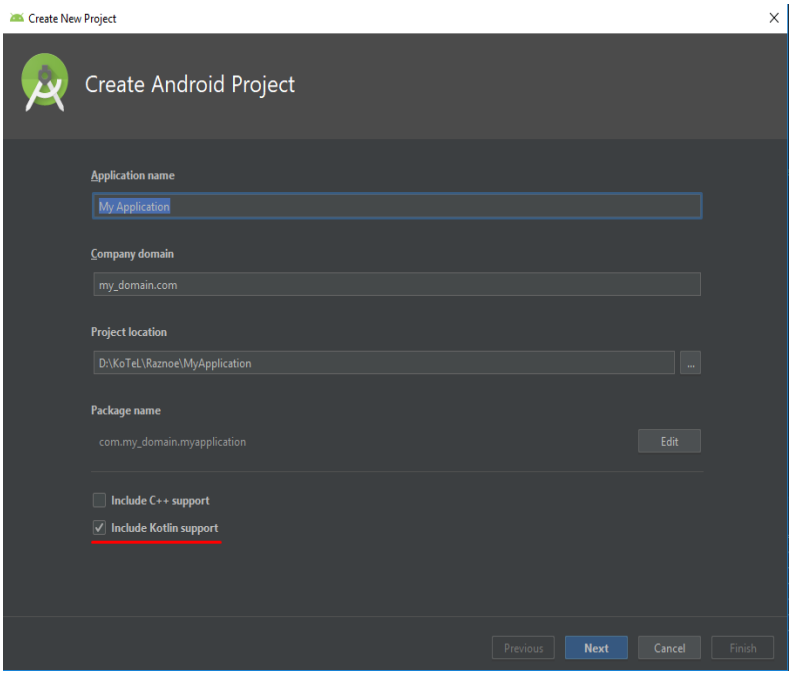
Руководство по выполнению ИПР 1

Приложение для ведения заметок

Для того что бы написать приложение на языке Kotlin, нужно в самом начале при создании проекта нажать галочку «Include Kotlin support».



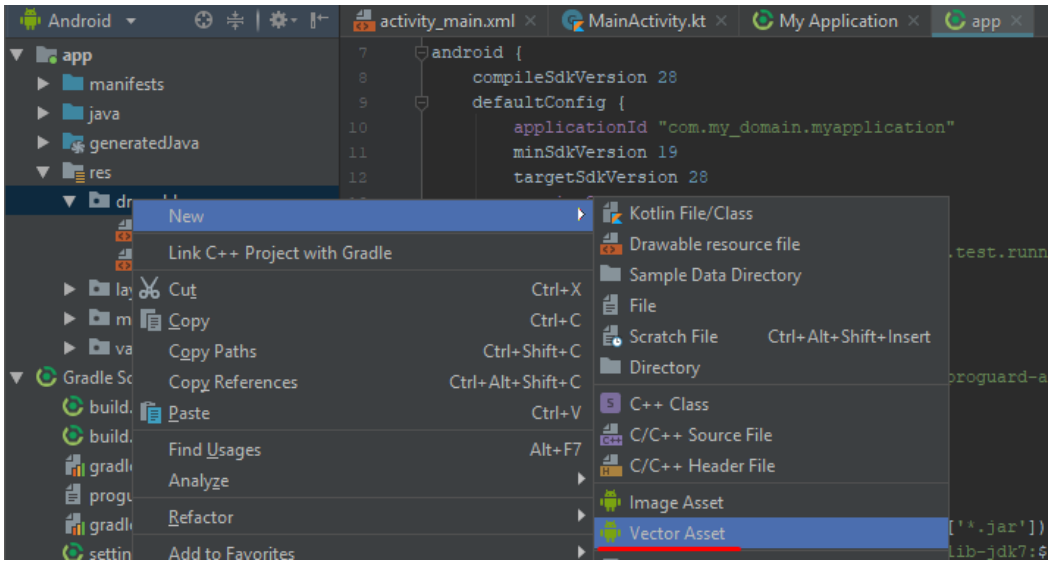
После того как чистый проект будет создан, для написания приложения «Заметки», нужно подключить 2 библиотеки:

```
implementation 'com.android.support:cardview-v7:28.0.0'
implementation 'com.android.support:design:28.0.0'
```

