МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

РАБОТА ПРОВЕРЕНА

Рецензент
Руководитель ИП «Бокарева С.А.
(фирма «Альфа-софт»)»

_____ С.А. Бокарева

“___”_____ 2020 г.

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой,
д.ф.-м.н., профессор

_____ Л.Б. Соколинский

“___”_____ 2020 г.

Разработка мобильного приложения для подсчета количества затраченного пластика для печати на 3D принтере.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 02.03.02.2020.308-420.ВКР

Научный руководитель,
к.ф.-м.н., доцент
_____А.Т. Латипова

Автор работы,
студент группы КЭ-401
_____А.С. Марченяков

Ученый секретарь
(нормоконтролер)
_____ И.Д. Володченко
“___”_____ 2020 г.

Челябинск-2020

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
09.02.2020

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра
студенту группы КЭ-401
Марченякову Андрею Сергеевичу,
обучающемуся по направлению
02.03.02 «Фундаментальная информатика и информационные технологии»

1. **Тема работы** (утверждена приказом ректора от 24.04.2020 № 627)
Разработка мобильного приложения для подсчета количества затраченного пластика для печати на 3D принтере.
2. **Срок сдачи студентом законченной работы:** 05.06.2020.
3. **Исходные данные к работе**
3.1. James S. Cho The Beginner’s Guide to Android Game Development,2014
3.2. Antonio Leiva Kotlin for Android Developers. – CreateSpace Independent Publishing Platform, 2016
3.3. Филипс Б., Стюарт К., Марсикано К. Android программирование для профессионалов. 3-е изд. // СПб.:Питер. - 2017
4. **Перечень подлежащих разработке вопросов**
4.1. Выполнить анализ предметной области.
4.2. Спроектировать мобильное приложение.
4.3. Реализовать мобильное приложение.
4.4. Провести тестирование.
5. **Дата выдачи задания:** 08.02.2020.

Научный руководитель
к.ф.-м.н., доцент кафедры СП А.Т. Латипова

Задание принял к исполнению А.С. Марченяков

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность темы работы

С появлением и повсеместным распространением 3D-принтеров, выявились проблемы, связанные напрямую с видом используемого пластика для печати. Основными проблемами являлись физические свойства используемых видов пластиков, разные виды пластика созданы из различных материалов, представляющих собой как нефтяные фракции, так и органический пластик [9,10].

При расчете расхода пластиков для 3D-печати специалистам требуется учитывать следующие параметры.

1. Толщина слоя печати.

2. Различия в температурных режимах печати.

3. Усадка пластика при остывании.

Рассматривая приведенные выше аспекты 3D-печати, можно прийти к выводу, что для различных видов пластика на одинаковые по размеру фигуры будет затрачен разный объем материала. В общем, на расход материала для напечатанной на 3D-принтере модели влияет огромное количество разнообразных случайных факторов, но даже учесть хотя бы несколько из них является трудоемкой задачей. Величина коэффициента усадки также зависит от большого числа факторов, поэтому как правило, коэффициент усадки для используемого вида материала, настроек принтера и геометрии модели определяют печатью пробного экземпляра. Поставщики пластиков для 3D-печати в документации зачастую указывают примерные значения коэффициента усадки для разных видов материалов или дают примерную формулу для его подсчета, но даже при этом коэффициент усадки в большинстве является эмпирической переменной, поэтому учет коэффициента усадки является актуальной задачей[11,16].

Цели и задачи

Целью данной работы является разработка мобильного приложения для подсчета количества затраченного пластика для печати на 3D принтере с учетом коэффициента усадки.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Провести анализ требований;
2. Провести обзор аналогов;
3. Реализовать мобильное приложение;
4. Протестировать мобильное приложение;

Объем и структура работы

Работа состоит из введения, трех глав, заключения и библиографии. Объем работы составляет 37 страницы, объем библиографии – 16 источников, приложение – 1 .

Краткое содержание работы

Во введении описаны актуальность исследуемой темы, цели и задачи исследования, структура и оббьем работы.

В первой главе рассматриваются различные виды пластиков, а также их физические свойства, рассматриваются существующие аналоги разрабатываемого приложения.

Во второй главе описаны функциональные и нефункциональные требования к приложению, рассмотрены варианты использования мобильного приложения, а также описаны его компоненты и архитектура.

В третьей главе описаны подробности реализации мобильного приложения.

В четвертой главе приведены результаты тестирования системы.

В заключении сделаны выводы о проделанной работе.

1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 Виды пластика

Для печати на 3D принтере существует множество различных видов пластика. Для того чтобы спроектировать приложение и понять, с чем мы будем работать, необходимо выделить основные материалы-пластики для печати, рассмотреть их физические свойства, а затем на основе этих данных создать формулы для реализации подсчета затраченного пластика[6,7,8,15].

1.2 Обзор существующих видов пластика

На данный момент существует широкое разнообразие пластиков для печати, рассмотрим самые популярные, легкодоступные и безопасные виды. Для выбранных видов пластиков представим их физические свойства:

1. PLA (Полилактид) - биоразлагаемый пластик, в основе которого находится молочная кислота. Производится из сахарного тростника или кукурузы. Может также производиться из других натуральных продуктов, таких как картофельный крахмал или целлюлоза

2. ABS (акрилонитрилбутадиенстирол) - ударопрочный пластик, очень популярен в промышленности и 3D-печати. Изделия из ABS достаточно прочны, поэтому его часто используют для печати функциональных объектов, имеющих практическое применение.

3. HIPS (высокопрочный полистирол) - достаточно мягкий пластик, создавался для использования совместно с ABS, для поддержек при двуэкструдерной 3D-печати. Этому способствовали его следующие свойства: одинаковая с ABS температура экструзии, низкая спекаемость с ABS, наличие растворителя (D-Limonene), который растворяет HIPS и не растворяет ABS.

4. PETG (полиэтилентерефталат-гликоль) - относительно новый, по сравнению с тем же ABS, материал, но уже завоевавший заслуженное признание у 3D-печатников. Пластик достаточно ударопрочный, а

спекаемость слоев получается такой, что при нагрузке изделие часто ломается против слоев, а не вдоль.

5. SBS (стиролбутадиен–стирол) - еще один из относительно новых игроков на рынке пластиков для 3D-печати. Характеризуется низкой токсичностью и усадкой, а также высокой прочностью. Основное его преимущество в его прозрачности. Изделия, напечатанные этим пластиком и обработанные сольвентом, приобретают прозрачность окрашенного стекла.

6. Flex (полиуретан) - мягкий резиноподобный материал. Используется там, где нужна гибкость и эластичность готовых изделий.

7. Nylon (нейлон - синтетический материал из семейства полиамидов) — очень стоек к истиранию

8. PC (Поликарбонат) - один из самых крепких материалов в этом списке. Устойчив к физическому и тепловому воздействию. Выдерживает температуру до 110°C [8,9,10,11,13,15].

