**Министерство образования и науки Российской Федерации**

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

**Санкт-Петербургский национальный исследовательский университет**

**информационных технологий, механики и оптики**

Факультет Фотоники и оптоинформатики

Кафедра Компьютерной фотоники и видеоинформатики

**Отчет по практике**

|  |  |  |
| --- | --- | --- |
|  | Выполнил:  Скрипачев И.С. |  |
|  | Группа: V3316 |  |
|  | Преподаватель: | Кудрявцев А. С. |

Санкт-Петербург, 2017

Оглавление

[Цель проведения практики 3](#_Toc507346109)

[Задание №1. Знакомство с системой контроля версий Git 4](#_Toc507346110)

[Задание №2. «Форматирование и стиль» 11](#_Toc507346111)

[Задание №3. «Тестирование 14](#_Toc507346112)

[Проектная работа. Приложение-виджет для андроид 18](#_Toc507346113)

[Приложение А. Презентация с семинара по С++11 на тему: Tuple and tie. 23](#_Toc507346114)

# Цель проведения практики

Освоение навыков использования C++ и изучение приемов разработки программного обеспечения. Практика проходит в компьютерном классе и состоит из лекционных занятий и практических заданий.

# Задание №1. Знакомство с системой контроля версий Git

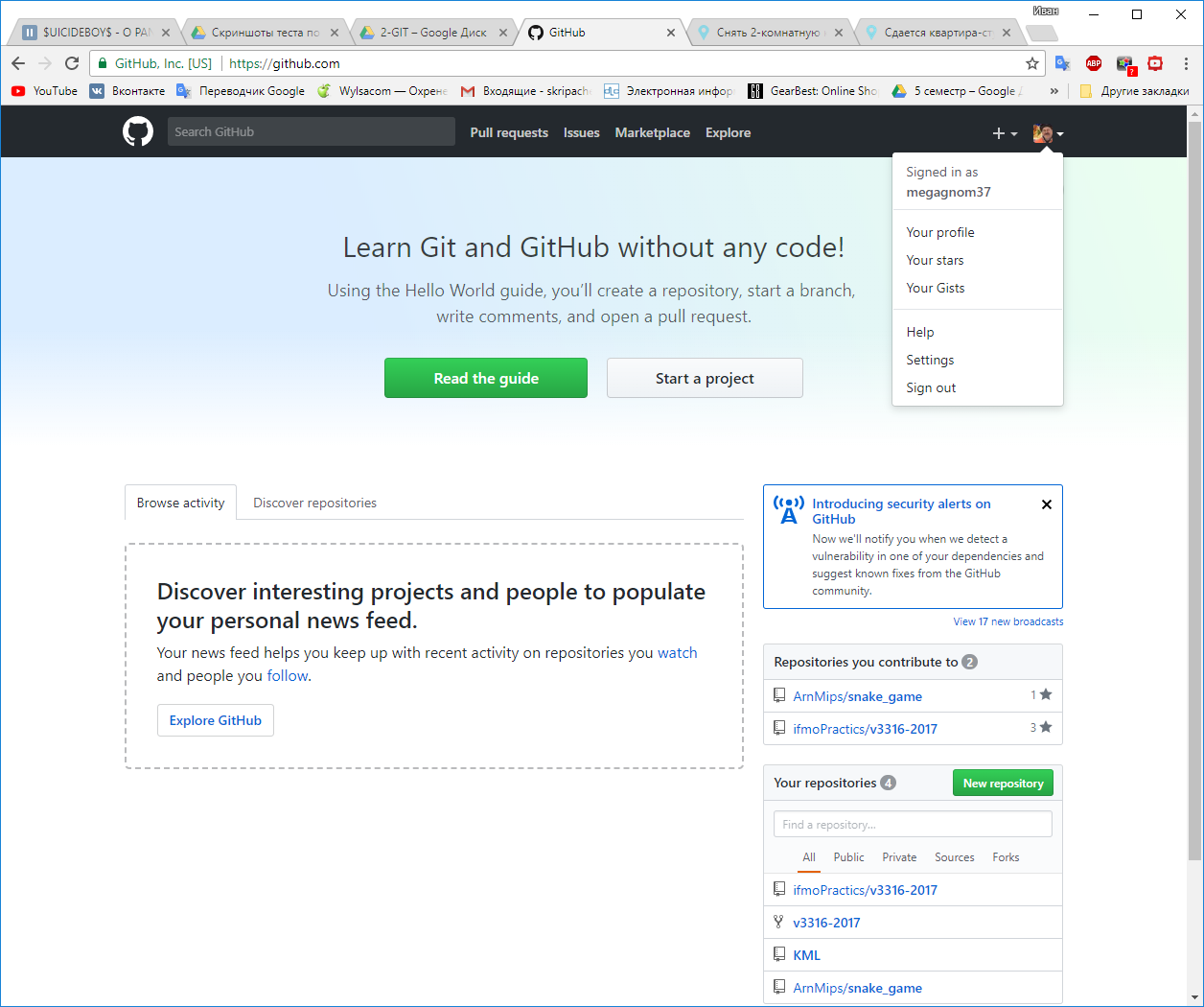
1. Завести аккаунт на github.com. На рис.1 представлен скриншот зарегистрированного пользователя.  
   

Рисунок 1- аккаунт на github.com

1. Создать репозиторий для лабораторных работ. На Рис.2 представлен процесс создания репозитория и его первоначальная настройка. Рис.3 демонстрирует наличие созданного репозитория.

