ФГБОУ ВО

Национальный исследовательский университет «МЭИ»

Институт радиотехники и электроники им. В.А. Котельникова

Кафедра радиотехнических систем

Курсовой проект

по дисциплине «Аппаратура потребителей спутниковых радионавигационных систем»

«Расчет траектории движения спутника GPS по данным с демодулятора его сигнала»

|  |  |
| --- | --- |
| Группа: | ЭР-15-17 |
| ФИО студента: | Танкина А.М. |
| ФИО преподавателя: | Корогодин И.В. |

Москва 2022

# РЕФЕРАТ

Курсовой проект по теме «Расчет траектории движения спутника GPS по данным с демодулятора его сигнала» содержит 30 страниц текстового документа, 13 рисунков, 3 приложения, 4 использованных источника.

Цель проекта – разработка модулей разбора навигационного сообщения GPS и расчета положения спутника, предназначенных для использования в составе навигационного приемника.

В рамках курсового проекта были поставлены задачи:

* разработка модуля разбора символов навигационного сообщения;
* расчет положения КА в Matlab/Python и его проверка сторонними сервисами;
* реализация модуля расчета положения КА на С/С++ и его тестирование.

Конечная цель всего курсового проекта – получить библиотеку функций на Си++, позволяющую рассчитывать положение спутника GPS по данным с демодулятора его сигнала L1 C/A. На первом этапе был реализован модуль разбора навигационного сообщения до структуры эфемерид и проведено сравнение результатов со сторонней программой.

СОДЕРЖАНИЕ

[РЕФЕРАТ 2](#_Toc105002069)

[ВВЕДЕНИЕ 4](#_Toc105002070)

[1 Обработка логов навигационного приемника 6](#_Toc105002071)

[1.1 Задание 6](#_Toc105002072)

[1.2 Разработка программы обработки исходного файла и вывода таблицы эфемерид 7](#_Toc105002073)

[1.3 Сравнение результатов с таблицей из программы RTKNAVI 8](#_Toc105002074)

[ВЫВОД 10](#_Toc105002075)

[2 Моделирование траектории движения 10](#_Toc105002076)

[2.1 Задание 10](#_Toc105002077)

[2.2 Разработка программы расчета положения спутника 10](#_Toc105002078)

[2.3 Результаты работы программы 12](#_Toc105002079)

[2.4 Сравнение результатов с Trimble GNSS Planning Online 14](#_Toc105002080)

[ВЫВОД 15](#_Toc105002081)

[3 Реализация модуля расчета координат 16](#_Toc105002082)

[3.1 Задание 16](#_Toc105002083)

[3.2 Разработка программы расчета положения спутника на языке С++ 17](#_Toc105002084)

[3.3 Потребляемая память и утечки 18](#_Toc105002085)

[ВЫВОД 19](#_Toc105002086)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 20](#_Toc105002087)

[ПРИЛОЖЕНИЕ А 21](#_Toc105002088)

[ПРИЛОЖЕНИЕ Б 24](#_Toc105002089)

[ПРИЛОЖЕНИЕ В 27](#_Toc105002090)

# ВВЕДЕНИЕ

Спутниковая система навигации – комплексная электронно-техническая система, состоящая из совокупности наземного и космического оборудования, предназначенная для определения местоположения (географических координат и высоты), а также параметров движения (скорости и направления движения) для наземных, водных и воздушных объектов. Основными элементами спутниковой системы являются:

* Орбитальная группировка, состоящая из нескольких (от 2 до 30) спутников, излучающих специальные радиосигналы; наземная система управления и контроля, включающая блоки измерения текущего положения спутников и передачи на них полученной информации для корректировки информации об орбитах;
* Приемное клиентское оборудование, используемое для определения координат;
* Информационная радиосистема для передачи пользователям поправок, позволяющих значительно повысить точность определения координат.

Принцип работы спутниковых систем навигации основан на измерении расстояния от антенны на объекте (координаты которого необходимо получить) до спутников, положение которых известно с большой точностью. Таблица положений всех спутников называется альманахом, которым должен располагать любой спутниковый приемник до начала измерений. Обычно приемник сохраняет альманах в памяти со времени последнего выключения и если он не устарел – мгновенно использует его. Каждый спутник передает в своём сигнале весь альманах. Таким образом, зная расстояния до нескольких спутников системы, с помощью обычных геометрических построений, на основе альманаха, можно вычислить положение объекта в пространстве.

Навигационные спутники передают два вида данных — альманах и эфимерис. Данные эфимериса содержат очень точные корректировки параметров орбит и часов для каждого спутника, что требуется для точного определения координат. Каждый навигационный спутник передает данные только своего собственного эфимериса. Первый этап курсового проекта нацелен на разработку модуля разбора навигационного сообщения до структуры эфемерид.

# Обработка логов навигационного приемника

## 1.1 Задание

В неизвестной локации установлен навигационный приемник, принимающий сигналы GPS L1C/A и логирующий результаты этого приема в формате NVS BINR. Собранный на пятиминутном интервале файл приложен в архиве под именем BINR.bin. Файл содержит наблюдения псевдодальностей и прочих радионавигационных параметров, демодулированные и разобранные данные навигационного сообщения.

Для удобства студентов данные демодулятора продублированы в текстовый файл in.txt. Каждая строка файла содержит данные одного сабфрейма одного навигационного сигнала в формате:

1 0 013 0R GpsL1CA 13 212130404 29 125 53 1000101110...

