Федеральное агентство связи
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет
телекоммуникаций и информатики»

Лабораторная работа по теме:
«Базы данных SQLite»

Выполнили:
студентки 3 курса
ИВТ, гр. ИП-712
Гервас А.В.
Онищенко А.В.

Новосибирск 2020

Оглавление

Задание

Создать базу данных студентов (Имя, вес, рост, возраст - сгенерировать случайно). Вывести из базы данных все записи, отсортированные по возрасту, в таблицу (TableLayout).

Скриншоты

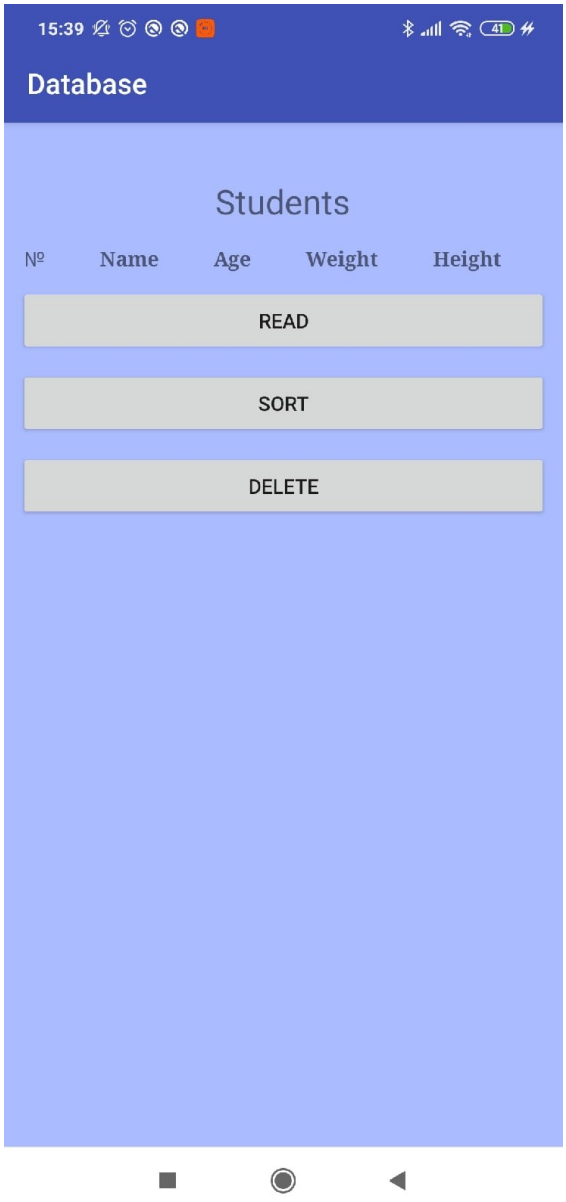

Рис. 1. Стартовое окно приложения

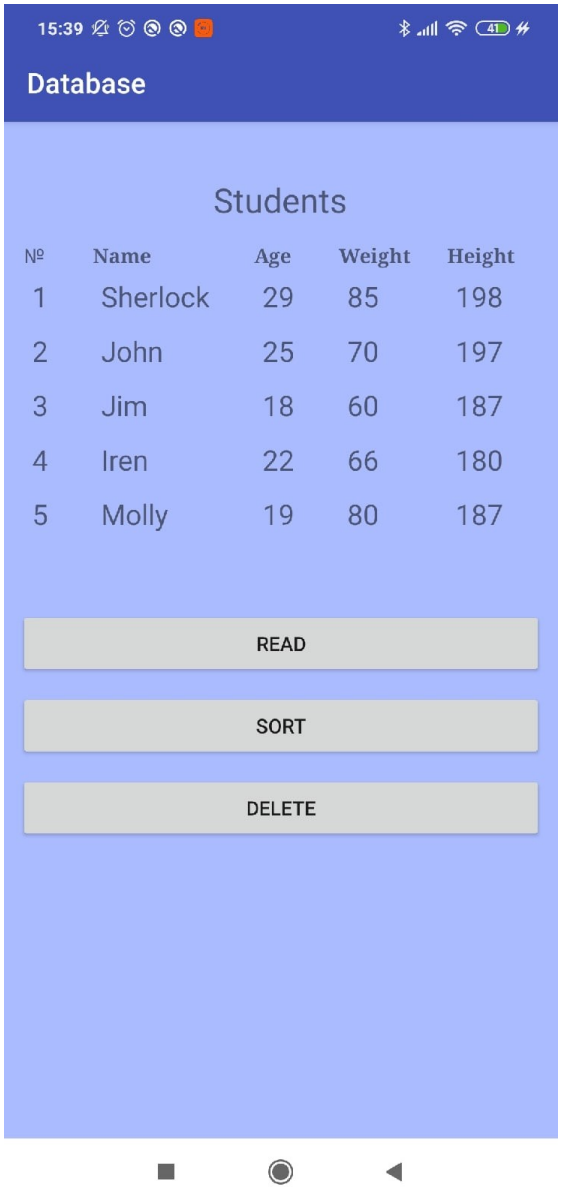

Рис. 2. Чтение таблицы

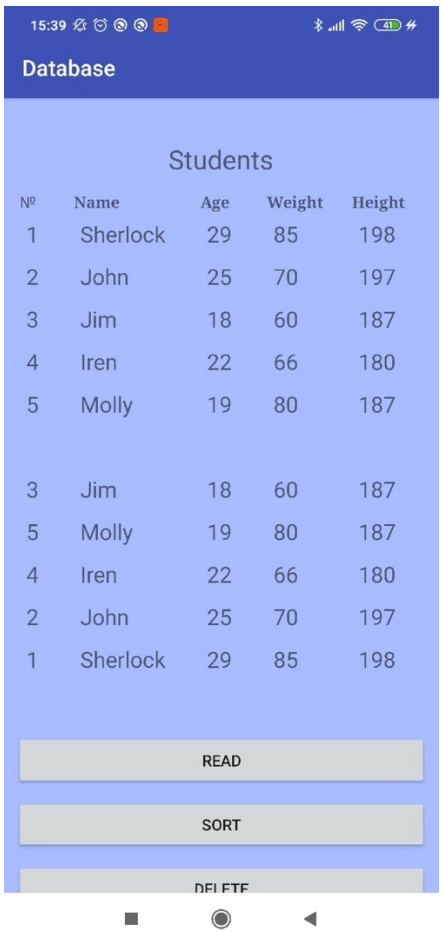

Рис. 3. Сортировка БД по возрасту

Листинг кода

Приложение написано на языке Kotlin.