1.3 Анализ и сравнение аналогов

На данный момент на рынке существует достаточное количество как бесплатных, так и платных программ аналогов, но все они отличаются друг от друга графическим интерфейсом и/или функциональностью. Кроме того существуют программы, предназначенные для принтеров, выпущенными компаниями-производителями данных принтеров. Большинство популярных программ размещены на сайте компании-разработчика (клиент с сайта может загрузить программу на компьютер). Как правило, программы предназначены для моделирования 3D-модели и её печати. К достоинствам таких программ можно отнести [14].

1. Удобный графический интерфейс.

2. Работа с моделью в самой программе.

3. Огромное количество настроек.

К основным недостаткам можно отнести следующие:

1. Для работы данных приложений требуется мощный компьютер, отсутствует мобильная версия приложения.

2. Программы настроены для работы с определенных марок принтеров.

Рассмотрим далее самые популярные программы-аналоги.

MakerWare- это программное обеспечение для управления 3D-принтерами моделей *Replicator* и *Replicator 2X*. Как можно понять, данная программа предназначена только для определенных принтеров, внешний вид и интерфейс представлены на рисунке 1.

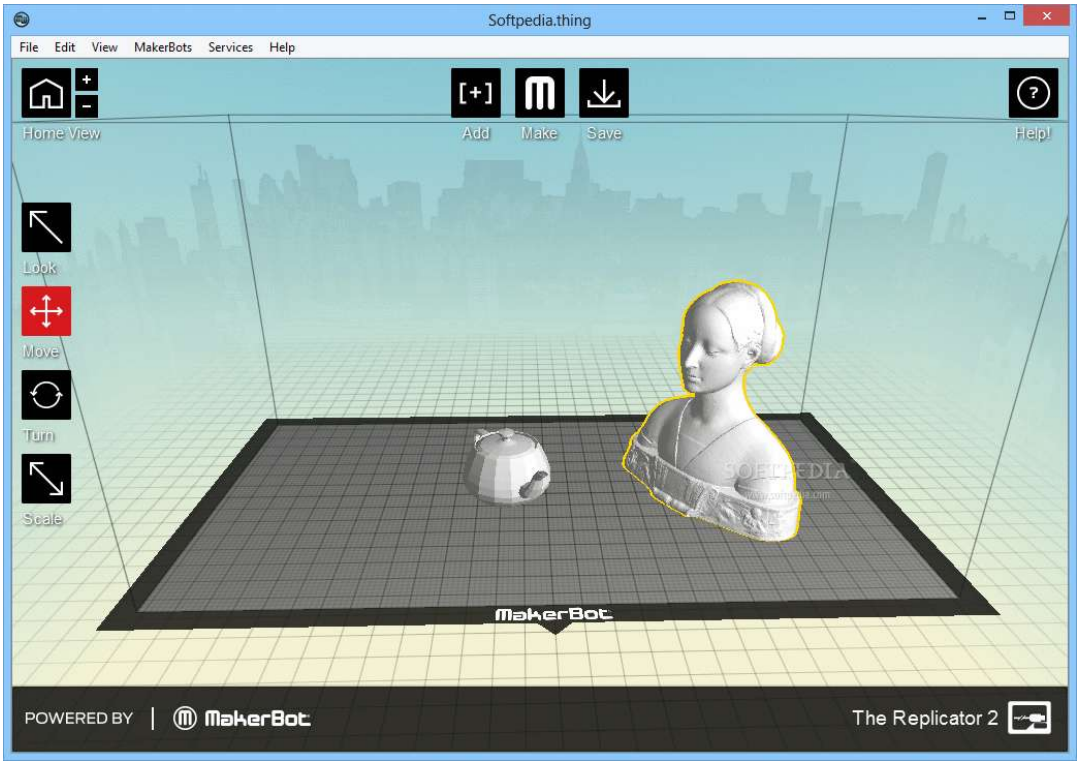

Рис. 1. Интерфейс MakerWare

Другим рассматриваемым приложением является *Cura*. Данная программа разрабатывается для 3D принтеров *Ultimaker*, но при этом совместима и с рядом других видов принтеров для печати [14]. Графический интерфейс данной программы представлены на рисунке 2.

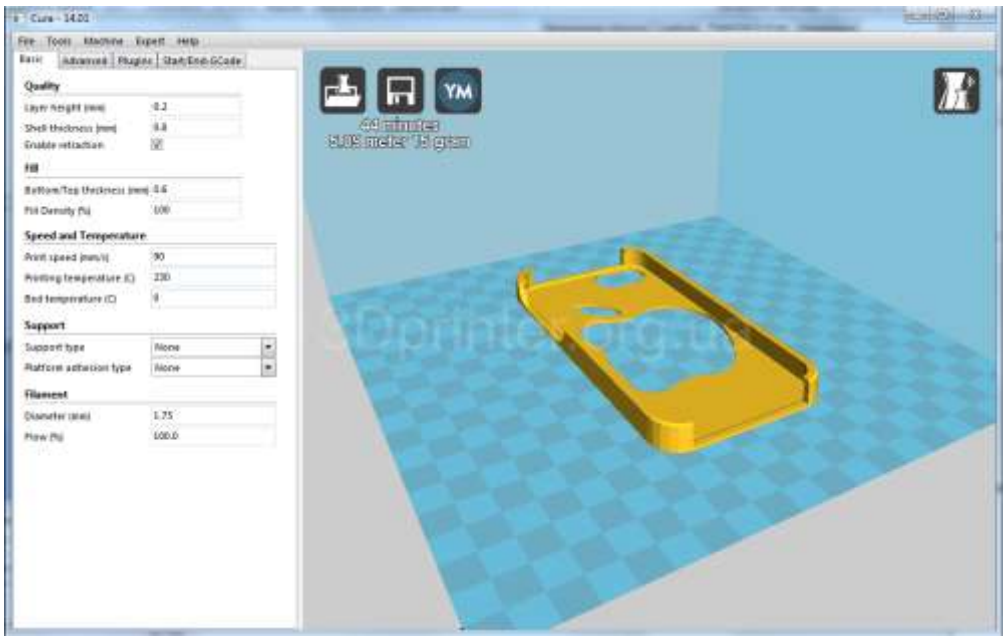

Рис. 2. Интерфейс Cura

Последним рассматриваемым приложением является *Repetier-Host.* Данное приложение по сравнении с вышеперечисленными не обладает современным интерфейсом, простотой и удобством, однако в отличие от остальных оно русифицировано, имеет большее количество настроек, которые позволяют пользователю настраивать абсолютно каждый шаг печати модели. Кроме того, на базе *Repetier* многие компании создают свои собственные приложения для своих принтеров [14]. Интерфейс и внешний вид программы представлен на рисунке 3.