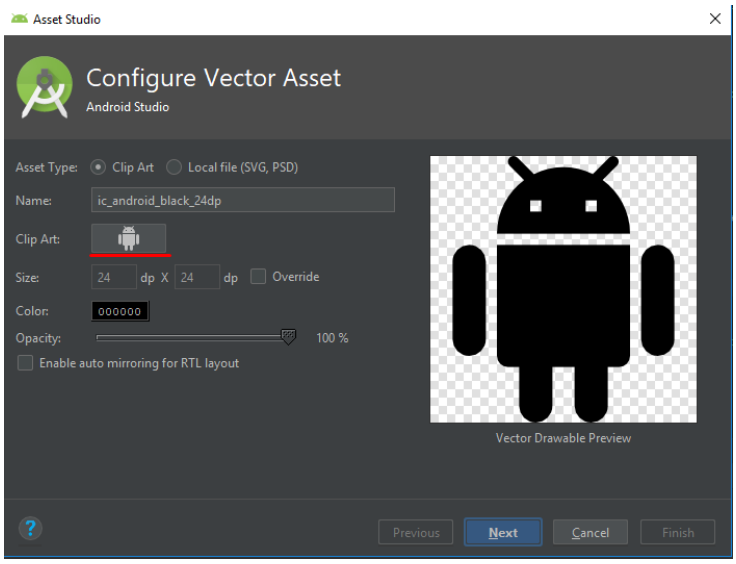
```
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
    implementation 'com.android.support:appcompat-v7:28.0.0'
    implementation 'com.android.support.constraint:constraint-layout:1.1.3'
    implementation 'com.android.support:cardview-v7:28.0.0'
    implementation 'com.android.support:design:28.0.0'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:runner:1.0.2'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'
}
```

И нажать Sync Now в правом верхнем углу чтобы применить изменения в Gradle.

Далее добавляем иконки которые пригодятся в процессе написания приложения.

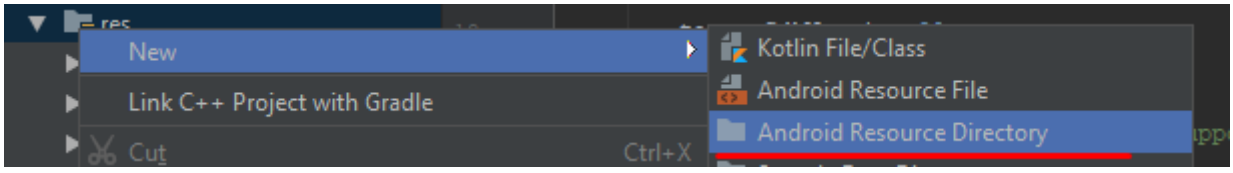


В открывшемся окне нажимаем на иконку Андроида и попадаем в каталог стандартных векторных изображений Android.



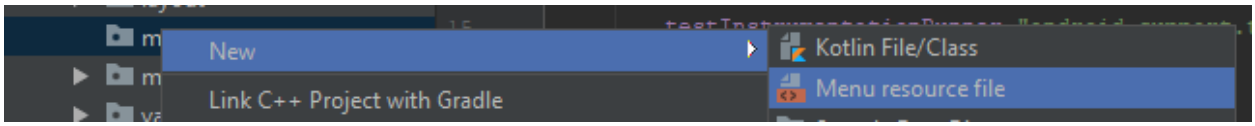
Находим нужную и нажимаем «Ok» -> «Next» -> «Finish». Нужно подобным образом поочерёдно добавить иконки с именами «Search», «Add», «Content copy», «Delete», «Share». В каталоге присутствует поиск таким образом можно найти и использовать другие иконки.