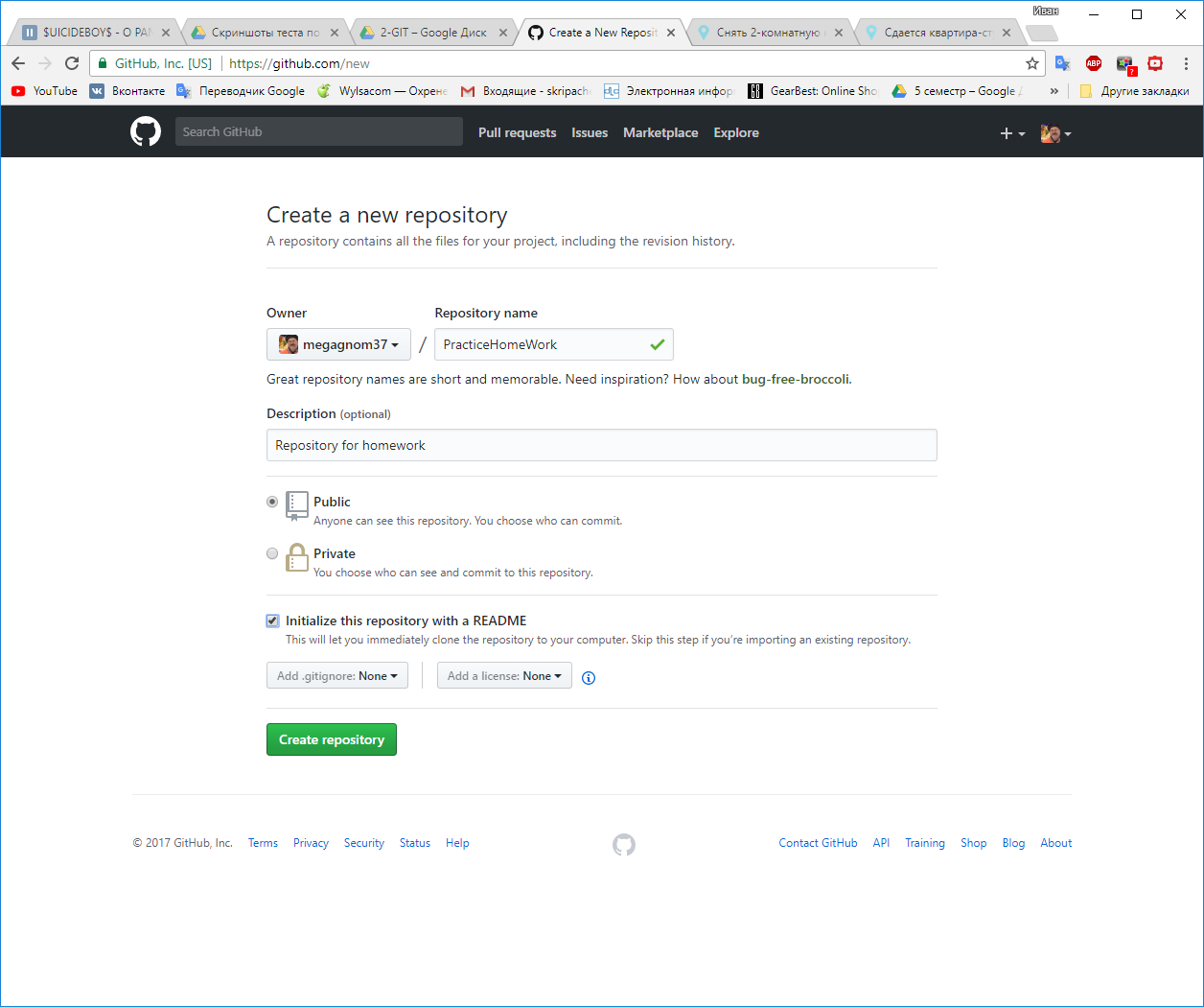


Рисунок 2-создание репозитория

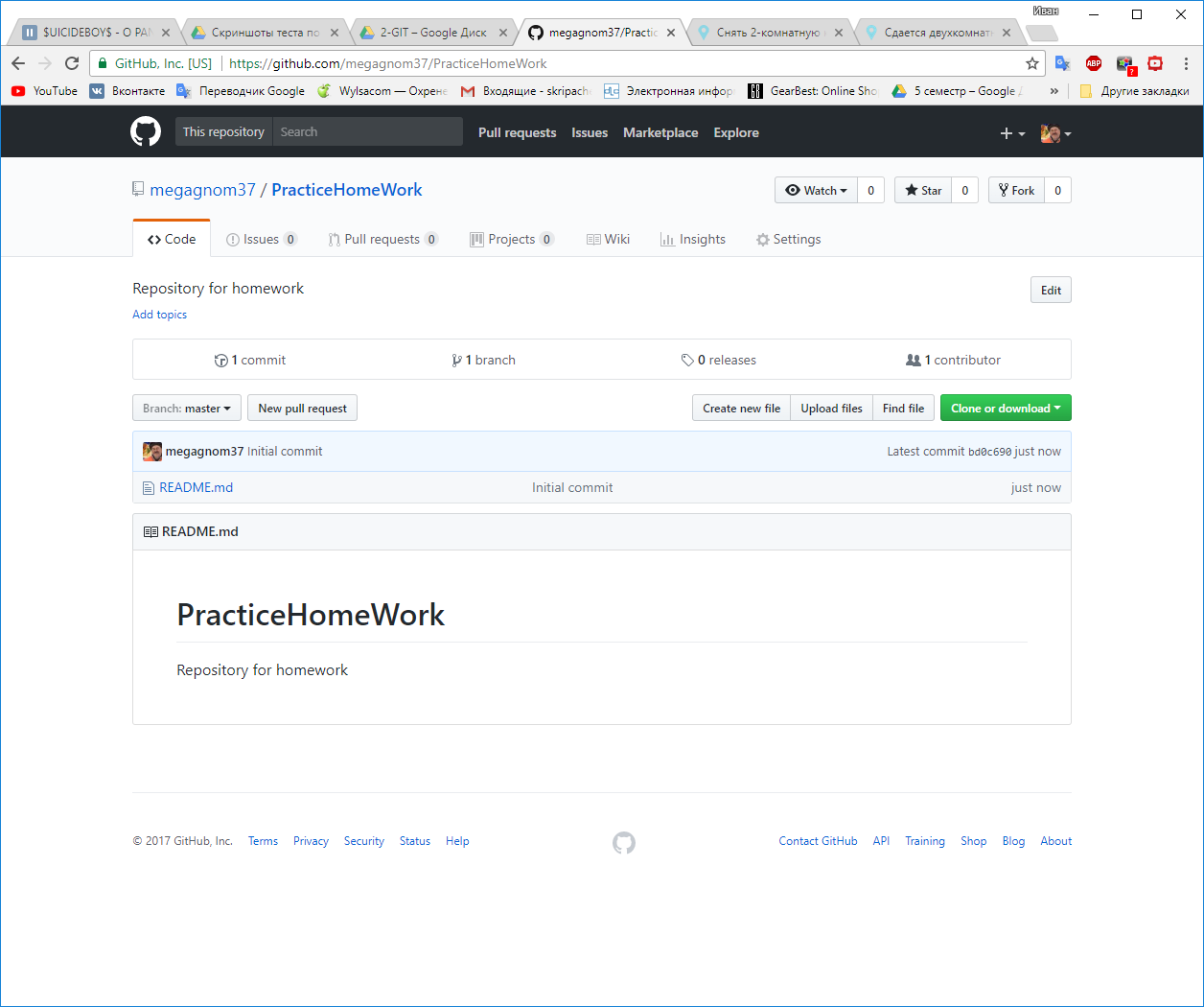


Рисунок 3-скриншот созданного репозитория

1. Склонировать репозиторий на компьютер. На Рис.4 можно видеть, что для клонирования репозитория нужно иметь ссылку на него. Копируем и в консоли с помощью команды git clone <LINK> клонируем репозиторий в нужное нам место Рис. 5.