где 13 - номер спутника, 212130404 - счетчик сабфреймов в сигнале, 53 - ID сабфрейма в навигационном сообщении, где в первых трех битах содержится номер сабфрейма в фрейме (5 в данном примере), а далее - номер фрейма в сообщении (6 в данном примере), 1000101110... символы с демодулятора в порядке возрастания времени слева направо.

Требуется:

1. Разработать программу, обрабатывающую файл in.txt и выводящую в файл out.txt таблицу эфемерид для спутника согласно варианту в заданном формате.
2. Обработать файл BINR.bin с помощью программы RTKNAVI из состава RTKLIB. Определить день и место проведения наблюдений, значения эфемерид для спутника согласно номеру варианта (меню открывается в левом нижнем углу экрана по нажатию на квадрат)
3. Сравнить полученные таблицы
4. Оформить код программы и разместить на Github
5. Оформить отчет по этапу и разместить на Github
6. Завести Pull Request

Программа должна компилироваться gcc, все входные данные брать из in.txt, весь вывод осуществлять в out.txt.

## 1.2 Разработка программы обработки исходного файла и вывода таблицы эфемерид

Для решения поставленной задачи необходимо подгрузить в разрабатываемую программу файл исходных данных, содержимое которого представлено на рисунке 1.

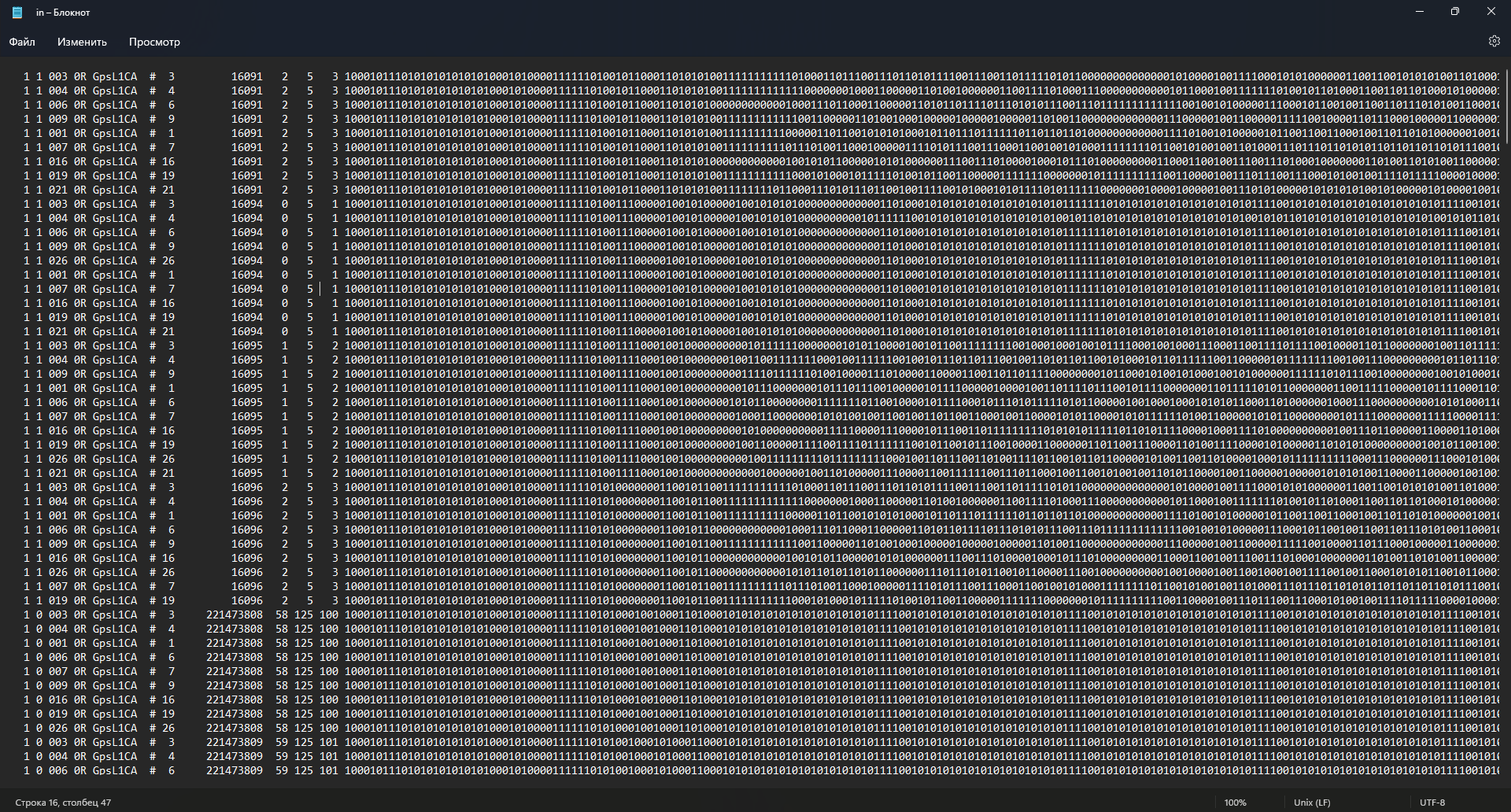


Рисунок 1 – Содержимое файла in.txt

Из этого файла нас интересуют данные, исходящие от девятого спутника. Информация об эфемеридах содержится в первых трех сабфреймах передаваемого сообщения, поэтому дальнейшая работа будет осуществляться со структурой WorkingSub, в которую производится извлечение этих данных.

Из полученной структуры декодируются параметры орбиты и ухода часов с учетом знаковости величины, ее формы представления, а также местоположения описывающих ее бит.