Далее создадим меню приложения. Для этого необходимо нажать правой кнопкой на папку «res» затем «New» и выбрать «Android Resource Direcrory»,



Вверху в выпадающем меню «Resource type» выбрать «menu» и нажать «Ok». Создастся папка в которой будут храниться файлы меню. Теперь создадим сам

файл меню, для этого нажимаем правой кнопкой на «menu» -> «New» -> «Menu resource file»



Присваиваем имя «main_menu» и нажмём «Ok». Вносим туда следующую вёрстку компонентов меню используя картинки, которые добавляли ранее.

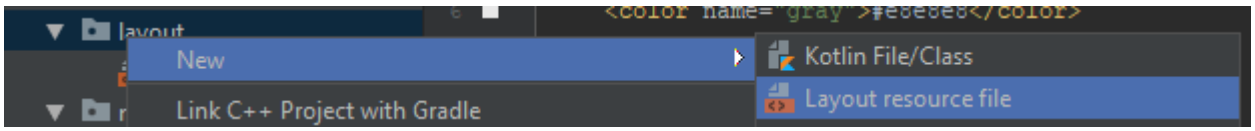
main_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
            android:id="@+id/app_bar_search"
            android:icon="@drawable/ic_search_black_24dp"
            android:title="Search"
            app:actionViewClass="android.support.v7.widget.SearchView"
            app:showAsAction="always" />
    <item
            android:id="@+id/addNote"
            android:icon="@drawable/ic_add_black_24dp"
            android:title="Add Nore"
            app:showAsAction="always" />
</menu>
```

В файл colors.xml находящийся в папке «values» добавляем следующие строки:

```
<color name="gray">#e8e8e8</color>
<color name="white">#fff</color>
<color name="black">#000</color>
```

Создадим файл «layout» в который внесем вёрстку отдельных компонентов списка заметок, правой кнопкой по папке «layout»->«New»-> «Layout resource file» назовём его «row» :



Напишем код вёрстки:

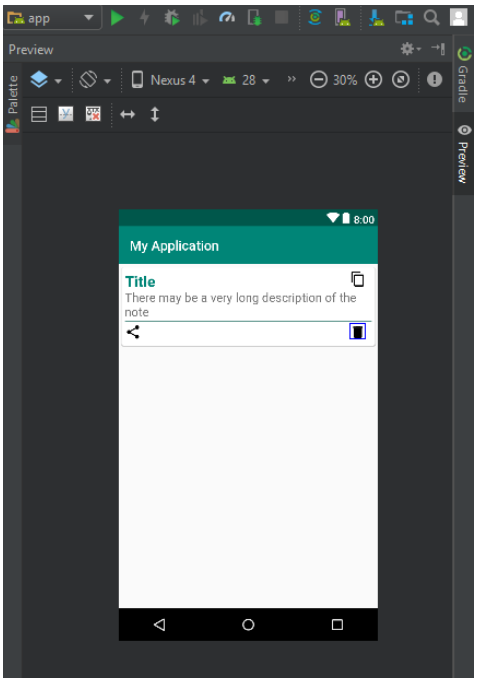
row.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:cardBackgroundColor="@color/white"
        app:cardCornerRadius="3dp"
        app:cardElevation="3dp"
```

```
    app:cardUseCompatPadding="true">

<LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="5dp"
        android:gravity="end|bottom"
        android:orientation="vertical">
    <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_weight="15"
            android:orientation="horizontal"
            android:weightSum="100">
        <LinearLayout
                android:layout width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="99">
            <TextView
                    android:id="@+id/titleTv"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:text="Title"
                    android:textColor="@color/colorPrimary"
                    android:textSize="22sp"
                    android:textStyle="bold" />
        </LinearLayout>
        <LinearLayout
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:gravity="end">
            <ImageButton
                    android:id="@+id/copyBtn"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginEnd="5dp"
                    android:layout_marginRight="5dp"
                    android:background="@null"
                    android:gravity="end"
                    android:src="@drawable/ic_content_copy_black_24dp"/>

        </LinearLayout>

</LinearLayout>



    <TextView
            android:id="@+id/descTv"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="There may be a very long description of the note"
            android:textSize="18sp" />

    <View
            android:layout width="match parent"
            android:layout_height="1dp"
            android:layout_marginBottom="3dp"
            android:layout_marginTop="2dp"
            android:background="@color/colorPrimaryDark" />
<LinearLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="15"
```

```
            android:orientation="horizontal"
            android:weightSum="100">
        <LinearLayout
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="99">
            <ImageButton
                    android:id="@+id/shareBtn"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginEnd="5dp"
                    android:layout_marginRight="5dp"
                    android:background="@null"
                    android:src="@drawable/ic_share_black_24dp" />
        </LinearLayout>
        <LinearLayout
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:gravity="end">
            <ImageButton
                    android:id="@+id/deleteBtn"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginEnd="5dp"
                    android:layout_marginRight="5dp"
                    android:background="@null"
                    android:src="@drawable/ic_delete_black_24dp"/>

        </LinearLayout>
    </LinearLayout>
    </LinearLayout>

</android.support.v7.widget.CardView>
```

