Министерство образования и науки Российской Федерации

ФГБОУ ВО Ульяновский государственный технический университет

Кафедра «Вычислительная техника»

Лабораторная работа №5

«Базы данных SQLite»

Выполнил студент

группы ИВТАПбд-21

Мухаметзянов Т. А.

Ульяновск, 2022

**Задача:**

Требуется доработать лабораторную работу №4 таким образом, чтобы программа сохраняла в базе данных SQLite расширенную статистику игрового процесса – записывать данные каждого игрового сеанса, такие, как дата и время начала, длительность игрового процесса, количество набранных очков. По статистике осуществлять выбор самого длительного сеанса игры, самого результативного (по очкам), вывести список сеансов игр, отсортированный по убыванию количества очков или по убыванию длительности игрового сеанса. Можно реализовать оба варианта.

**Особенности реализации**

1. Класс DBHelper предназначен для инициализации БД и ее создания, а также ее обновления.

onUpgrade(SQLiteDatabase *sqLiteDatabase*, int *i*, int *i1*) **–** вызывается при обновлении структуры БД.

onCreate(SQLiteDatabase *sqLiteDatabase*) // Создание таблицы в БД. Срабатывает при создании базы данных

1. Класс NameBase предназначен для работы с БД.

В нем реализованы методы отображения таблицы, методы отображения с различной сортировкой по колонкам, в зависимости от сортировки возвращается либо полная таблица либо только нужные строчки.

// названия колонок  
public static final String COLUMN\_ID = "\_id";  
public static final String COLUMN\_DATE = "date";  
public static final String COLUMN\_TIME = "time";  
public static final String COLUMN\_STEPS = "steps";  
public static final String COLUMN\_POINTS = "points";  
  
// номера колонок  
public static int **NUM\_COLUMN\_ID** = 0;  
public static int **NUM\_COLUMN\_DATE** = 1;  
public static int **NUM\_COLUMN\_TIME** = 2;  
public static int **NUM\_COLUMN\_POINTS** = 3;  
public static int **NUM\_COLUMN\_STEPS** = 4;

Сортировка по очкам и вывод лучшего результата:

// сортировка по очкам  
String orderBy = "points";  
public Names bestPointsSelect() {  
  
 Cursor mCursor = mDataBase.query(TABLE\_NAME, null, null, null, null, null, orderBy + " DESC");  
 ArrayList<Names> list = new ArrayList<Names>();  
 mCursor.moveToFirst();  
  
 if (!mCursor.isAfterLast()) {  
 do {  
 long id = mCursor.getLong(**NUM\_COLUMN\_ID**);  
 String date = mCursor.getString(**NUM\_COLUMN\_DATE**);  
 int time = mCursor.getInt(**NUM\_COLUMN\_TIME**);  
 int points = mCursor.getInt(**NUM\_COLUMN\_POINTS**);  
 int steps = mCursor.getInt(**NUM\_COLUMN\_STEPS**);  
 Names stats = new Names(id, date, time, points, steps);  
 list.add(stats);  
 } while (mCursor.moveToNext());  
 }  
  
 mCursor.close();  
  
 return list.get(0);  
}

1. В MainActivity у нас 2 кнопки для перехода на активити игры или же БД:



В этой активити создаются методы для обновления списка по разным критерям, в которых используются различные методы сортировки из NameBase:

// вывод списка по убыванию времени  
public void UpdateListTimeUb () {  
 adapter.clear();  
 for(Names n: DBConnector.worstTimeSelect()) {  
 adapter.add(n.getId() + ": Дата: " + n.getDate() + ", Время: " + n.getTime() + ", Очки: "  
 + n.getPoints() + ", Шаги: " + n.getSteps());  
 }  
}

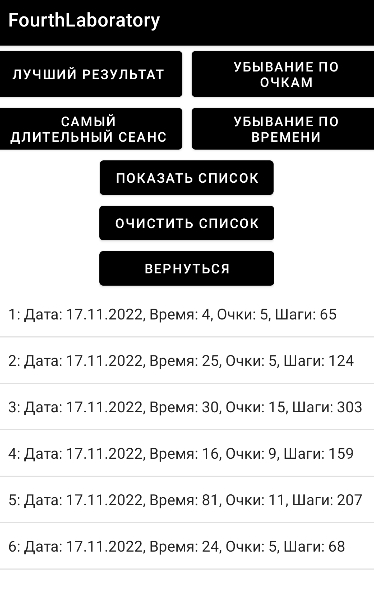
Получаем данные намерением из другого класса:

// получение статистики из gameView  
date = getIntent().getStringExtra("date");  
time = getIntent().getIntExtra("time", 0);  
points = String.**valueOf**(getIntent().getIntExtra("points", 0));  
steps = getIntent().getIntExtra("steps", 0);

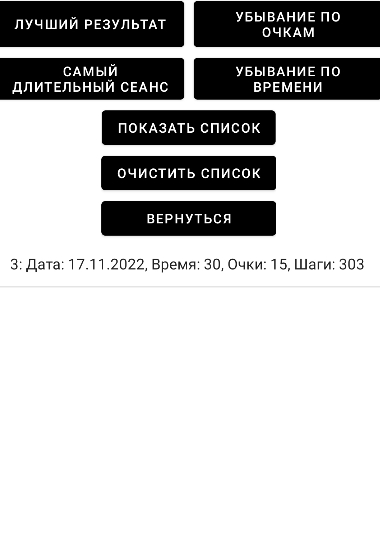
Добавление в бд и адаптер:

// добавление статистики в список и бд, если данные не null  
if(date != null && time != 0 && points != null && steps != 0) {  
  
 adapter = new ArrayAdapter<String>(this, android.R.layout.simple\_list\_item\_1, stats);  
 DBConnector.add(date, String.**valueOf**(time), points, String.**valueOf**(steps));  
 UpdateList();  
  
}

**Тестирование:**

****

**Рис. 1. Вид списка**

****

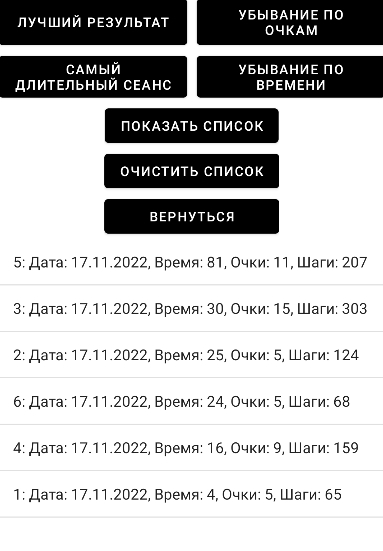
**Рис. 2. Лучший результат**

****

**Рис. 3. Убывание по очкам**

****

**Рис. 4. Самый длительный сеанс**

****

**Рис. 5. Убывание по длительности игры**

**Вывод о проделанной работе:**

Работа с написанием созданием локальной БД и занесением туда данных о статистике после игры выполнена успешно, программа работает корректно.