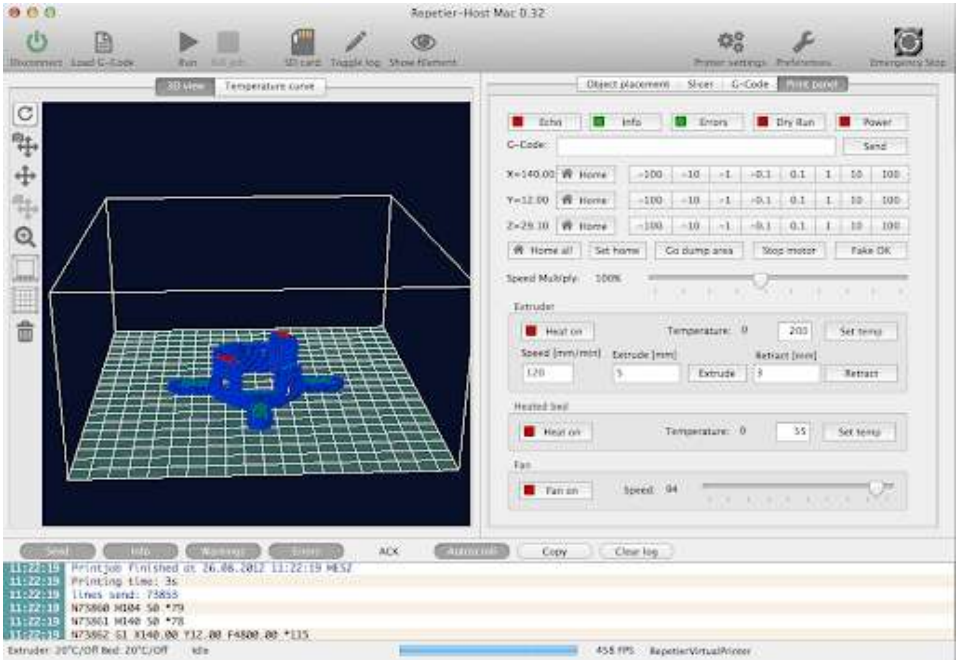

Рис.3. Интерфейс Repetier

Вывод

При рассмотрении аналогов основным недостатком, выделенным мной, является то, что большая часть приложений создана для определенных моделей принтеров и требует мощного персонального компьютера.

Было решено создать приложение для расчета затрат материала для 3D-печати с учетом коэффициента усадки. На финальный расход материала для распечатанной модели объекта влияет огромное количество факторов, однако коэффициент усадки можно определить эмпирически для заданного вида материала. Таким образом, благодаря созданию подобного приложения для телефонов на операционной системе Android, можно автоматизировать трудоемкий процесс расчета расхода полимеров. В создаваемом мною приложении будет реализован расчет затраченного пластика при печати различных стандартных геометрических фигур с учетом коэффициента усадки. С развитием мобильных технологий и увеличением вычислительной мощности телефонов можно разработать мобильное приложение для расчета расхода материала для более сложных 3D-моделей[14,16].

2 ПРОЕКТИРОВАНИЕ МОБИЛЬНОГО ПРИЛОЖЕНИЯ

2.1 Требования к мобильному приложению

В результате анализа предметной области и обзора существующих аналогов были сформированы следующие функциональные и нефункциональные требования к разрабатываемому приложению.

2.2 Функциональные требования

Функциональные требования определяют функциональность программного обеспечения, то есть описывают, что необходимо реализовать в продукте или системе, какие возможности должна предоставлять разрабатываемая система. Функциональные требования определяется заказчиком и должны быть реализованы. Функциональные требования включают в себя, пользовательские требования и бизнес требования. В ходе анализа и проектирования, были выделены следующие функциональные требования:

1) система должна выполнять запуск;

2) система должна отображать пользовательский интерфейс;

3) система должна корректно определять введение значения;

4) система должна рассчитывать количество затраченного материала;

5) система должна производить очистку полей.

2.3 Нефункциональные требования

Нефункциональные требования описывают свойства и ограничения, накладываемые на систему. Нефункциональные требования определяют, как должна работать система или программный продукт, и какими свойствами или характеристиками она должна обладать. Также в отличии от функциональных требований, заказчик не может полностью повлиять на нефункциональные требования. Исходя из условий работы, были сформулированы следующие нефункциональные требования:

1) приложение, должно быть, реализовано на языке Java;

2) приложение должно работать на платформе Android с версии 4.1.

2.4 Диаграмма вариантов использования

Диаграммы вариантов использования применяется для моделирования функциональных требований к программному продукту при его проектировании и разработке.

В диаграммах вариантов использования, определяются вариации использования разрабатываемого приложения. Помимо выявления функциональных и нефункциональных требований была создана диаграмма вариантов использования, представленная на рисунке 4. Данная диаграмма вариантов использования отражает отношение между актерами и прецедентами.

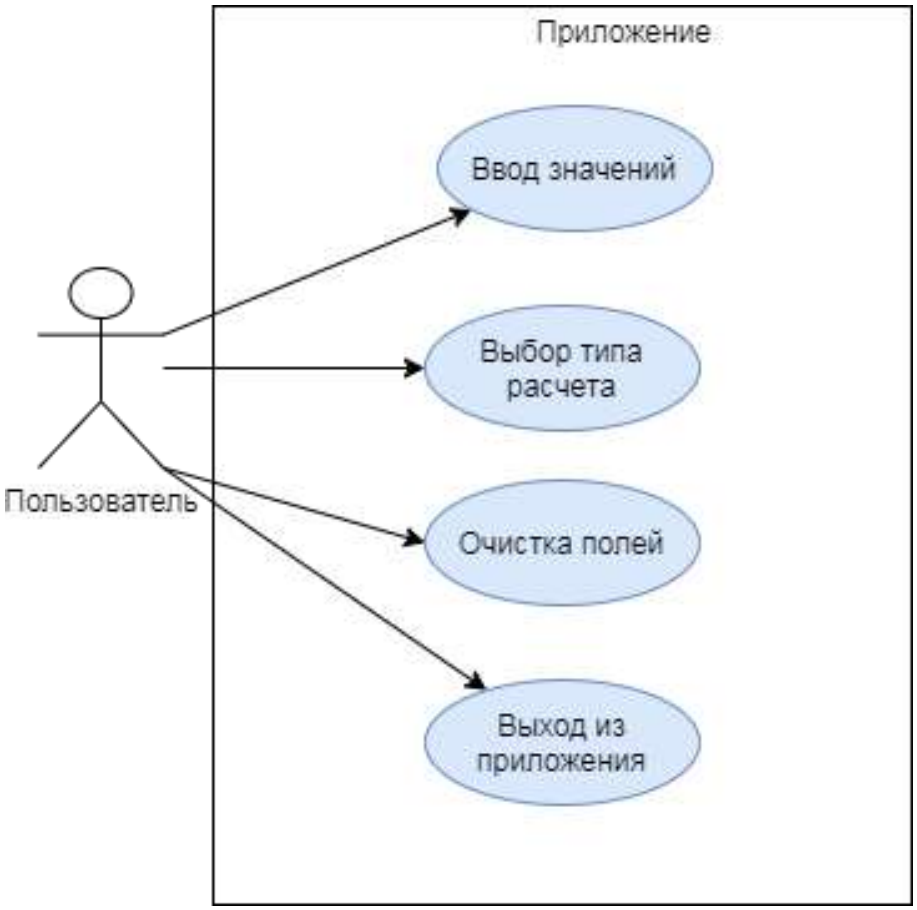

Рис.4. Диаграмма вариантов использования.