Также производится вывод полученных значений и сохранение их в требуемом формате в файл out.txt. На рисунке 2 представлены результаты работы программы, скомпилированной с помощью командной строки.

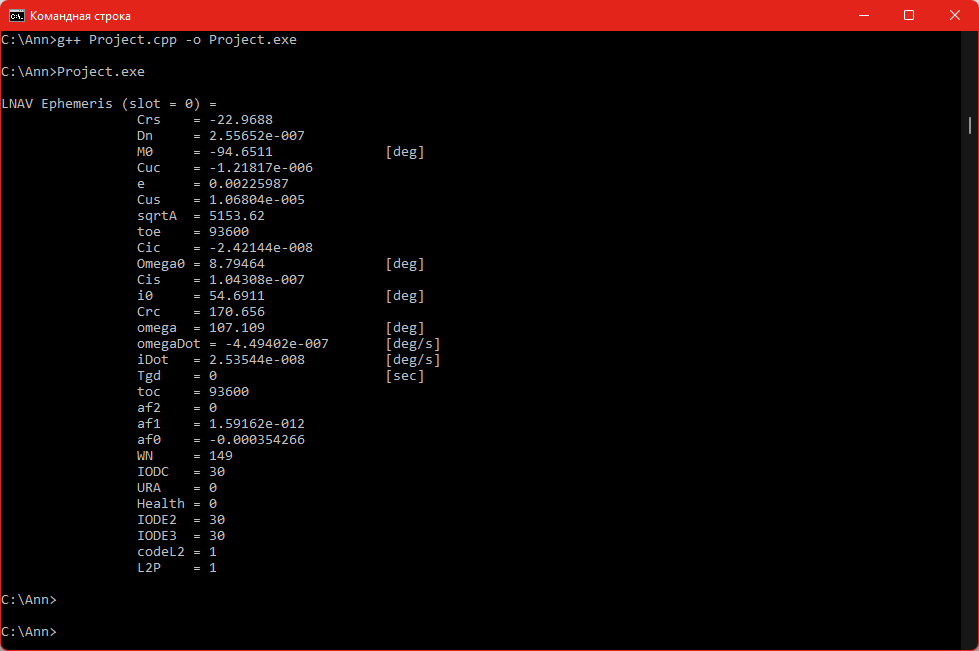


Рисунок 2 – Результаты работы программы

Листинг разработанной программы приведен в приложении.

## 1.3 Сравнение результатов с таблицей из программы RTKNAVI

Обработаем файл BINR.bin с помощью программы RTKNAVI из состава RTKLIB. Определим значения эфемерид для спутника №9 (рисунок 3) и сравним их с рассчитанными значениями (рисунок 4).

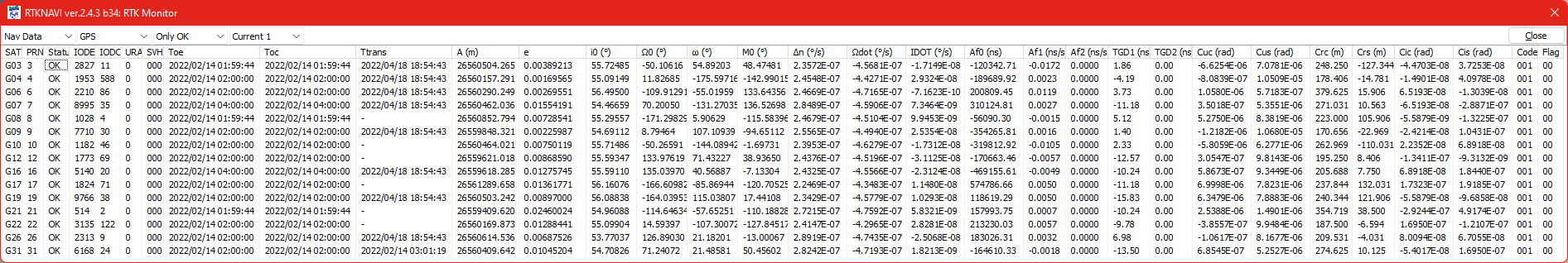


Рисунок 3 – Эфемериды из программы RTKNAVI

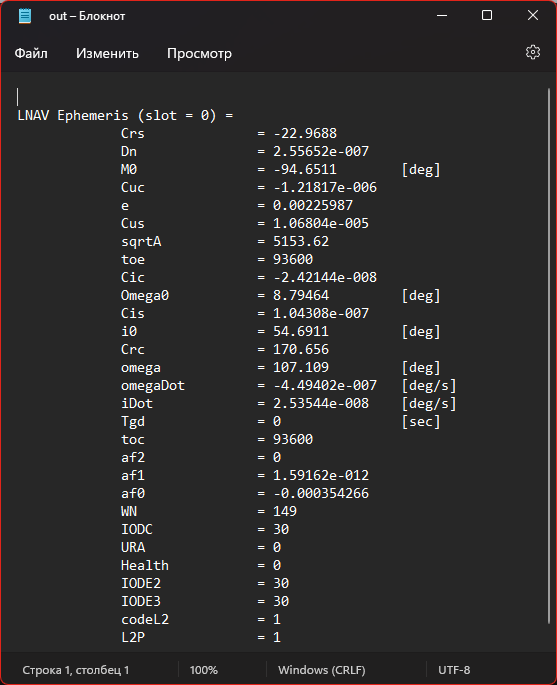


Рисунок 4 – Эфемериды разработанной программы

Из приведенных выше рисунков видно, что значения эфемерид, полученные сторонней программой, сходятся с результатами, полученными с помощью разработанной программы, что свидетельствует о ее корректной работе.