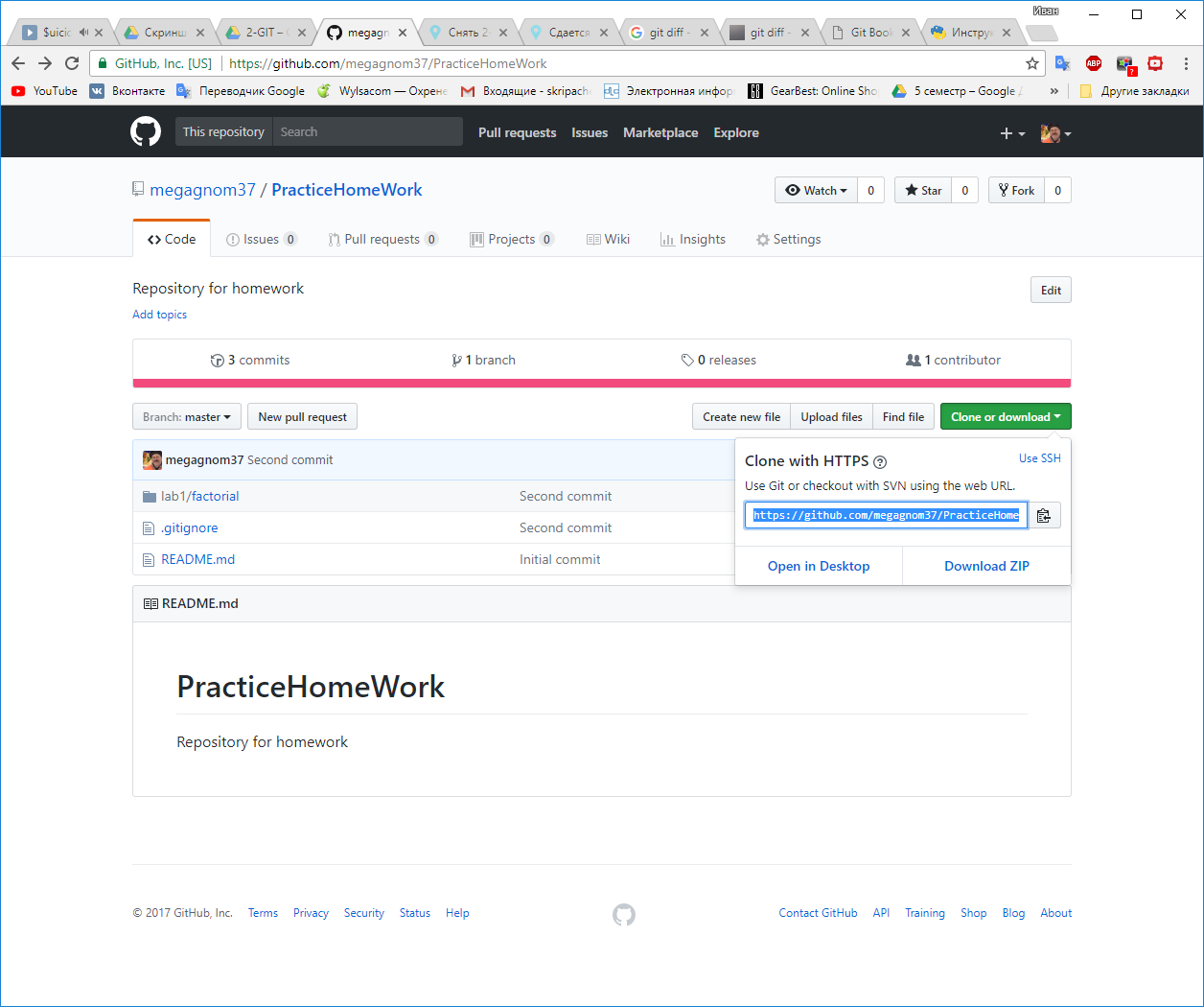


Рисунок 4-ссылка на репозиторий

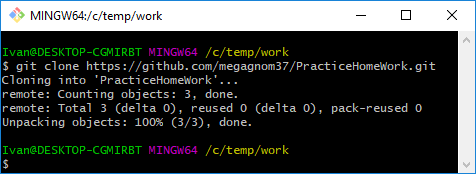


Рисунок 5-команда git clone <LINK>

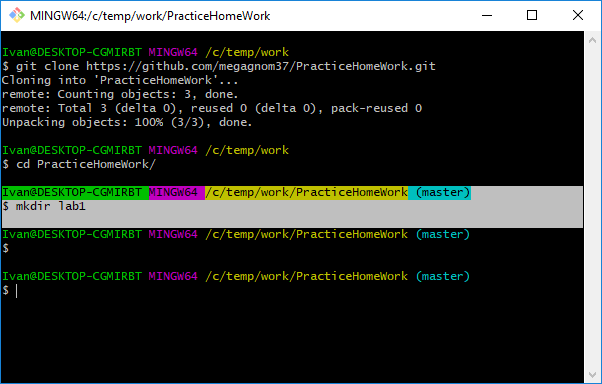
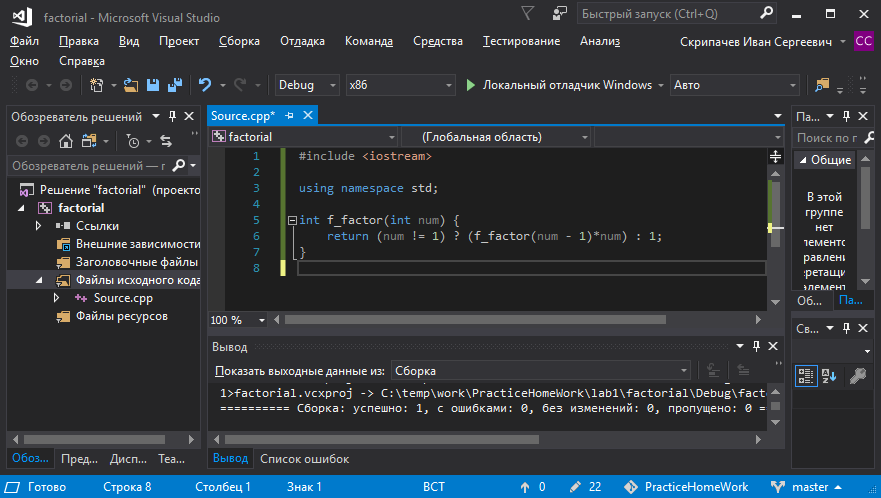
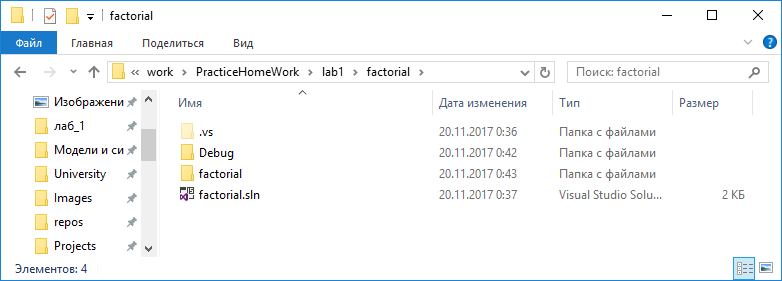
1. Создать папку lab1. Создаем папку с именем lab1  
   

Рисунок 6-создание папки

1. Написать в папке lab1 программу, вычисляющую функцию факториала.

  
Рисунок 8-функция нахождения факториала

  
Рисунок 9-директория с созданной программой