Таким образом, на представленной диаграмме у нас представлен один актер, именуемый *пользователь*.

Пользователь может взаимодействовать с приложением следующими способами: Ввести значения переменных, выбрать тип

расчета, который нужен пользователю, очистить поля и выйти из приложения.

Описание вариантов использования системы:

1) *ввод значений* – ввести значения в выделенные поля;
2) *выбор типа расчета* – выбрать один из типов расчета;
3) *очистка полей* – очистка полей для значений, а также очистка поля с результатом работы программы;
4) *выход из приложения* – выйти из приложения.

Вывод

Таким образом, на основе проделанной работы по изучению теоретической части, составления списка требований к приложению была спроектирована архитектура приложения. Были выделены функциональные требования, которые должны быть реализованы в программе, а также нефункциональные требования. Было проведено проектирование системы, на результаты которого будут использованы в процессе реализации и тестирования программы.

3 РЕАЛИЗАЦИЯ МОБИЛЬНОГО ПРИЛОЖЕНИЯ

В ходе реализации мобильного приложения, на основе определенных ранее требований, была создана следующая последовательность реализации мобильного приложения:

1) создание приложение на основе Android с помощью Android Studio, настройка компонентов Android Studio;

2) разработка формул для подсчета затраченного материала;

3) создание и работа с xml файлами, для графического вывода полученных данных.

3.1 Средства реализации

Для реализации приложения на платформе Android была выбрана среда разработки Android Studio, так как данная среда разработки позволяет не только работать с функциональным кодом, но и реализовывать графический интерфейс программы, который можно изменять, как нам нужно, в самой среде разработки. В качестве языкоа программирования был выбран Java, так как он является мультиплатформенным языком программирования, что позволяет создавать приложения как для персональных компьютеров, так и для других устройств, к которым относятся телефоны, часы, планшеты и телевизоры. Для реализации нашего приложения был установлен эмулятор *Genymotion* и виртуальная машина *OracleVm*. Эмулятор нужен для тестирования и запуска приложения, а виртуальная машина требуется для запуска и работы эмулятора [1,2,3,4,5,6].

3.2 Реализация компонентов мобильного приложения

Приступив к реализации мобильного приложения, была изучена литература по данной тематике и в процессе была выведена формула, для подсчета затраченного пластика, формула представляет собой следующее:

$V_{расх} = \frac{V_{дет}}{(1-k)}$, где $V_{расх}$ – количество затраченного материала, $V_{дет}$ – количество материала затраченного только для печати, без учета затрат на усадку, k – коэффициент усадки материала[8,11].

У разных видов материалов различные значения коэффициента усадки, поэтому в приложении была заведена переменная, хранящая величину 1-k для каждого материала (в приложении применялись значения: a = 0.92, b = 0.99, c =0.96, d =0.97, s =0.98). На величину коэффициента усадки влияет большое число случайных факторов, поэтому коэффициент усадки для материала как правило определяют печатью пробного экземпляра [10].

Дальнейшая реализация будет представлена в виде листинга кода программы. На рисунке 5 представлено начало исходного блока с функциональной частью: инициализация полей с водимыми данными и инициализация кнопок.

```
          public class MainActivity extends AppCompatActivity implements
View.OnClickListener {
          final int MENU_RESET_ID = 1;
          final int MENU_QUIT_ID = 2;
          EditText etNum1;
          EditText etNum2;
          EditText etNum3;
          Button btnAdd;
          Button btnSub;
          Button btnMult;
          Button btnDiv;
          Button btnParEm;
          Button btnCilEm;
          Button btnConEm;
          Button btnSqEm;
          Button btnNylon;
          Button btnParEm;
          Button btnCilEm;
          Button btnConEm;
          Button btnSqEm;
```

Рис.5. Инициализация полей и кнопок.

В этой части кода инициализируем создание объектов в виде трех полей для ввода значений, а также инициализируем создание двенадцати функциональных кнопок, к которым относятся кнопки для просто подсчета затраченного материала по введенным значениям, и кнопки, позволяющие рассчитать количество затраченного материала для создания стандартных геометрических фигур.

На рисунке 6 представлен класс, отвечающий за определение существования введенных ранее кнопок и полей.

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    etNum1 = (EditText) findViewById(R.id.etNum1);
    etNum2 = (EditText) findViewById(R.id.etNum2);
    etNum3 = (EditText) findViewById(R.id.etNum3);
    btnAdd = (Button) findViewById(R.id.btnAdd);
    btnSub = (Button) findViewById(R.id.btnSub);
    btnMult = (Button) findViewById(R.id.btnMult);
    btnDiv = (Button) findViewById(R.id.btnDiv);
    btnPetg = (Button) findViewById(R.id.btnPetg) ;
    btnFlex= (Button) findViewById(R.id.btnFlex);
    btnSbs= (Button) findViewById(R.id.btnSbs);
    btnNylon= (Button) findViewById(R.id.btnNylon);
    btnSqEm = (Button)findViewById(R.id.btnSqEm) ;
    btnConEm =(Button)findViewById(R.id.btnConEm);
    btnCilEm =(Button)findViewById(R.id.btnCilEm);
    btnParEm =(Button)findViewById(R.id.btnParEm);

    tvResult = (TextView) findViewById(R.id.tvResult);
```

Рис.6. Определение полей и кнопок.

На рисунке 6, помимо определения кнопок и полей, ранее проинициализированных, представлена инициализация поля вывода, называемого *tvResult*, куда в дальнейшем будет выводиться результат.

На рисунке 7 у нас приведена функция обработчика при помощи, которой программа проверяет поля для ввода значений на пустоту.

```
if (TextUtils.isEmpty(etNum1.getText().toString())
                || TextUtils.isEmpty(etNum2.getText().toString())
                || TextUtils.isEmpty(etNum3.getText().toString()))
            return;
```

Рис.7. Функция обработчик полей.

Благодаря функции на рисунке 7 программа не будет считать значения, когда поля для значений пустые.

Для обработки событий по нажатию кнопок, были созданы функции, представленные на рисунке 8. После этого приступим к присвоению функционала данным кнопкам.

```
btnAdd.setOnClickListener(this);
btnSub.setOnClickListener(this);
btnMult.setOnClickListener(this);
btnDiv.setOnClickListener(this);
btnPetg.setOnClickListener(this);
btnFlex.setOnClickListener(this);
btnSbs.setOnClickListener(this);
btnNylon.setOnClickListener(this);
btnSqEm.setOnClickListener(this);
btnConEm.setOnClickListener(this);
btnCilEm.setOnClickListener(this);
btnParEm.setOnClickListener(this);
```

Рис.8. Обработка нажатий на кнопки.

На рисунке 9 представлена функция для чтения полей, в которые вводятся числа, а также присваивание числовых значений полей переменным.