## ВЫВОД

На первом этапе курсового проекта была разработана программа, обрабатывающая данные демодулятора из файла in.txt и выводящая в файл out.txt таблицу эфемерид требуемого спутника.

# 2 Моделирование траектории движения

## 2.1 Задание

Требуется реализовать на языке Matlab или Python функцию расчета положения спутника GPS на заданный момент по шкале времени UTC. В качестве эфемерид использовать данные, полученные на предыдущем этапе.

Построить трехмерные графики множества положений спутника GPS с системным номером, соответствующим номеру студента по списку. Графики в двух вариантах: в СК ECEF WGS84 и соответствующей ей инерциальной СК. Положения должны соответствовать суточному интервалу на дне формирования наблюдений, определенном на предыдущем этапе. Допускается использовать одни и те же эфемериды на весь рассматриваемый интервал.

Вывести значения координат спутника в файл out.txt в системе ECEF WGS 84 в виде строк: Секунда\_от\_начала\_дня X Y Z.

Используя оценку местоположения с предыдущего этапа, построить Sky Plot за указанный временной интервал и сравнить результат с Trimble GNSS Planning Online.

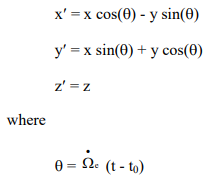
## 2.2 Разработка программы расчета положения спутника

Для решения поставленной задачи необходимо воспользоваться константами из ИКД и некоторыми значениями эфемерид, полученными на предыдущем этапе. Таблица использованных эфемерид приведена ниже.

Таблица 1 – Значения использованных эфемерид

|  |  |
| --- | --- |
| C\_rs | -22.9688 |
| delta\_n | 2.55652e-7 |
| M\_0 | -94.6511 |
| C\_uc | -1.21817e-6 |
| eks | 0.00225987 |
| C\_us | 1.06804e-5 |
| rootA | 5153.62 |
| t\_oe | 93600 + 18 - 7200 |
| C\_ic | -2.42144e-8 |
| Omega\_0 | 8.79464 |
| C\_is | 1.04308e-7 |
| i\_0 | 54.6911 |
| C\_rc | 170.656 |
| omega | 107.109 |
| Omega\_dot | -4.49402e-7 |
| IDOT | 2.53544e-8 |

На суточном интервале (86400 секунд) по алгоритму из ИКД рассчитываются координаты спутника в системе ECEF WGS 84. Перевод координат в инерциальную систему осуществляется по заданным формулам:



Далее с помощью встроенной функции ecef2enu центр декартовой системы координат перемещается в точку приема, местоположение которой определяется с помощью RTKNAVI (рисунок 5). Полученные координаты спутника относительно приемника c учетом их знака пересчитываются в полярную систему координат с помощью известных соотношений:



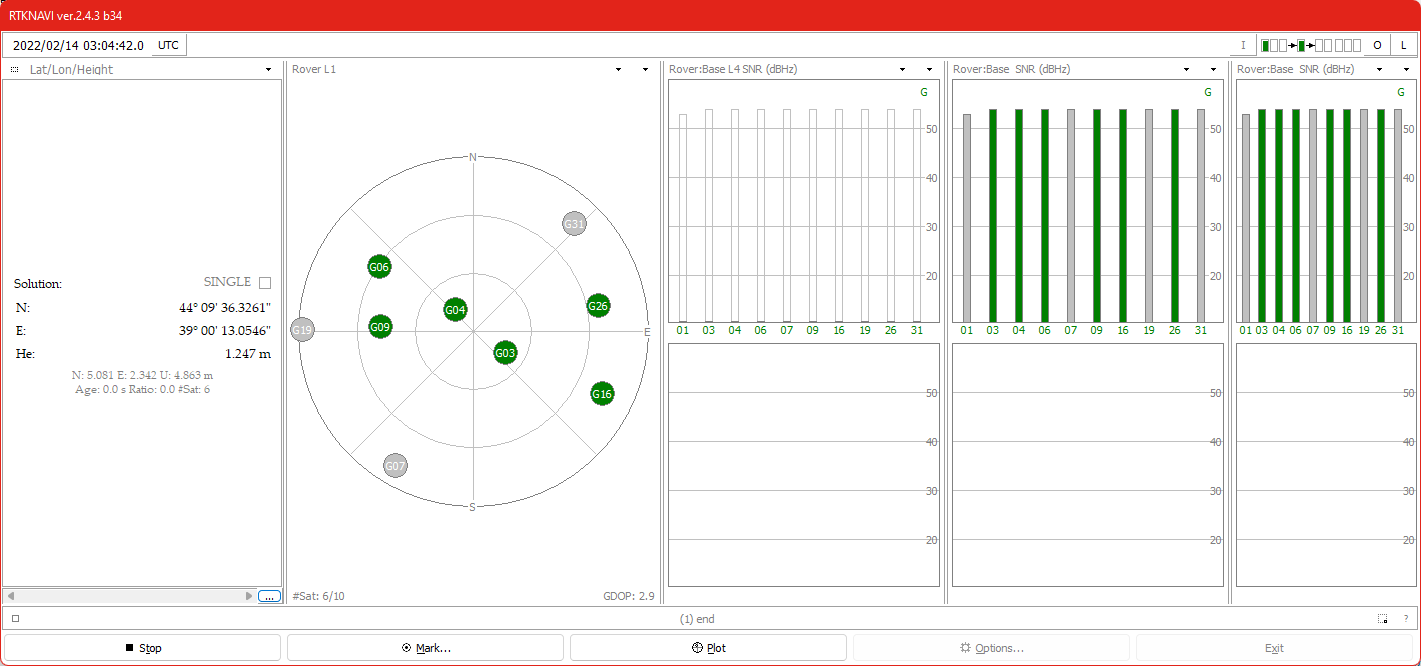


Рисунок 5 – Координаты приемника

Далее производится запись рассчитанных значений в файл out.txt и построение требуемых графиков. Листинг полученной программы представлен в приложении Б.