Нажав Preview можете увидеть как будет выглядеть каждый компонент списка:



Затем создадим Activity которая будет отвечать за добавление новой заметки «AddNoteActivity», если создание Activity произошло правильно, то помимо «row.xml» в папке «layout» будет ещё 2 фала «activity_add_note» и «activity_main» это файлы в которых буде вёрстка Activity для добавления новой заметки и главного экрана, соответственно внесем вёрстку в них.

activity_add_note.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView 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"
            tools:context=".AddNoteActivity">

    <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

        <android.support.v7.widget.CardView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_margin="5dp"
                app:cardBackgroundColor="@color/white"
                app:cardCornerRadius="3dp"
                app:cardElevation="3dp">

            <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:orientation="vertical">

                <EditText
                        android:id="@+id/titleEt"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:background="@null"
                        android:hint="Enter Title"
                        android:padding="10dp"
                        android:singleLine="true"
                        android:textStyle="bold" />

                <EditText
                        android:id="@+id/descEt"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:background="@null"
                        android:gravity="top"
                        android:hint="Enter description..."
                        android:minHeight="100dp"
                        android:padding="10dp" />
            </LinearLayout>
        </android.support.v7.widget.CardView>

        <Button
                android:id="@+id/addBtn"
                style="@style/Base.Widget.AppCompat.Button.Colored"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
```

```
                android:layout_gravity="end"
                android:layout_marginEnd="5dp"
                android:layout_marginRight="5dp"
                android:onClick="addFunc"
                android:text="Add" />

    </LinearLayout>

</ScrollView>
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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/gray"
              android:orientation="vertical"
              tools:context=".MainActivity">

    <ListView
            android:id="@+id/notesLv"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:divider="@null"
            android:dividerHeight="1dp" />

</LinearLayout>
```

Теперь дополнительно к уже существующим создадим 2 класса Kotlin «DbManager» и «Note» которые нужны для взаимодействия с Базой данных в которой будут храниться все заметки и работы с конкретными заметками соответственно.

Note.kt

```
class Note(nodeID: Int, nodeName: String, nodeDes: String) {

    var nodeID: Int? = nodeID
    var nodeName: String? = nodeName
    var nodeDes: String? = nodeDes

}
```

DbManager.kt – класс отвечающий за сохранение, удаление и добавление заметок в БД приложения.

```
class DbManager {

    //database name
    var dbName = "MyNotes"
    //table name
    var dbTable = "Notes"
    //columns
    var colID = "ID"
```

```
    var colTitle = "Title"
    var colDes = "Description"
    //database version
    var dbVersion = 1

    //CREATE TABLE IF NOT EXISTS MyNotes (ID INTEGER PRIMARY KEY,title TEXT,
Description TEXT);"
    val sqlCreateTable = "CREATE TABLE IF NOT EXISTS " + dbTable + " (" + colID +
" INTEGER PRIMARY KEY," + colTitle + " TEXT, " + colDes + " TEXT);"

    var sqlDB: SQLiteDatabase? = null

    constructor(context: Context) {
        var db = DatabaseHelperNotes(context)
        sqlDB = db.writableDatabase
    }

    inner class DatabaseHelperNotes : SQLiteOpenHelper {
        var context: Context? = null

        constructor(context: Context) : super(context, dbName, null, dbVersion) {
            this.context = context
        }

        override fun onCreate(db: SQLiteDatabase?) {
            db!!.execSQL(sqlCreateTable)
            Toast.makeText(this.context, "database created...",
Toast.LENGTH_SHORT).show()
        }

        override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion:
Int) {
            db!!.execSQL("Drop table if Exists" + dbTable)
        }