MainActivity.kt

```
package ru.sibsutis.database

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.Gravity
import android.widget.*
import kotlinx.android.synthetic.main.activity_main.*
import kotlin.random.Random

val Names = arrayOf("Sherlock", "John", "Jim", "Iren", "Molly")

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val db = DBHelper(this)
        val tl = findViewById<TableLayout>(R.id.tl)
        tl.isShrinkAllColumns = true
        val readBtn = findViewById<Button>(R.id.btn_read)
        var readCount = 0
        var sortCount = 0
        for (i in 0..4) {
            val student = User(
                Names[i],
                Random.nextInt(18, 30), //age
                Random.nextInt(60, 100), //weight
                Random.nextInt(160, 210) //height
            )
            db.insertData(student)
        }

        readBtn.setOnClickListener {
            sortCount = 0
            readCount++
            if (readCount <= 1) {

                val students = db.readData()
                if (students.isEmpty()) {
                    Toast.makeText(this, "Database is empty", Toast.LENGTH_SHORT).show()
                } else {
                    students.trimToSize()
                    for (i in 0..4) {
                        val tr = TableRow(this)
                        tr.layoutParams = TableLayout.LayoutParams(
                            TableRow.LayoutParams.MATCH_PARENT,
                            TableRow.LayoutParams.WRAP_CONTENT
                        )
                        tr.gravity = Gravity.CENTER

                        val tv01 = TextView(this)
                        tv01.text = students[i].id.toString()
                        tv01.textSize = 20f
                        tv01.setPadding(16, 16, 16, 16)
                        tr.addView(tv01)

                        val tv02 = TextView(this)
                        tv02.text = students[i].name
                        tv02.textSize = 20f
                        tv02.setPadding(16, 16, 16, 16)
```

```
                tr.addView(tv02)

                val tv03 = TextView(this)
                tv03.text = students[i].age.toString()
                tv03.textSize = 20f
                tv03.setPadding(16, 16, 16, 16)
                tr.addView(tv03)

                val tv04 = TextView(this)
                tv04.text = students[i].weight.toString()
                tv04.textSize = 20f
                tv04.setPadding(16, 16, 16, 16)
                tr.addView(tv04)

                val tv05 = TextView(this)
                tv05.text = students[i].height.toString()
                tv05.textSize = 20f
                tv05.setPadding(16, 16, 16, 16)
                tr.addView(tv05)

                tl.addView(tr)
            }

            val tr2 = TableRow(this)
            tr2.layoutParams = TableLayout.LayoutParams(
                TableRow.LayoutParams.MATCH_PARENT,
                TableRow.LayoutParams.WRAP_CONTENT
            )
            tr2.gravity = Gravity.CENTER

            val tv11 = TextView(this)
            tv11.text = ""
            tv11.textSize = 20f
            tv11.setPadding(16, 16, 16, 16)
            tr2.addView(tv11)

            val tv12 = TextView(this)
            tv12.text = ""
            tv12.textSize = 20f
            tv12.setPadding(16, 16, 16, 16)
            tr2.addView(tv12)

            val tv13 = TextView(this)
            tv13.text = ""
            tv13.textSize = 20f
            tv13.setPadding(16, 16, 16, 16)
            tr2.addView(tv13)

            val tv14 = TextView(this)
            tv14.text = ""
            tv14.textSize = 20f
            tv14.setPadding(16, 16, 16, 16)