## 2.3 Результаты работы программы

На рисунке 6 представлен трехмерный график множества положений спутника в системе ECEF WGS84. На рисунке 7 отображены полученные положения спутника в инерциальной системе координат, а на рисунке 8 отображен соответствующий Sky Plot.

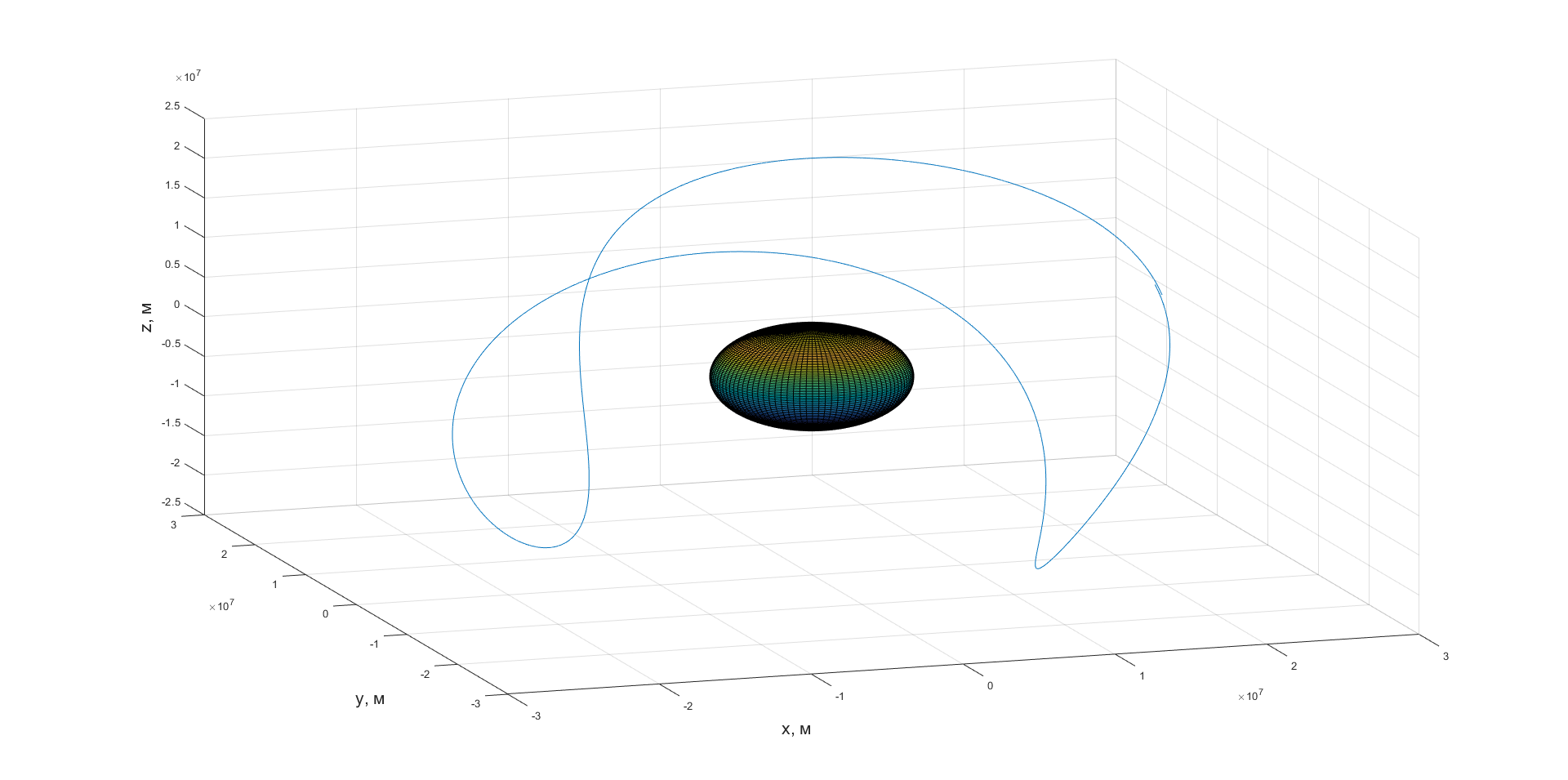


Рисунок 6 — Множество положений спутника GPS в системе ECEF WGS84