    }

    fun insert(values: ContentValues): Long {
        val ID = sqlDB!!.insert(dbTable, "", values)
        return ID
    }

    fun Query(projection: Array<String>, selection: String, selectionArgs:
Array<String>, sorOrder: String): Cursor {
        val qb = SQLiteQueryBuilder();
        qb.tables = dbTable
        val cursor = qb.query(sqlDB, projection, selection, selectionArgs, null,
null, sorOrder)
        return cursor
    }

    fun delete(selection: String, selectionArgs: Array<String>): Int {
        val count = sqlDB!!.delete(dbTable, selection, selectionArgs)
        return count
    }

    fun update(values: ContentValues, selection: String, selectionArgs:
Array<String>): Int {
        val count = sqlDB!!.update(dbTable, values, selection, selectionArgs)
        return count
    }
}
```

MainActivity.kt – класс в котором записан функционал главного экрана приложения.

```
class MainActivity : AppCompatActivity() {

    var listNotes = ArrayList<Note>()

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

        //Load from DB
        LoadQuery("%")
    }

    override fun onResume() {
        super.onResume()
        LoadQuery("%")
    }

    private fun LoadQuery(title: String) {
        var dbManager = DbManager(this)
        val projections = arrayOf("ID", "Title", "Description")
        val selectionArgs = arrayOf(title)
        val cursor = dbManager.Query(projections, "Title like ?", selectionArgs,
"Title")
        listNotes.clear()
        if (cursor.moveToFirst()) {

            do {
                val ID = cursor.getInt(cursor.getColumnIndex("ID"))
                val Title = cursor.getString(cursor.getColumnIndex("Title"))
                val Description =
cursor.getString(cursor.getColumnIndex("Description"))

                listNotes.add(Note(ID, Title, Description))

            } while (cursor.moveToNext())
        }

        //adapter
        var myNotesAdapter = MyNotesAdapter(this, listNotes)
        //set adapter
        notesLv.adapter = myNotesAdapter

        //get total number of tasks from ListView
        val total = notesLv.count
        //actionbar
        val mActionBar = supportActionBar
        if (mActionBar != null) {
            //set to actionbar as subtitle of actionbar
            mActionBar.subtitle = "You have $total note(s) in list..."
        }
    }

    override fun onCreateOptionsMenu(menu: Menu?): Boolean {
        menuInflater.inflate(R.menu.main_menu, menu)

        //searchView
        val sv: SearchView = menu!!.findItem(R.id.app_bar_search).actionView as
SearchView
```

```
        val sm = getSystemService(Context.SEARCH_SERVICE) as SearchManager
        sv.setSearchableInfo(sm.getSearchableInfo(componentName))
        sv.setOnQueryTextListener(object : SearchView.OnQueryTextListener {
            override fun onQueryTextSubmit(query: String?): Boolean {
                LoadQuery("%" + query + "%")
                return false
            }

            override fun onQueryTextChange(newText: String?): Boolean {
                LoadQuery("%" + newText + "%")
                return false
            }
        });

        return super.onCreateOptionsMenu(menu)
    }

    override fun onOptionsItemSelected(item: MenuItem?): Boolean {
        if (item != null) {
            when (item.itemId) {
                R.id.addNote -> {
                    startActivity(Intent(this, AddNoteActivity::class.java))
                }
            }
        }
        return super.onOptionsItemSelected(item)
    }

    inner class MyNotesAdapter : BaseAdapter {
        var listNotesAdapter = ArrayList<Note>()
        var context: Context? = null

        constructor(context: Context, listNotesAdapter: ArrayList<Note>) : super()
{
            this.listNotesAdapter = listNotesAdapter
            this.context = context
        }