```
num1 = Float.parseFloat(etNum1.getText().toString());
num2 = Float.parseFloat(etNum2.getText().toString());
num3 = Float.parseFloat(etNum3.getText().toString());
```

Рис.9. Функция чтения полей

Функции обработчика клика на кнопки представлена в приложении 1 .

При нажатии на кнопки расчета для заданной геометрической фигуры, применяется формула либо для куба $V = a^3$, где a – это длина грани; либо для параллелепипеда $V = a * b * h$, где a – длина, b – ширина, h – высота; либо для конуса $V = \pi R^2 h$, где π – константа, равная 3.141592, R – радиус основания, h – высота; либо для цилиндра $V = \frac{1}{3}\pi R^2 h$, где π – константа равная 3.141592, R – радиус основания, h – высота. Данные кнопки показаны на рисунке 10.

```
                case R.id.btnSqEm:
                    oper = "Куб";
                    result = (float) ((num1*num1)*f);
                    for (;i < 1;)
                        i = result;
                    break;
                case R.id.btnConEm:
                    oper = "Конус";
                    result =(float) (p*num1*(num1*num2));
                    for (;i < 1;)
                        i = result;
                    break;
                case R.id.btnCilEm:
                    oper = "Цилл";
                    result =(float) (j*p*num1*(num1+num3));
                    for (;i < 1;)
                        i = result;
                    break;
                case  R.id.btnParEm:
                    oper = "Пар";
                    result =(float) (j*(num1*num2+num1*num3+num2*num3));
                    for (;i < 1;)
                        i = result;
                    break;
               default:
                    break;
```

Рис.10. Кнопки для геометрических фигур.

На рисунке 10 указана переменная *i*. Данная переменная используется для учета усадки при расходе материала на геометрические фигуры. Происходит это следующим образом: после подсчета объема фигуры результат параллельно выводу записывается в переменную, далее при нажатии на кнопку с определенным видом материала, происходит расчет расхода для данных материала и фигуры.

Создание меню для кнопок очисти полей и кнопки выхода из приложения на рисунке 11.

```
public boolean onCreateOptionsMenu(Menu menu) {
        menu.add(0, MENU_RESET_ID, 0, "Reset");
        menu.add(0, MENU_QUIT_ID, 0, "Quit");
        return super.onCreateOptionsMenu(menu);
```

Рис.11. Меню и кнопки очистки и выхода

Для обработки нажатий на пункты меню, используются функции представленные на рисунке 12.

```
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case MENU_RESET_ID
})
```

Рис.12. Обработка нажатий на пункты меню.

Функция, представленная на рисунке 13 – это функция, позволяющая производить очистку полей.

```
{
            case MENU_RESET_ID
etNum1.setText("");
                etNum2.setText("");
                etNum3.setText("");
                tvResult.setText("");
                break;
}
```

Рис. 13. Функция для очистки полей.

Функция выхода из приложения на рисунке 14, позволяет выйти из приложения.

```
        case MENU_QUIT_ID:
                    finish();
                    break;
              }
```

Рис.14. Выход из приложения.

Таким образом, на представленных выше рисунках была реализована функциональная часть приложения, находящая в *mainActivity*. Были созданы и инициализированы кнопки приложения; были созданы и реализованы поля для ввода и вывода значений в приложении, а также кнопки выхода из приложения и очистки полей приложения.

3.3 Реализация пользовательского интерфейса

Пользовательский интерфейс в Android Studio был реализован с помощью языка разметок XML, в данном блоке были выполнены: графическое создание полей приложения, создание кнопок приложения и создание кнопок выхода из приложения и очистки полей приложения. Данный блок будет представлен в виде рисунков кода самой программы.[1,2,3,4,5,6]

Для визуальной части используется и язык разметки типа XML, первым шагом для создания приложения станет инициализация рабочего пространства, первый шаг представлен на рисунке 15.

```
        <LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent">
```

Рис.15. Инициализация рабочего пространства.

Следующим шагом становится разделение пространства на блоки, который представлен на рисунке. 16 первый блок это раздел где осуществляется ввод переменных. Для ввода переменных были выделены 3 окна, которые представляют собой переменные X Y Z, эти окна представлены на рисунке 17.

```
<LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/linearLayout1"
        android:layout_marginLeft="10pt"
        android:layout_marginRight="10pt"
        android:layout_marginTop="3pt"
        tools:ignore="ExtraText">
```

Рис.16. Разделение на блоки.

```
<EditText
            android:layout_weight="1"
            android:layout_height="wrap_content"
            android:layout_marginRight="5pt"
            android:id="@+id/etNum1"
            android:layout_width="match_parent"
            android:inputType="numberDecimal">
        </EditText>
        <EditText
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:layout_marginLeft="5pt"
            android:id="@+id/etNum2"
            android:layout_width="match_parent"
            android:inputType="numberDecimal">
        </EditText>
        <EditText
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:layout_marginLeft="5pt"
            android:id="@+id/etNum3"
            android:layout_width="match_parent"
            android:inputType="numberDecimal">
        </EditText> }
```

Рис.17. Выделение 3 вводных пространств.

Следующим блоком является блок инициализации раздела кнопок, для выбора вида пластика для подсчета, данный блок представлен на рис/ 18.

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/linearLayout2"
    android:layout_marginTop="3pt"
    android:layout_marginLeft="5pt"
    android:layout_marginRight="5pt">
```

Рис.18. Блок инициализации кнопок.

Создание первых кнопок осуществляется на рисунке 19. Всего используется 8 кнопок для 8 разных видов пластика. Для удобного отображения в программе данный блок был разбит на 2 одинаковых поля. В ходе изучения теоретической части по данной теме, было выделено, что одним из параметров расхода пластика при печати, который можно эмпирически оценить, является коэффициент усадки, зависящий от вида пластика, поэтому было сделано именно 8 кнопок.

```
        <Button
                android:layout_height="wrap_content"
                android:layout_width="match_parent"
                android:layout_weight="1"
                android:text=" Abs"
                android:textSize="8pt"
                android:id="@+id/btnAdd">
            </Button>
            <Button
                android:layout_height="wrap_content"
                android:layout_width="match_parent"
                android:layout_weight="1"
                android:text=" Pla"
                android:textSize="8pt"
                android:id="@+id/btnSub">
            </Button>
            <Button
                android:layout_height="wrap_content"
                android:layout_width="match_parent"
                android:layout_weight="1"
                android:text="Hips"
                android:textSize="8pt"
                android:id="@+id/btnMult">
            </Button>
            <Button
                android:layout_height="wrap_content"
                android:layout_width="match_parent"
                android:layout_weight="1"
                android:text=" Pc"
                android:textSize="8pt"
                android:id="@+id/btnDiv">
            </Button>
```

Рис.19. Инициализация кнопок 1 поля.

Блок, представленный на рисунке 20, является продолжением создания кнопок (создано еще 4 кнопки для других 4 материалов).

```
    <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/linearLayout3"
            android:layout_marginTop="3pt"
            android:layout_marginLeft="5pt"
            android:layout_marginRight="5pt">
            <Button
                android:layout_height="wrap_content"
                android:layout_width="match_parent"
                android:layout_weight="1"
                android:text="Petg "
                android:textSize="8pt"
                android:id="@+id/btnPetg">
            </Button>
            <Button
                android:layout_height="wrap_content"
                android:layout_width="match_parent"
                android:layout_weight="1"
                android:text="Flex "
                android:textSize="8pt"
                android:id="@+id/btnFlex">
            </Button>
            <Button
                android:layout_height="wrap_content"
                android:layout_width="match_parent"
                android:layout_weight="1"
                android:text="Sbs "
                android:textSize="8pt"
                android:id="@+id/btnSbs">
            </Button>
            <Button
                android:layout_height="wrap_content"
                android:layout_width="match_parent"
                android:layout_weight="1"
                android:text="Nylon "
                android:textSize="8pt"
                android:id="@+id/btnNylon">
            </Button>
        </LinearLayout>
```

Рис.20. Инициализация кнопок 2 поля.