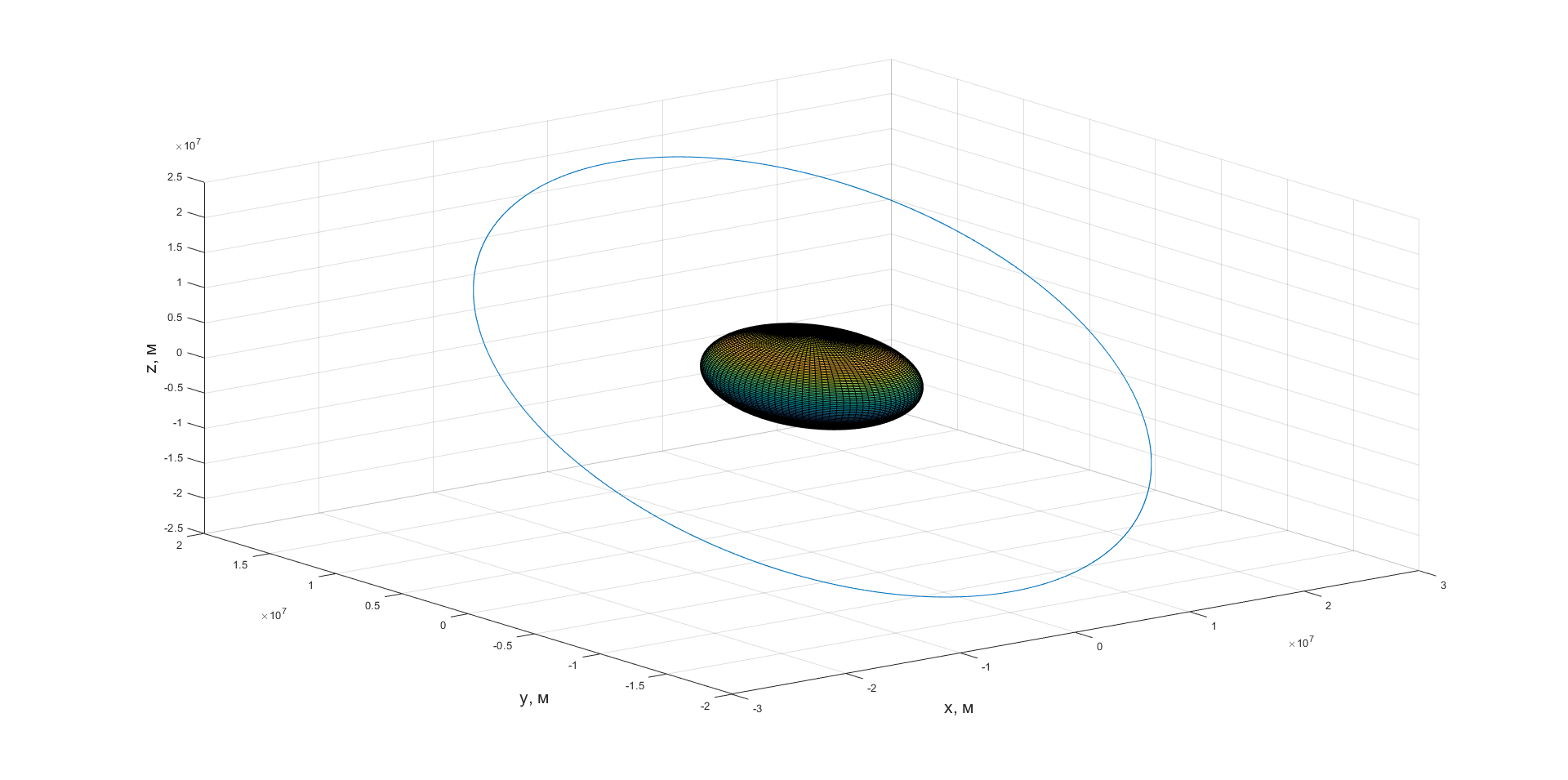


Рисунок 7 — Множество положений спутника GPS в системе ECI



Рисунок 8 — Sky Plot

## 2.4 Сравнение результатов с Trimble GNSS Planning Online

Для построения SkyView с помощью Trimble GNSS Planning Online необходимо выбрать заданный спутник, дату, рассматриваемый период, а также координаты точки приема. Полученные настройки приведены на рисунке 9.