1. Создать .gitignore, добавив в игнорируемые файлы – промежуточные файлы компиляции (объектные файлы и другие временные файлы) и исполняемый файл. В результате должны остаться только файл проекта и исходные файлы.

На рис.10 показан созданный файл .gitignore. На Рис. 11 видно содержимое данного файла с игнорируемыми расширениями.

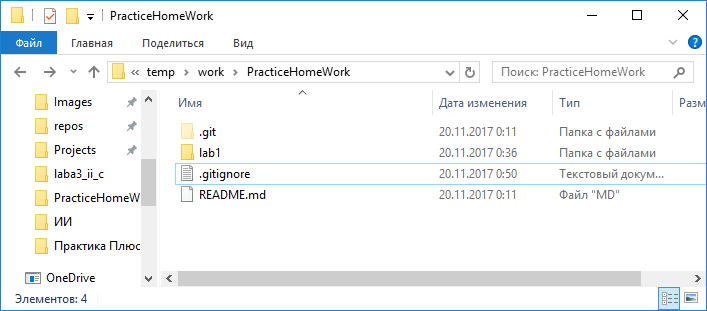
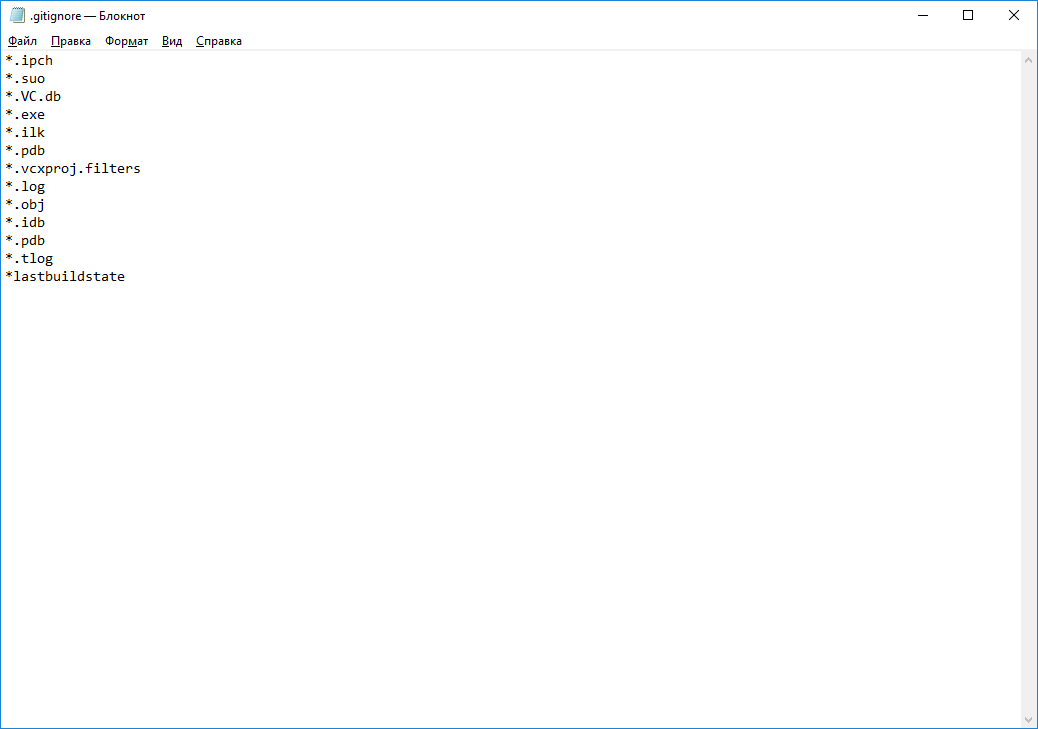


Рисунок 10-файл .gitignore

  
Рисунок 11-содержимое файла .gitignore

1. Сделать по крайней мере два коммита.