На рисунке 21 представлена инициализация кнопок для расчета расхода материала для 4 стандартных видов геометрических фигур.

```
<LinearLayout
        android:id="@+id/linearLayout3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="10pt"
        android:layout_marginTop="4pt"
        android:layout_marginRight="10pt"
        >
    <Button
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:layout_weight="1"
        android:text="Пар"
        android:textSize="8pt"
        android:id="@+id/btnParEm">
</Button>
        <Button
            android:layout_height="wrap_content"
            android:layout_width="match_parent"
            android:layout_weight="1"
            android:text="Цилл"
            android:textSize="8pt"
            android:id="@+id/btnCilEm">
        </Button>
        <Button
            android:layout_height="wrap_content"
            android:layout_width="match_parent"
            android:layout_weight="1"
            android:text="Кон"
            android:textSize="8pt"
            android:id="@+id/btnConEm">
        </Button>
        <Button
            android:layout_height="wrap_content"
            android:layout_width="match_parent"
            android:layout_weight="1"
            android:text="Куб"
            android:textSize="8pt"
            android:id="@+id/btnSqEm">
        </Button>
```

Рис.21. Инициализация кнопок для геометрических фигур.

На рисунке 22 представлена инициализация и размеры зоны вывода результата подсчетов.

```
        <TextView
            android:id="@+id/tvResult"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="5pt"
           android:layout_marginTop="3pt"
            android:layout_marginRight="5pt"
            android:gravity="center_horizontal"
            android:textSize="12pt"></TextView>
    </LinearLayout>
```

Рис.22. Инициализация зоны вывода.

На представленных выше листингах показана настройка визуальной части разработанного приложения с применением языка разметки XML.

Вывод

В результате проделанной работы на основе разработанной проектной модели и выявленных требований к приложению были реализованы как визуальные инициализации кнопок и полей приложения, так и функциональные.

4 ТЕСТИРОВАНИЕ

Тестирование системы состоит из функционального тестирования мобильного приложения. Было проведено функциональное тестирование приложения, которое направлено на проверку реализуемости функциональных требований.

4.1 Функциональное тестирование

Результаты функционального тестирования представлены в таблице 1.

Табл. 1. Функциональное тестирование

№	Название теста	Ожидаемый результат	Полученный результат	Тест пройден?
1	Отображение интерфейса программы	На экране отображается интерфейс программы со всеми полями и кнопками	На экране отображается интерфейс программы со всеми полями и кнопками	+
2	Работа полей ввода	При вводе в поле появляются вводимые значения	В поле ввода появляются вводимые значения	+
3	Работа кнопок для подсчета	При нажатии на любую из представленных кнопок происходит расчет	При нажатии на кнопки, в поле результата появляется результат работы программы	+
4	Работа кнопки очистки полей приложения	При нажатии на кнопку очистки, очищаются поля ввода и поле результата	При нажатии на кнопку очистки все поля программы очищаются	+
5	Работа кнопки выхода из приложения	При нажатии на кнопку выхода происходит выход из приложения	При нажатии на кнопку, происходит выход из приложения	+

Для проведения функционального тестирования программы на эмуляторе было создано устройства HTC One с уровнем API 26, Pixel с уровнем API 24 и Pixel с уровнем API 29. Результаты представлены в таблице 1.

Результаты тестирования приложения представлены на рисунках ниже.

Рис. 23. Запущенное приложение.

На рисунке 23 представлено запущенное приложение

Рис. 24. Ввод значений в поля.

На рисунке 24 представлен ввод значений в поля приложения.

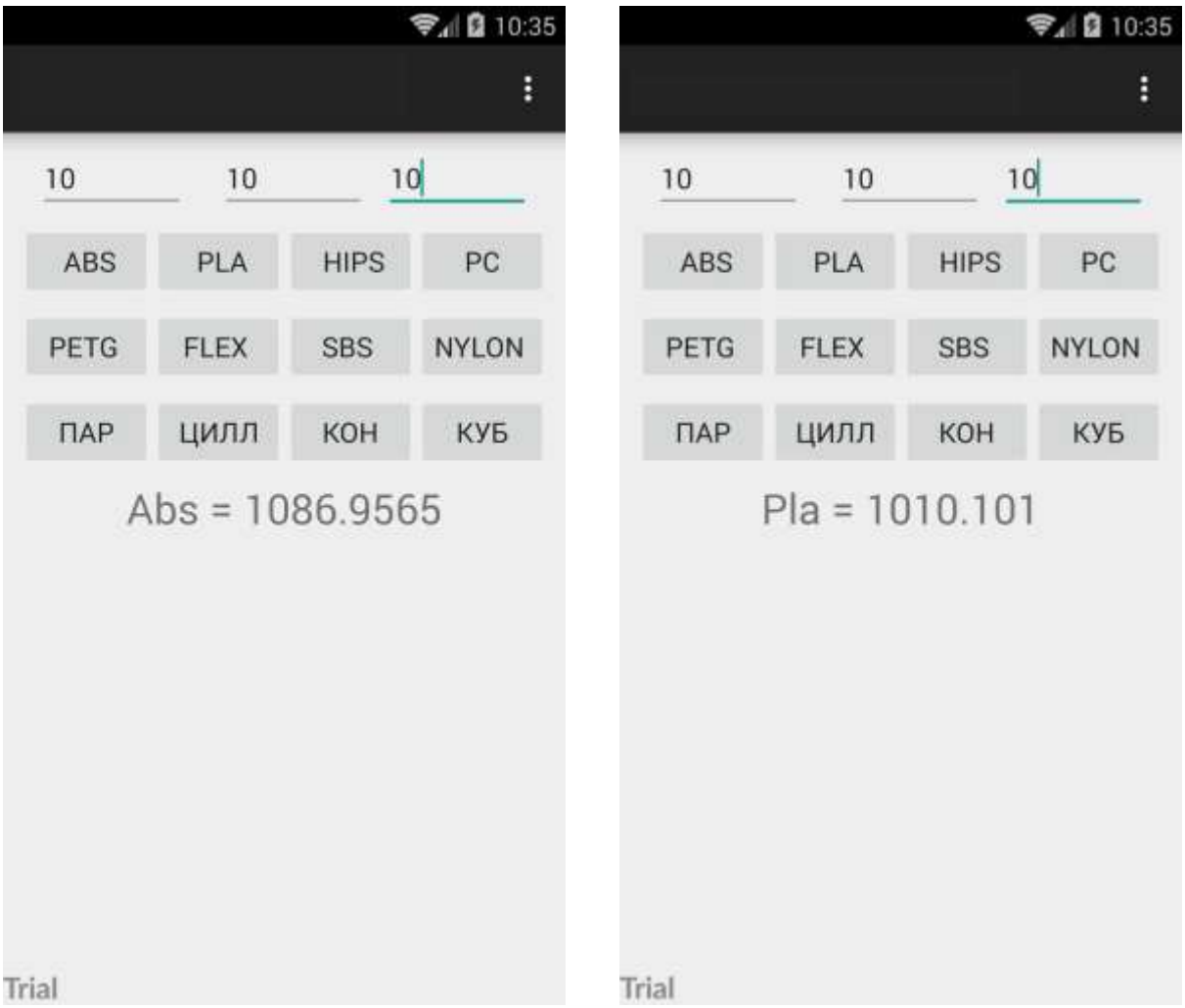

Рис. 25. Работа кнопок приложения.