            tr2.addView(tv14)

            val tv15 = TextView(this)
            tv15.text = ""
            tv15.textSize = 20f
            tv15.setPadding(16, 16, 16, 16)
            tr2.addView(tv15)

            tl.addView(tr2)
        }
    }
}

btn_sort.setOnClickListener {
//      db.updateData()
//      btn_read.performClick()
```

```
        readCount = 0
        sortCount++
        if (sortCount > 1) {
        } else {
            val students = db.sort()
            if (students.isEmpty()) {
                Toast.makeText(this, "Database is empty", Toast.LENGTH_SHORT).show()
            } else {
                for (i in 0..4) {
                    val tr = TableRow(this)

                    tr.layoutParams = TableLayout.LayoutParams(
                        TableRow.LayoutParams.MATCH_PARENT,
                        TableRow.LayoutParams.WRAP_CONTENT
                    )
                    tr.gravity = Gravity.CENTER

                    val tv01 = TextView(this)
                    tv01.text = students[i].id.toString()
                    tv01.textSize = 20f
                    tv01.setPadding(16, 16, 16, 16)
                    tr.addView(tv01)

                    val tv02 = TextView(this)
                    tv02.text = students[i].name
                    tv02.textSize = 20f
                    tv02.setPadding(16, 16, 16, 16)
                    tr.addView(tv02)

                    val tv03 = TextView(this)
                    tv03.text = students[i].age.toString()
                    tv03.textSize = 20f
                    tv03.setPadding(16, 16, 16, 16)
                    tr.addView(tv03)

                    val tv04 = TextView(this)
                    tv04.text = students[i].weight.toString()
                    tv04.textSize = 20f
                    tv04.setPadding(16, 16, 16, 16)
                    tr.addView(tv04)

                    val tv05 = TextView(this)
                    tv05.text = students[i].height.toString()
                    tv05.textSize = 20f
                    tv05.setPadding(16, 16, 16, 16)
                    tr.addView(tv05)

                    tl.addView(tr)
                }

                val tr2 = TableRow(this)
                tr2.layoutParams = TableLayout.LayoutParams(
                    TableRow.LayoutParams.MATCH_PARENT,
                    TableRow.LayoutParams.WRAP_CONTENT
                )
                tr2.gravity = Gravity.CENTER

                val tv11 = TextView(this)
                tv11.text = ""
                tv11.textSize = 20f
                tv11.setPadding(16, 16, 16, 16)
                tr2.addView(tv11)

                val tv12 = TextView(this)
                tv12.text = ""
                tv12.textSize = 20f
                tv12.setPadding(16, 16, 16, 16)
                tr2.addView(tv12)
```

```
                val tv13 = TextView(this)
                tv13.text = ""
                tv13.textSize = 20f
                tv13.setPadding(16, 16, 16, 16)
                tr2.addView(tv13)

                val tv14 = TextView(this)
                tv14.text = ""
                tv14.textSize = 20f
                tv14.setPadding(16, 16, 16, 16)
                tr2.addView(tv14)

                val tv15 = TextView(this)
                tv15.text = ""
                tv15.textSize = 20f
                tv15.setPadding(16, 16, 16, 16)
                tr2.addView(tv15)

                tl.addView(tr2)
            }
        }
    }

    btn_del.setOnClickListener {
        db.deleteDB()
        btn_read.performClick()
        // Toast.makeText(context, "Deleted!", Toast.LENGTH_SHORT).show()
    }
  }
}
```