Рис.12 commit до изменения файла с программой.

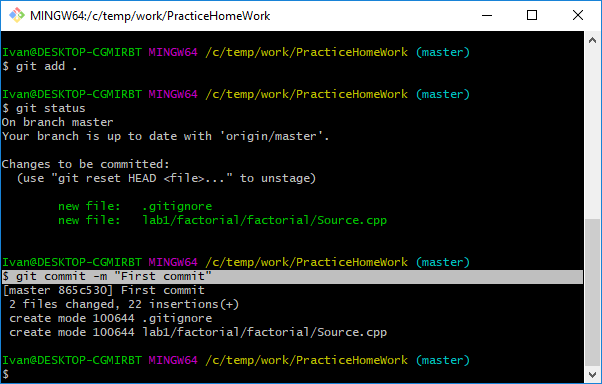
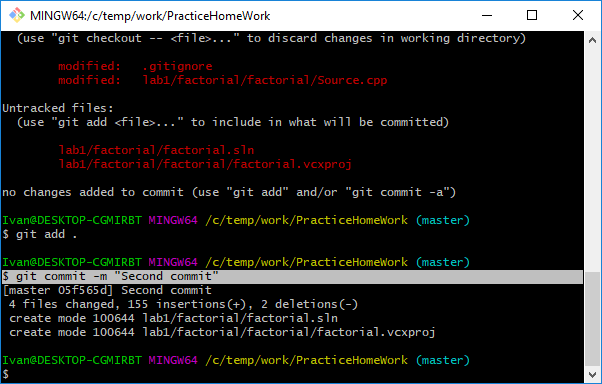
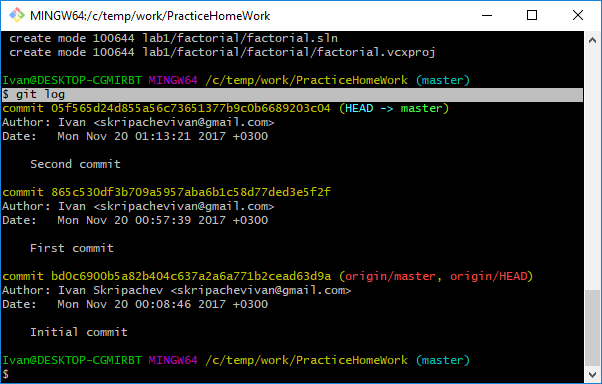


Рисунок 12- первый commit

Рис.13 commit после внесения изменений в файл с программой.

  
Рисунок 13- второй commit

1. Привести результаты работы “git log”. Рис.14 показывает нам результат работы команды git log, а именно информацию о первоначальном инициализирующем ‘commit’ и двух последующих, добавленных нами вручную.

  
Рисунок 14- команда git log

1. Показать при помощи “git diff” изменение файлов между двумя коммитами. Результат выполнения команды показан на рис.15.

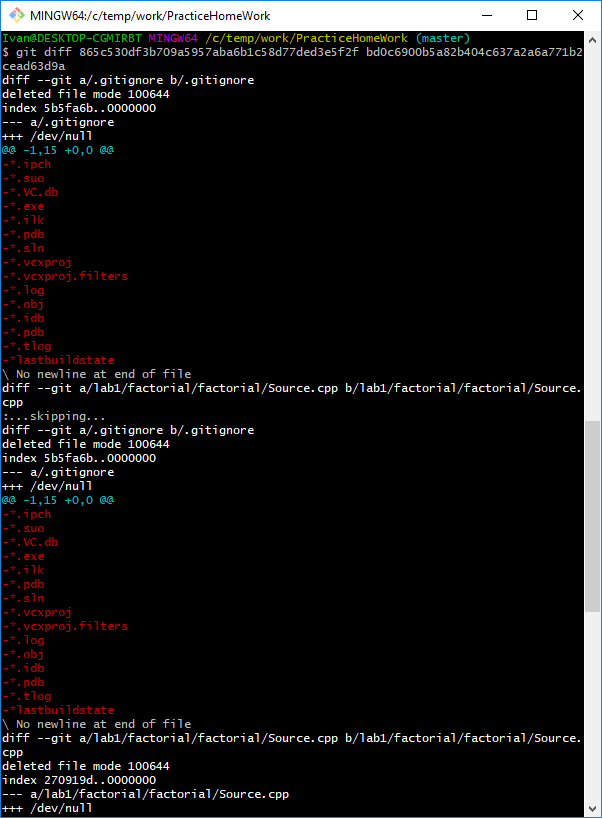
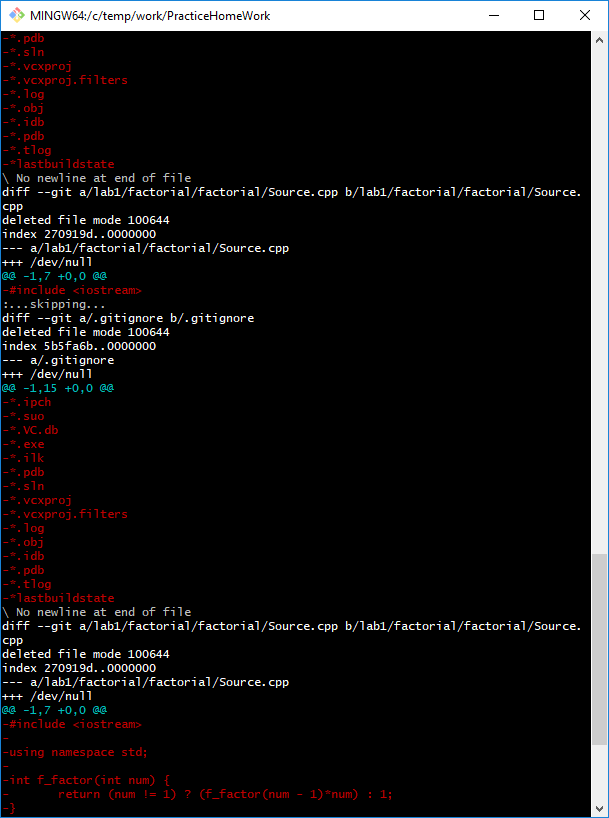
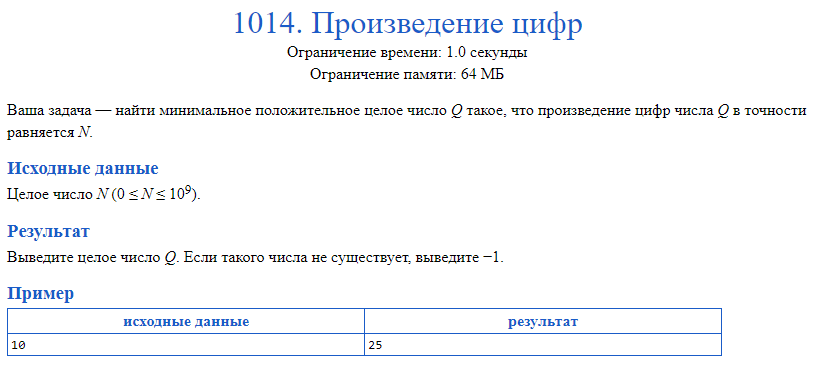
  