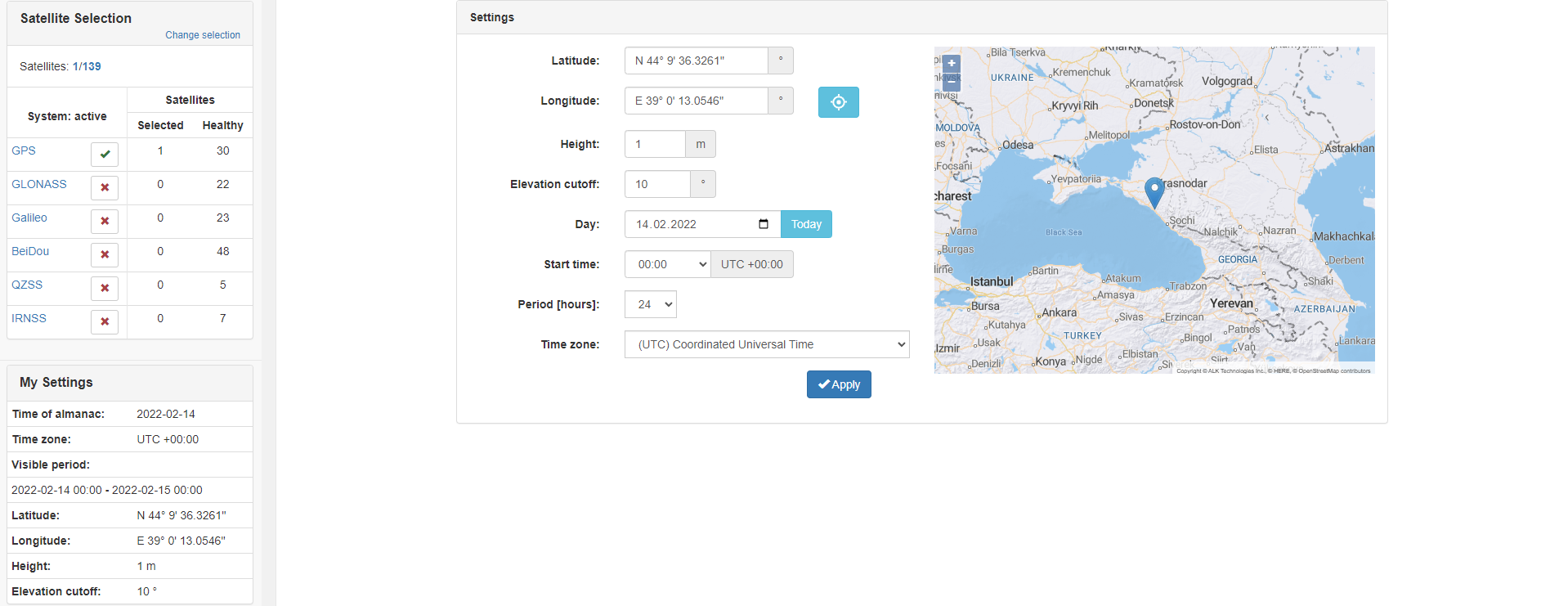


Рисунок 9 — Исходные данные для Trimble GNSS Planning Online

На рисунке 10 приведен SkyView спутника GPS №9 на интервале времени 14.02.2022 00:00 - 15.02.2022 00:00.

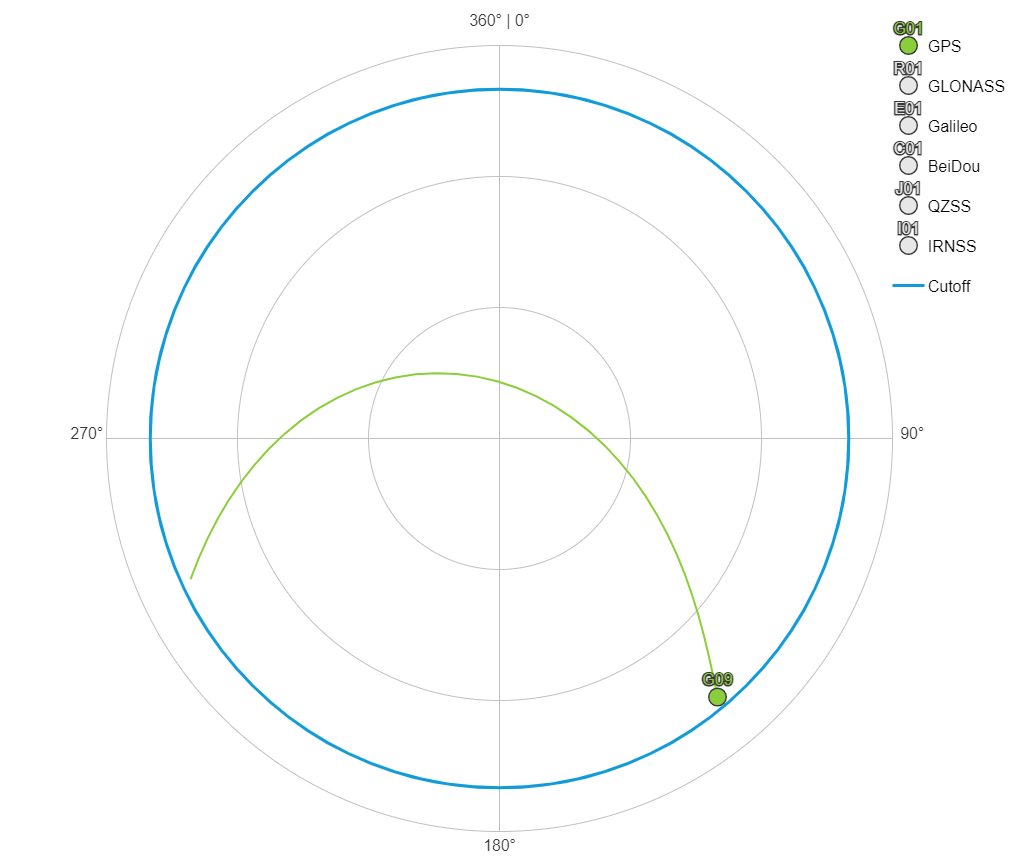


Рисунок 10 – SkyView из Trimble GNSS Planning Online

## ВЫВОД

На втором этапе курсового проекта была разработана программа, выполняющая расчет положения спутника GPS, выводящая в файл out.txt значения координат спутника в заданном формате. При сравнении графиков на рисунках 8 и 10 замечается сходство рассчитанного SkyView и полученного с помощью Trimble GNSS Planning Online.

# 3 Реализация модуля расчета координат

## 3.1 Задание

Требуется разработать на языке С/С++ функцию расчета положения спутника GPS на заданное время по шкале UTC, минимизируя время её исполнения и количество затрачиваемой оперативной памяти. Вызов функции не должен приводить к выбросу исключений или утечкам памяти при любом наборе входных данных.

Функция расчета положения спутника в Matlab/Python относительно проста, т.к. доступны библиотеки линейной алгебры и решения уравнений. Но при разработке встраиваемого ПО приходится сохранять лицензионную частоту, минимизировать вычислительную нагрузку и затраты памяти. Поэтому отобразить модель из Matlab/Python в прошивку приемника дословно, как правило, не получается. В рассматриваемом примере потребуется, как минимум, выполнить свою реализацию решения трансцендентного уравнения.

Программный модуль должен сопровождаться unit-тестами (например, используя Check):

* Тесты функции решения уравнения Кеплера
* Тест расчетного положения спутника в сравнении с Matlab/Python

Во время второго теста должно вычисляться и выводиться средняя длительность исполнения функции. Допускается использовать одни и те же эфемериды на весь рассматриваемый интервал (как на предыдущем этапе).