На рисунке 25 происходит работа первых двух кнопок приложения, чтобы проверить их.

Помимо тестирования основных кнопок было проведено тестирование кнопок очистки поля, данные тестирования представлены на рисунке 26.

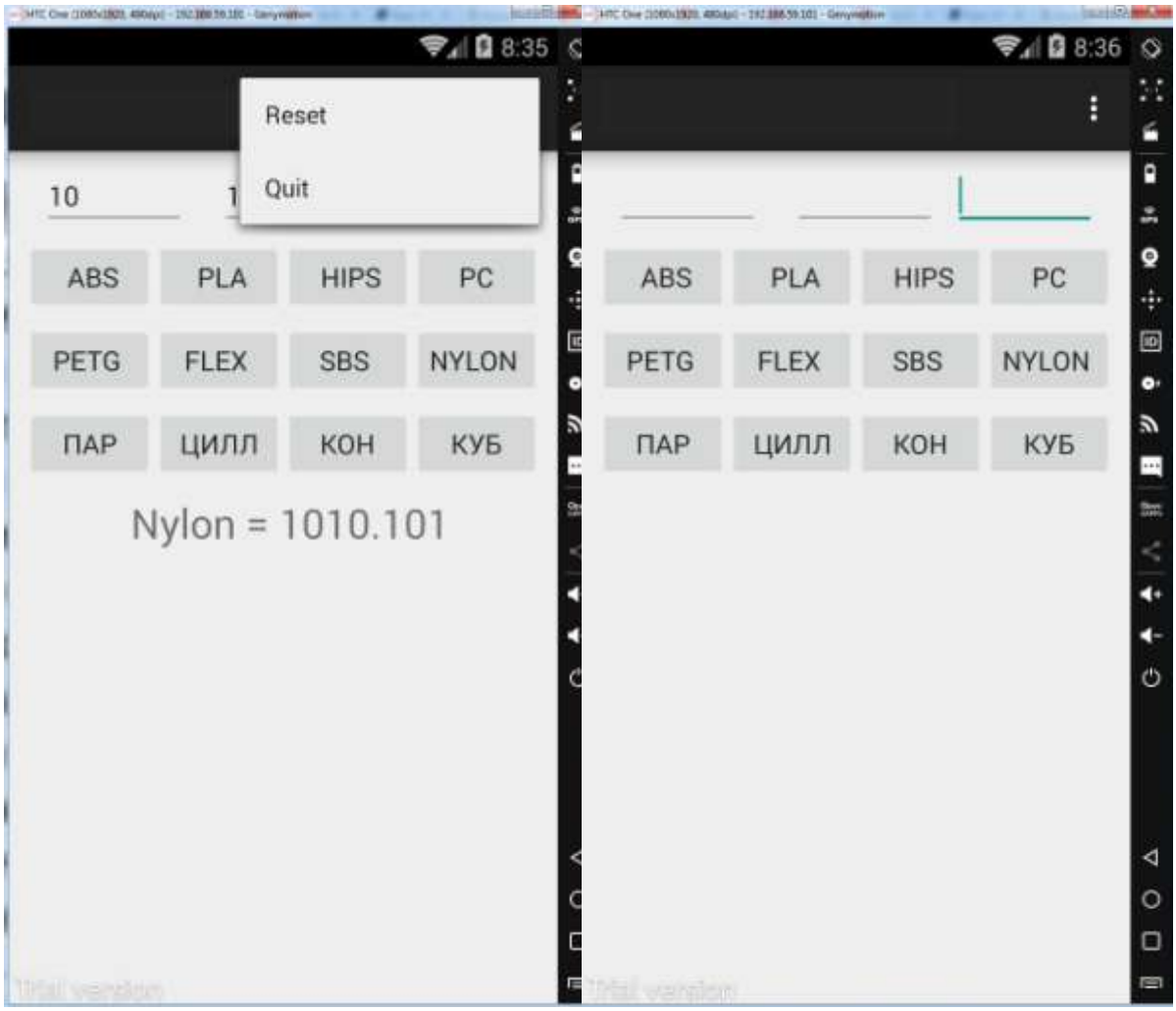

Рис. 26. Открытие меню с кнопками очистки и выхода и очистка полей приложения.

