступа: 30.10.2015

1. Урок: базовые алгоритмы определения столкновений //[Электронный ресурс]. 2004 - http://noregret.org/tutor/n/collision/ - Дата доступа: 05.12.2015

# ПРИЛОЖЕНИЕ А. МОДУЛЬНАЯ СХЕМА ПРОГРАММЫ

ARKANOID

OBJECTS

GRAPHICS

Рисунок А.1 – Модульная схема программы

**ПРИЛОЖЕНИЕ Б. ФУНКЦИОНАЛЬНАЯ СХЕМА ПРОГРАММЫ**

WinMain

glutMainLoop

Init

ChangeSize

MouseMove

LoadTexture

MouseClick

KeyboardUpFunc

SpecialReleaseFunc

KeyboardFunc

SpecialPressFunc

ProcessBall

HitDetection

MoveBall

TimerFunc

ProcessPrize

EndGame

InitBlocks

Draw

DrawBlocks

LoadFile

DrawPrize

SaveFile

HighscrDraw

Draw\_Text

Рисунок Б.1 – Функциональная схема программы

**ПРИЛОЖЕНИЕ В. ПРОТОТИП ГРАФИЧЕСКОГО ИНТЕРФЕЙСА**

Рисунок В.1 – Прототип графического интерфейса