Рисунок 75- команда git diff

# Задание №2. «Форматирование и стиль»

1) Условия задания:



2) Описание алгоритма:

На каждом шаге цикла пытаемся сократить значение (входное) на максимальный делитель (максимальная цифра – 9). Если число делится на него без остатка, то к текущему полученному числу Q добавляем недостающее число с умножением на коэффициент разрядности. Обновляем этот коэффициент. Меняем значение текущего числа на его целую часть от деления. Если в результате переданное значение равно 1, то возвращаем найденное Q. Иначе возвращаем -1 (по условию).

В начале метода проверяем и обрабатываем исключительные ситуации:

* Переданное число равно 0 (возвращаем 10)
* Переданное число равно 1 (возвращаем 1)

3) Текст программы:

#include <iostream>

long long MulOfNumbers(long long aNumber) {

if (aNumber == 0) {

    return 10;

  } else if (aNumber == 1) {

    return 1;

  }

  //

  long long q(0);

  long long multiplier(1);

  //

  for (long long i(9); i > 1; --i) {

    while ((aNumber % i) == 0) {

      q = q + (multiplier \* i);

      multiplier = multiplier \* 10;

      aNumber = aNumber / i;

    }

  }

  //

  return (aNumber == 1) ? q : -1;

}

int main() {

  long long n;

  std::cin >> n;

  std::cout << MulOfNumbers(n) << std::endl;

  return 0;

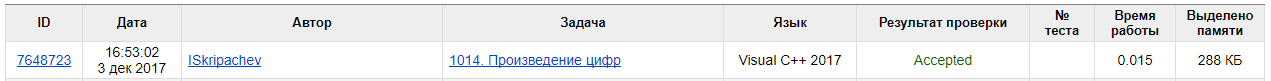
}

4) Исходные данные, используемые при вводе в программу и результат выполнения программы:

*Таблица 1. Значения входных и выходных данных.*

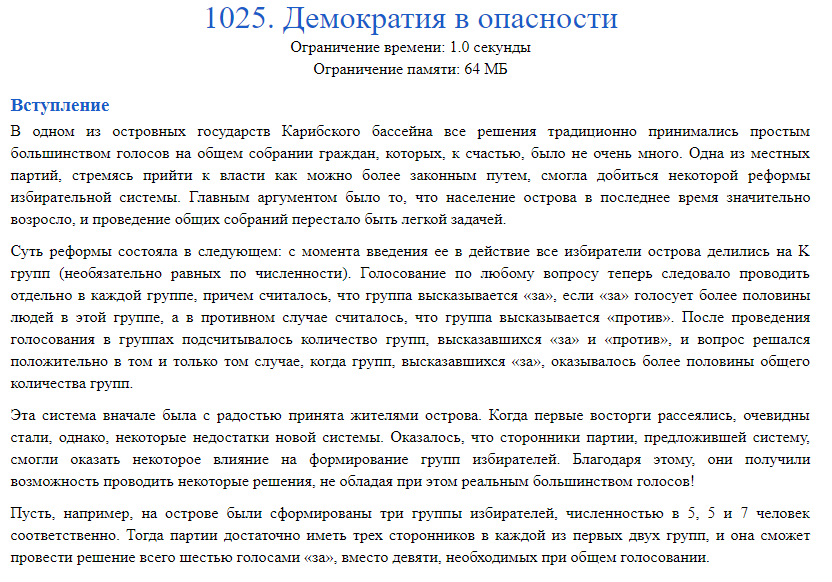
|  |  |
| --- | --- |
| Исходные данные | Результат |
| 0 | 10 |
| 1 | 1 |
| 6 | 6 |
| 10 | 25 |
| 120 | 358 |
| 333 | -1 |
| 557 | -1 |
| 1678 | -1 |
| 150000 | 5555568 |
| 1000000000 | 555555555888 |

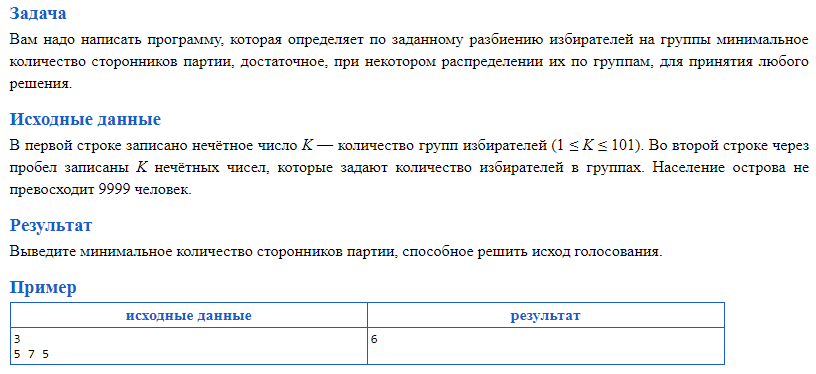
5) Отчет о результатах работы:

  
*Рисунок 16. Результат проверки задания на timus.*

# Задание №3. «Тестирование

1) Условия задания:

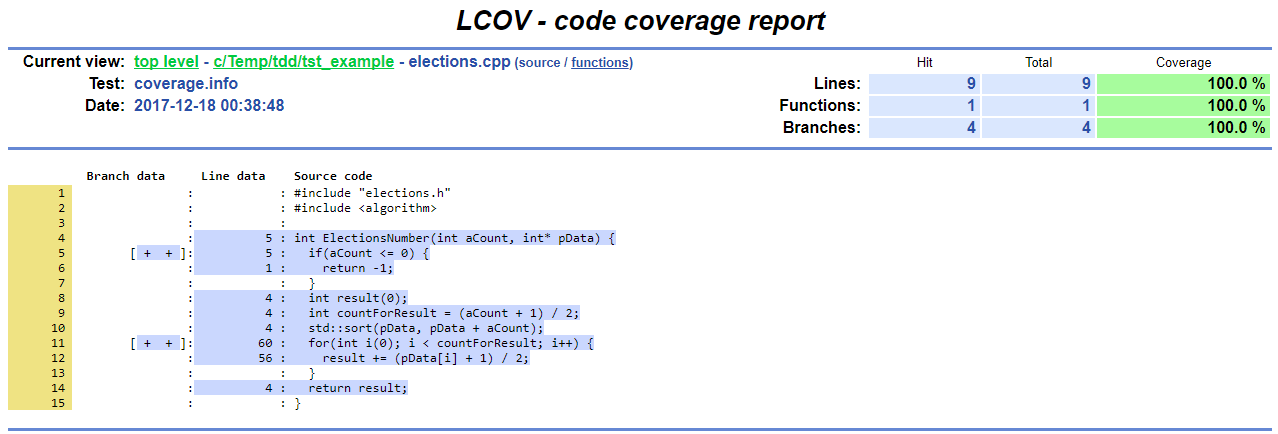
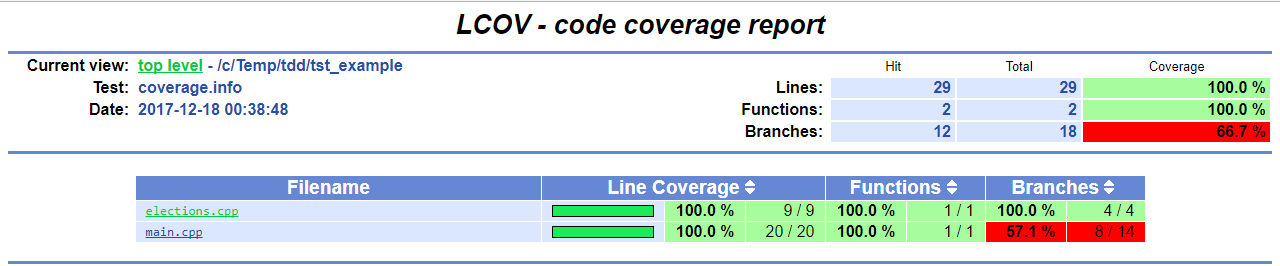




2) Описание алгоритма:

Сортируем значения количества людей по возрастанию. Берем количество групп на 1 больше половины. Берем количество людей на 1 больше половины. Добавляем к результату полученное значение.

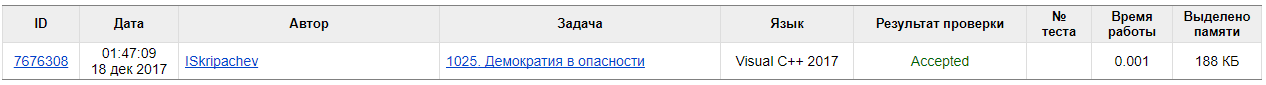
3) Результаты тестирования



4) Ссылка на репозиторий с заданием

<https://github.com/megagnom37/PracticeHomeWork/tree/master/lab_3_tdd/test>

5) Результат прохождения задания на timus



6) Текст программы

main.cpp  
#include "elections.h"

#define MY\_DEF\_USE\_LIBTAP

#ifdef MY\_DEF\_USE\_LIBTAP

#define TAP\_COMPILE

#include "libtap\cpp\_tap.h"

*using* *namespace* std;

int main(int, char \*[]) {

*//*

plan\_tests(5);

int\* arr = *new* int[1000];

*//*

arr[0] = 5;

arr[1] = 7;

arr[2] = 5;

ok(ElectionsNumber(3, arr) == 6, "count of 3 groups with data: [5,7,5] is equal 6");

*//*

arr[0] = 3;

arr[1] = 5;

arr[2] = 8;

arr[3] = 9;

ok(ElectionsNumber(4, arr) == 5, "count of 4 groups with data: [3,5,8,9] is equal 5");

*//*

ok(ElectionsNumber(0, arr) == -1, "invalid count of groups");

*//*

arr[0] = 3;

ok(ElectionsNumber(1, arr) == 2, "count of 1 groups with data: [3] is equal 2");

*//*

*for*(int i(0); i < 101; i++) {

arr[i] = 99;

}

ok(ElectionsNumber(101, arr) == 2550, "count of 101 groups with data: [99, ..., 99] is equal 2550");

*delete*[] arr;

*return* exit\_status(); *//* *вывод* *отчета* *по* *тестам*

*return* 0;

}

#endif

elections.cpp  
#include "elections.h"

#include <algorithm>

int ElectionsNumber(int aCount, int\* pData) {

*if*(aCount <= 0) {

*return* -1;

}

int result(0);

int countForResult = (aCount + 1) / 2;

std::sort(pData, pData + aCount);

*for*(int i(0); i < countForResult; i++) {

result += (pData[i] + 1) / 2;

}

*return* result;

}

# Проектная работа. Приложение-виджет для андроид

1) Условия задания:

Создать приложение-виджет для устройств на платформе Android.