        override fun getView(position: Int, convertView: View?, parent:
ViewGroup?): View {
            //inflate layout row.xml
            var myView = layoutInflater.inflate(R.layout.row, null)
            val myNote = listNotesAdapter[position]
            myView.titleTv.text = myNote.nodeName
            myView.descTv.text = myNote.nodeDes
            //delete button click
            myView.deleteBtn.setOnClickListener {
                var dbManager = DbManager(this.context!!)
                val selectionArgs = arrayOf(myNote.nodeID.toString())
                dbManager.delete("ID=?", selectionArgs)
                LoadQuery("%")
            }
            //edit//update button click
            myView.setOnClickListener { GoToUpdateFun(myNote) }
            //copy btn click
            myView.copyBtn.setOnClickListener {
                //get title
                val title = myView.titleTv.text.toString()
                //get description
                val desc = myView.descTv.text.toString()
                //concatinate
                val s = title + "\n" + desc
```

```
                val cb = getSystemService(Context.CLIPBOARD_SERVICE) as
ClipboardManager
                cb.text = s // add to clipboard
                Toast.makeText(this@MainActivity, "Copied...",
Toast.LENGTH_SHORT).show()
            }
            //share btn click
            myView.shareBtn.setOnClickListener {
                //get title
                val title = myView.titleTv.text.toString()
                //get description
                val desc = myView.descTv.text.toString()
                //concatenate
                val s = title + "\n" + desc
                //share intent
                val shareIntent = Intent()
                shareIntent.action = Intent.ACTION_SEND
                shareIntent.type = "text/plain"
                shareIntent.putExtra(Intent.EXTRA_TEXT, s)
                startActivity(Intent.createChooser(shareIntent, s))
            }

            return myView
        }

        override fun getItem(position: Int): Any {
            return listNotesAdapter[position]
        }

        override fun getItemId(position: Int): Long {
            return position.toLong()
        }

        override fun getCount(): Int {
            return listNotesAdapter.size
        }

    }

    private fun GoToUpdateFun(myNote: Note) {
        var intent = Intent(this, AddNoteActivity::class.java)
        intent.putExtra("ID", myNote.nodeID) //put id
        intent.putExtra("name", myNote.nodeName) //ut name
        intent.putExtra("des", myNote.nodeDes) //put description
        startActivity(intent) //start activity
    }
}
```

AddNoteActivity.kt – класс с функционалом на добавление новой заметки.

```
class AddNoteActivity : AppCompatActivity() {

    val dbTable = "Notes"
    var id = 0

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

        try {
            val bundle:Bundle = intent.extras
            id = bundle.getInt("ID", 0)
```

```
            if (id!=0){
                titleEt.setText(bundle.getString("name"))
                descEt.setText(bundle.getString("des"))
            }
        }catch (ex:Exception){}
    }

    fun addFunc(view: View){
        var dbManager = DbManager(this)

        var values = ContentValues()
        values.put("Title", titleEt.text.toString())
        values.put("Description", descEt.text.toString())

        if (id ==0){
            val ID = dbManager.insert(values)
            if (ID>0){
                Toast.makeText(this, "Note is added", Toast.LENGTH_SHORT).show()
                finish()
            }
            else{
                Toast.makeText(this, "Error adding note...",
Toast.LENGTH_SHORT).show()
            }
        }
        else{
            var selectionArgs = arrayOf(id.toString())
            val ID = dbManager.update(values, "ID=?", selectionArgs)
            if (ID>0){
                Toast.makeText(this, "Note is added", Toast.LENGTH_SHORT).show()
                finish()
            }
            else{
                Toast.makeText(this, "Error adding note...",
Toast.LENGTH_SHORT).show()
            }
        }
    }
}
```

Как можно заметить в коде классов встречаются комментарии на Английском языке, они написаны для удобства и понимания в коде приложения. Также следует помнить в процессе написания кода подключать библиотеки («Alt» + «Enter» -> Import), это так же будет подсказывать сама среда разработки.

Если всё выполнено правильно, то у получится приложение как на скриншоте.