DBHelper.kt

```
package ru.sibsutis.database

import android.content.ContentValues
import android.content.Context
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper
import android.widget.Toast

val DATABASE_NAME = "sql"
val TABLENAME = "students"
val COL_ID = "_id"
val COL_NAME = "name"
val COL_AGE = "age"
val COL_WEIGHT = "weight"
val COL_HEIGHT = "height"

class DBHelper(var context: Context) : SQLiteOpenHelper(context, DATABASE_NAME, null, 1) {

  override fun onCreate(db: SQLiteDatabase?) {
      val createTable = "CREATE TABLE $TABLENAME (" +
              COL_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
              COL_NAME + " TEXT, " +
              COL_AGE + " INTEGER, " +
              COL_WEIGHT + " INTEGER, " +
              COL_HEIGHT + " INTEGER);"
      db?.execSQL(createTable)
  }

  override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {}

  fun insertData(user: User) {
      val db = this.writableDatabase
      val cv = ContentValues()
      cv.put(COL_NAME, user.name)
```

```
        cv.put(COL_AGE, user.age)
        cv.put(COL_WEIGHT, user.weight)
        cv.put(COL_HEIGHT, user.height)
        db.insert(TABLENAME, null, cv)

//        if (res == (-1).toLong())
//            Toast.makeText(context, "Failed", Toast.LENGTH_SHORT).show()
//        else Toast.makeText(context, "Success", Toast.LENGTH_SHORT).show()

        db.close()
    }

    fun readData(): ArrayList<User> {
        val list = ArrayList<User>()
        val db = this.readableDatabase
        val res = db.rawQuery("SELECT * FROM $TABLENAME", null)
        var user: User
        if (res.moveToFirst()) {
            do {
                user = User()
                user.id = res.getInt(res.getColumnIndex(COL_ID))
                user.name = res.getString(res.getColumnIndex(COL_NAME))
                user.age = res.getInt(res.getColumnIndex(COL_AGE))
                user.weight = res.getInt(res.getColumnIndex(COL_WEIGHT))
                user.height = res.getInt(res.getColumnIndex(COL_HEIGHT))
                list.add(user)

            } while (res.moveToNext())
        }
        res.close()
        db.close()
        return list
    }

    fun updateData() {
        val db = this.writableDatabase
        val res = db.rawQuery("SELECT * FROM $TABLENAME", null)
        if (res.moveToFirst()) {
            do {
                var cv = ContentValues()
                cv.put(COL_AGE, (res.getInt(res.getColumnIndex(COL_AGE)) + 1))
                db.update(
                    TABLENAME, cv,
                    COL_ID + "=? AND " + COL_NAME + "=? AND " + COL_WEIGHT + "=? AND " + COL_HEIGHT + "=?",
                    arrayOf(
                        res.getString(res.getColumnIndex(COL_ID)),
                        res.getString(res.getColumnIndex(COL_NAME)),
                        res.getString(res.getColumnIndex(COL_WEIGHT)),
                        res.getString(res.getColumnIndex(COL_HEIGHT))
                    )
                )
            } while (res.moveToNext())
        }
        res.close()
        db.close()
    }

    fun sort(): ArrayList<User> {
        val students = ArrayList<User>()
        val db = this.readableDatabase
        val c = db.rawQuery("SELECT * FROM $TABLENAME ORDER BY $COL_AGE", null)
        var user: User
        if (c.moveToFirst()) {
            do {
                user = User()
                user.id = c.getInt(c.getColumnIndex(COL_ID))
                user.name = c.getString(c.getColumnIndex(COL_NAME))
                user.age = c.getInt(c.getColumnIndex(COL_AGE))
                user.weight = c.getInt(c.getColumnIndex(COL_WEIGHT))
```

```
            user.height = c.getInt(c.getColumnIndex(COL_HEIGHT))
            students.add(user)

        } while (c.moveToNext())
    }
    c.close()
    db.close()
    return students
}

fun deleteDB() {
    val db = this.writableDatabase
    db.delete(TABLENAME, null, null)
    db.close()
}

}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/screenBackground"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Students"
        android:textSize="24sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.057" />

    <androidx.constraintlayout.widget.Guideline
        android:id="@+id/guideline"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        app:layout_constraintGuide_begin="84dp" />

    <ScrollView
        android:id="@+id/scrollView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintHorizontal_bias="0.756"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/guideline"
        app:layout_constraintVertical_bias="0.003">

        <TableLayout
            android:id="@+id/tl"
            android:layout_width="374dp"
            android:layout_height="142dp"
            android:shrinkColumns="*"
            android:stretchColumns="*"
```

```
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.486"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView"
        app:layout_constraintVertical_bias="0.043">

        <TableRow
            android:id="@+id/tableRow2"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <TextView
                android:id="@+id/TextView01"
                android:text="№" />

            <TextView
                android:id="@+id/TextView02"
                android:text="Name"
                android:textStyle="bold"
                android:typeface="serif" />

            <TextView
                android:id="@+id/TextView03"
                android:text="Age"
                android:textStyle="bold"
                android:typeface="serif" />

            <TextView
                android:id="@+id/TextView04"
                android:text="Weight"
                android:textStyle="bold"
                android:typeface="serif" />

            <TextView
                android:id="@+id/TextView05"
                android:text="Height"
                android:textStyle="bold"
                android:typeface="serif" />
        </TableRow>
    </TableLayout>
</ScrollView>

<Button
    android:id="@+id/btn_read"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:padding="10dp"
    android:text="Read"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.842"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/scrollView2"
    app:layout_constraintVertical_bias="0.0" />

<Button
    android:id="@+id/btn_sort"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:layout_marginStart="8dp"
    android:padding="10dp"
    android:text="Sort"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/btn_read"
    app:layout_constraintVertical_bias="0.0" />
```

```
    <Button
        android:id="@+id/btn_del"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        android:padding="10dp"
        android:text="Delete"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.842"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btn_sort"
        app:layout_constraintVertical_bias="0.0" />


</androidx.constraintlayout.widget.ConstraintLayout>
```