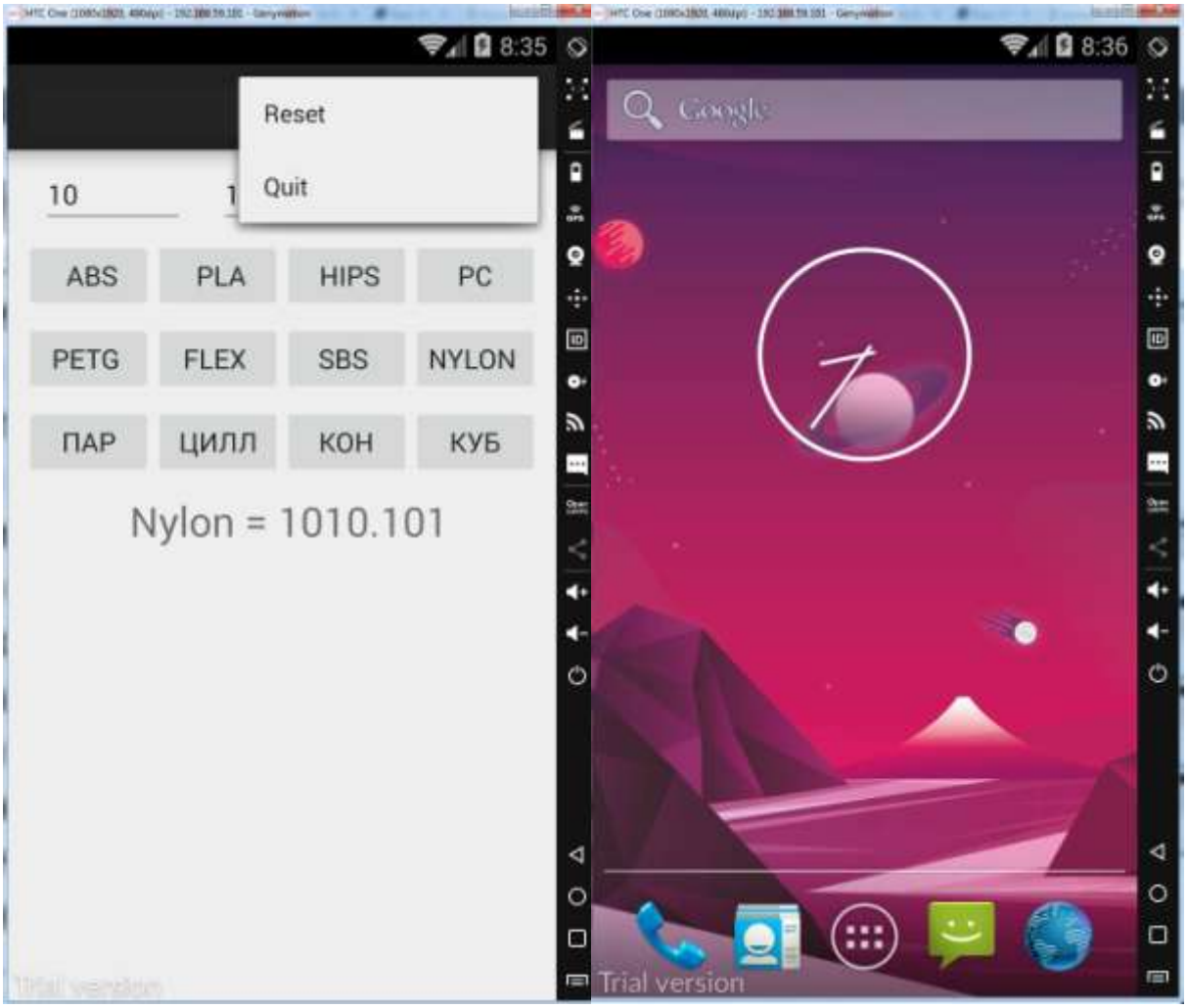

Рис. 27. Выход из приложения.

Так же было проведено тестирование кнопки выхода, результаты представлены на рисунке 27

ВЫВОД

Было произведено тестирование разработанного приложения, тестирование показало, что приложение соответствует поставленным при проектировании задачам.

ЗАКЛЮЧЕНИЕ

В рамках данной работы было разработано мобильного приложения для подсчета количества затраченного пластика для печати на 3D принтере.

При этом были решены следующие задачи:

1) проведен анализ предметной области;

2) определены требования к системе и разработаны варианты ее использования;

3) разработано приложение;

4) приложение было протестировано.

Планируется развитие данного приложения в будущем:

1) добавление функционала для расчета моделей без заполнения;

2) добавление функционала для расчета готовых моделей, созданных на специализированных приложениях для 3D моделирования.

ЛИТЕРАТУРА

1) Медникс З. Программирование под Android. 2-е изд. /Дорнин Л., Мик Б., Накамура М. — СПб.: Питер, 2013. — 560 с.: ил.

2) П. Дейтел. Android для программистов: создаём приложения. /Х. Дейтел, Э. Дейтел — СПб.: Питер, 2013. —560 с.: ил.

3) Харди Б. Android. Программирование для профессионалов. 2-е изд. /Филлипс Б., Стюарт К., Марсикано К. — СПб.: Питер, 2016. — 640 с.: ил.

4) Васильев А.Н. Java. Объектно-ориентированное программирование, 2011. —397 с.

5) James S. Cho The Beginner’s Guide to Android Game Development,2014

6) Antonio Leiva Kotlin for Android Developers. – CreateSpace Independent Publishing Platform, 2016

7) Sean Aranda 3D Printing Failures: 2020 Edition: How to Diagnose and Repair ALL Desktop 3D Printing Issues

8) Расслаивание и усадка. Виды пластика - Тема 2 [Электронный ресурс] URL: https://ru.coursera.org/lecture/additivnye-tekhnologii/rasslaivaniie-i-usadka-vidy-plastika-xrbTR (дата обращения 20.05.2020).

9) 3D печать, 3D моделирование деталей (моделей) из ABS и PLA пластиков [Электронный ресурс] URL: http://infor59.ru/3d_print (дата обращения 20.05.2020).

10) Пластики для 3D печати: сравнение pla, abs, petg, hips, pp, asa, ngen, pc, flex, нейлона и композитов [Электронный ресурс] URL: https://www.qbed.space/knowledge/blog/filament-comparison (дата обращения 28.05.2020).

11) 3D принтер. Как рассчитать расход пластика и время печати. Себестоимость 3D печати [Электронный ресурс] URL: https://3dprinter.ua/raskhod-materialov/ (дата обращения 28.05.2020).

12) Подробный гид по выбору пластика для 3D-печати [Электронный ресурс] URL: https://top3dshop.ru/blog/podrobnyj-gid-po-vyboru-plastika-dlja-3d-pechati.html#start (дата обращения 28.05.2020).

13) Дмитрий Горьков, 3D-печать с нуля 2015 год 400 с.

14) Программы для 3D печати и 3D принтера [Электронный ресурс] URL: https://make-3d.ru/articles/programmy-dlya-3d-pechati-i-3d-printera/ (дата обращения 30.05.2020).

15) Обзор актуальных материалов для 3D-печати [Электронный ресурс] URL: http://edurobots.ru/2017/01/3d-printing-materials/ (дата обращения 30.05.2020).

16) Технологии 3D-печати [Электронный ресурс] URL: https://www.ixbt.com/printer/3d/3d_tech.shtml (дата обращения 30.05.2020).

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А. Функция обработки кнопок.

```
      switch (v.getId()) {
                  case R.id.btnAdd:
                      oper = "Abs";
                      result = (float) ((num1 * num2 * num3)/a);
                      if ( i>0 )
                      result = (float) (i/a);
                      break;
                  case R.id.btnSub:
                      oper = "Pla";
                      result = (float) ((num1 * num2 * num3)/b);
                      if ( i>0 )
                      result = (float) (i/b);
                      break;
                  case R.id.btnMult:
                      oper = "Hips";
                      result = (float) ((num1 * num2 * num3)/c);
                      if ( i>0 )
                      result = (float) (i/c);
                      break;
                  case R.id.btnDiv:
                      oper = "Pc";
                      result = (float) ((num1 * num2 * num3)/d);
                          if ( i>0 )
                      result =(float) (i/d);
                      break;
                  case R.id.btnPetg:
                      oper="Petg";
                      result = (float) ((num1 * num2 * num3)/b);
                      if ( i>0 )
                      result = (float) (i*b);
                      break;
                  case R.id.btnFlex:
                      oper="Flex";
                      result = (float) ((num1 * num2 * num3)/a);
                      if ( i>0 )
                      result = (float) (i/a);
                      break;
                  case R.id.btnSbs:
                      oper="Sbs";
```

```
                result = (float) ((num1 * num2 * num3)/s);
                if ( i>0 )
                result = (float) (i/s);
                break;
            case R.id.btnNylon:
                oper="Nylon";
                result = (float) ((num1 * num2 * num3)/b);
                if ( i>0 )
                result = (float) (i/b);
                break;
```

Рис. А.1. Функция обработки кнопок.