Виджет должен отображать время прибытия автобуса к остановке и оставшееся время до его ухода от остановки. Данная информация должна отображаться для последнего ушедшего, следующего и после следующего автобусов.

Информация о расписании автобуса задается через Google-таблицы.

2) Описание приложения:

Внешний вид виджета можно увидеть на рис. 17.

Виджет визуально делится на несколько составляющих:

* Имя для расписания
* Визуализированная кнопка обновления информации о ближайших автобусах
* Блок из трех цветных строк и двух столбцов.
  + Левый столбец – время прибытия автобуса на остановку
  + Правый столбец – время до/после прибытия/отправления автобуса к/от остановки
  + Строки:
    - Красная – последний ушедший автобус
    - Зеленая – ближайший прибывающий
    - Желтая – следующий после ближайшего прибывающего

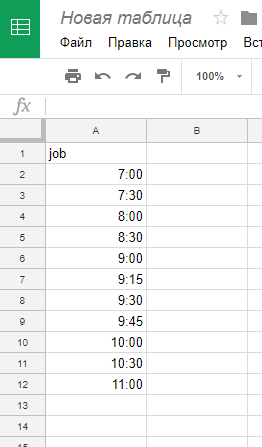
Виджет занимает пространство размером 2х2 на рабочем столе устройства.

Настройка виджета происходит при его создании. При создании виджета приложение предложит пользователю выбрать файл в формате CSV в одной из локальных директорий. После выбора пользователем нужного файла, программа автоматически проанализирует файл и сгенерирует виджет относительно введенных данных. Образец корректно заполненных CSV таблиц представлены на рис. 18.

Создание и экспорт таблиц в формате CSV можно делать даже онлайн (например в Google таблицах), что позволяет пользоваться виджетом без установки дополнительного софта.

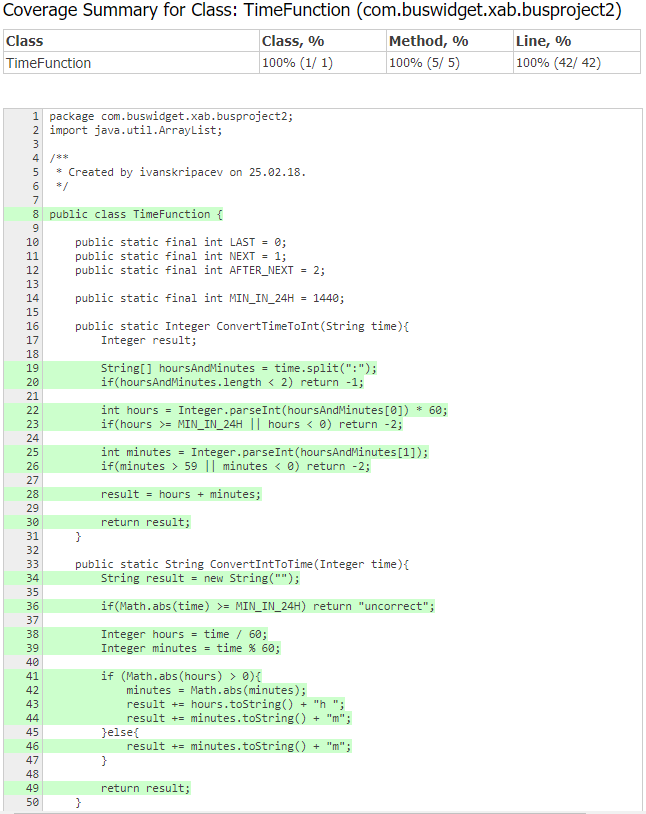


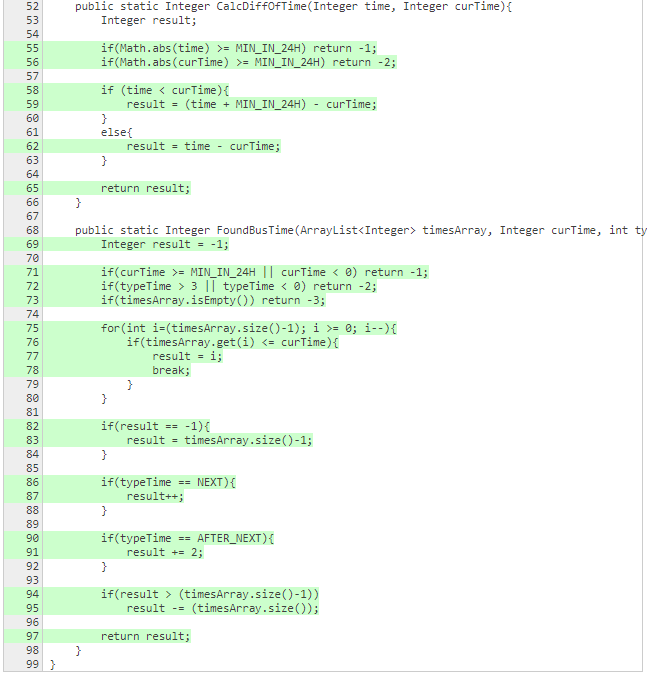
*Рисунок 17. Запущенное приложение-виджет на Android-устройстве.*



*Рисунок 18. Пример корректно заполненной CSV таблицы в сервисе Google таблицы.*

3) Результаты тестирования





4) Ссылка на репозиторий проекта

<https://github.com/megagnom37/PracticeHomeWork/tree/master/Project>

Ссылка на тестовый репозиторий

<https://github.com/megagnom37/PracticeHomeWork/tree/master/Project/BusProject2/test_report>

# Приложение А. Презентация с семинара по С++11 на тему: Tuple and tie.

Ссылка на презентацию:

<https://drive.google.com/drive/folders/0B-W-Xq4KaDMnLWpYVXlpUXB4VGM>