Требуется провести проверку на утечки памяти (например, с помощью утилиты valgrind).

Оформить отчет по результатам курсового проекта. В качестве первых двух глав использовать отчёты с предыдущих этапов, в третьей главе отразить результаты этого этапа:

1. Код реализации
2. Вывод тестов, включая анализ времени исполнения
3. Вывод проверок на утечку памяти
4. Вывод по этапу
5. Заключение по проекту

Программа должна компилироваться gcc и использовать в качестве входных данных in.txt с первого этапа. Результат должен записываться в out.txt в строки формата, определенного на втором этапе. При тестировании должны сравниваться файлы out.txt второго и третьего этапов.

## 3.2 Разработка программы расчета положения спутника на языке С++

Для реализации поставленной задачи был взят код из первого этапа курсового проекта, дополненный функцией расчета координат спутника, операцией сравнения результатов, полученных с помощью Matlab и C++, а также вычислением средней длительности исполнения функции.

Алгоритм расчета местоположения спутника включает в себя уравнение Кеплера. Реализация решения этого трансцендентного уравнения приведена на рисунке 11.

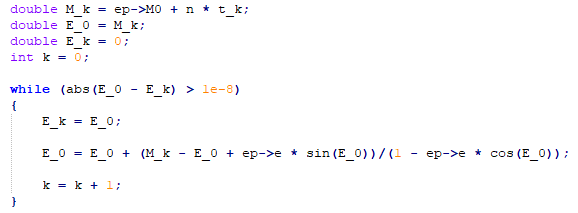


Рисунок 11 – Реализация уравнения Кеплера

Для реализации сравнения расчетного положения спутника с результатами, полученными на втором этапе курсового проекта, для повышения точности расчета в Matlab было принято решение передавать данные, полученные на первом этапе, не в форме, выводимой в out.txt, а с более высокой точностью. Результат сравнения массивов данных приведен на рисунке 12. Расхождение в значениях обусловлено округлением результатов промежуточных вычислений в Matlab. Была предпринята попытка повысить точность данных в Matlab с помощью функции vpa, однако при этом значительно падала производительность.

Также на рисунке 12 представлено время выполнения расчета.

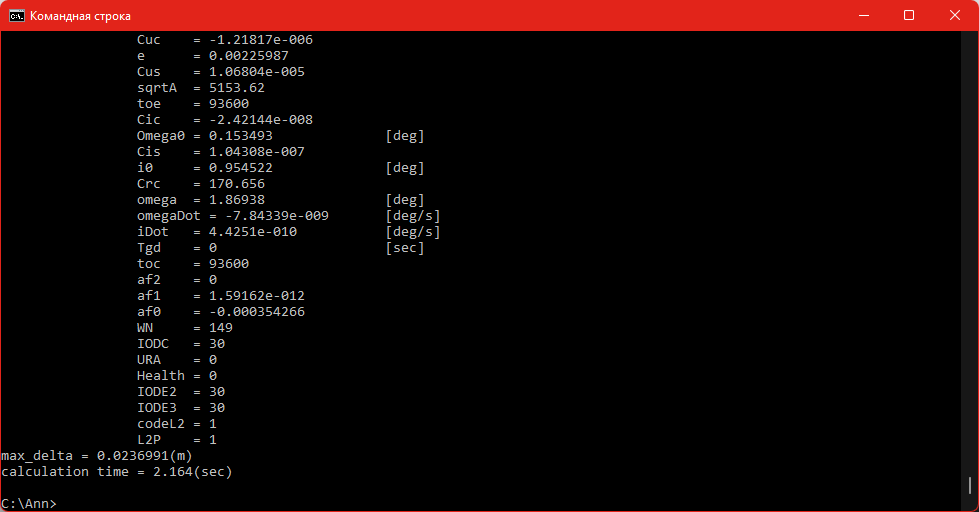


Рисунок 12 – Максимальная разница координат и длительность исполнения функции

Листинг программы, разработанной в ходе выполнения третьего этапа курсового проекта, представлен в приложении В.

## 3.3 Потребляемая память и утечки

Воспользуемся встроенными инструментами отладки Microsoft Visual Studio. Поставим точку остановки после заполнения массивов координат положения спутника и пронаблюдаем потребляемую память на рисунке 13.

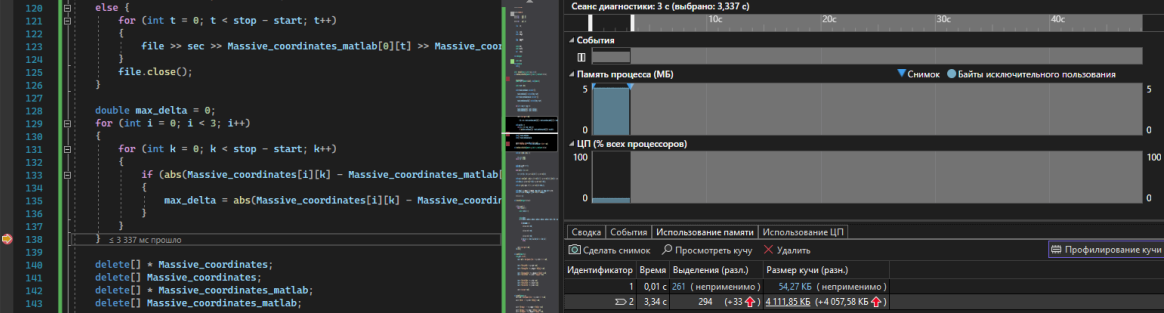


Рисунок 13 – Объем потребляемой памяти

## ВЫВОД

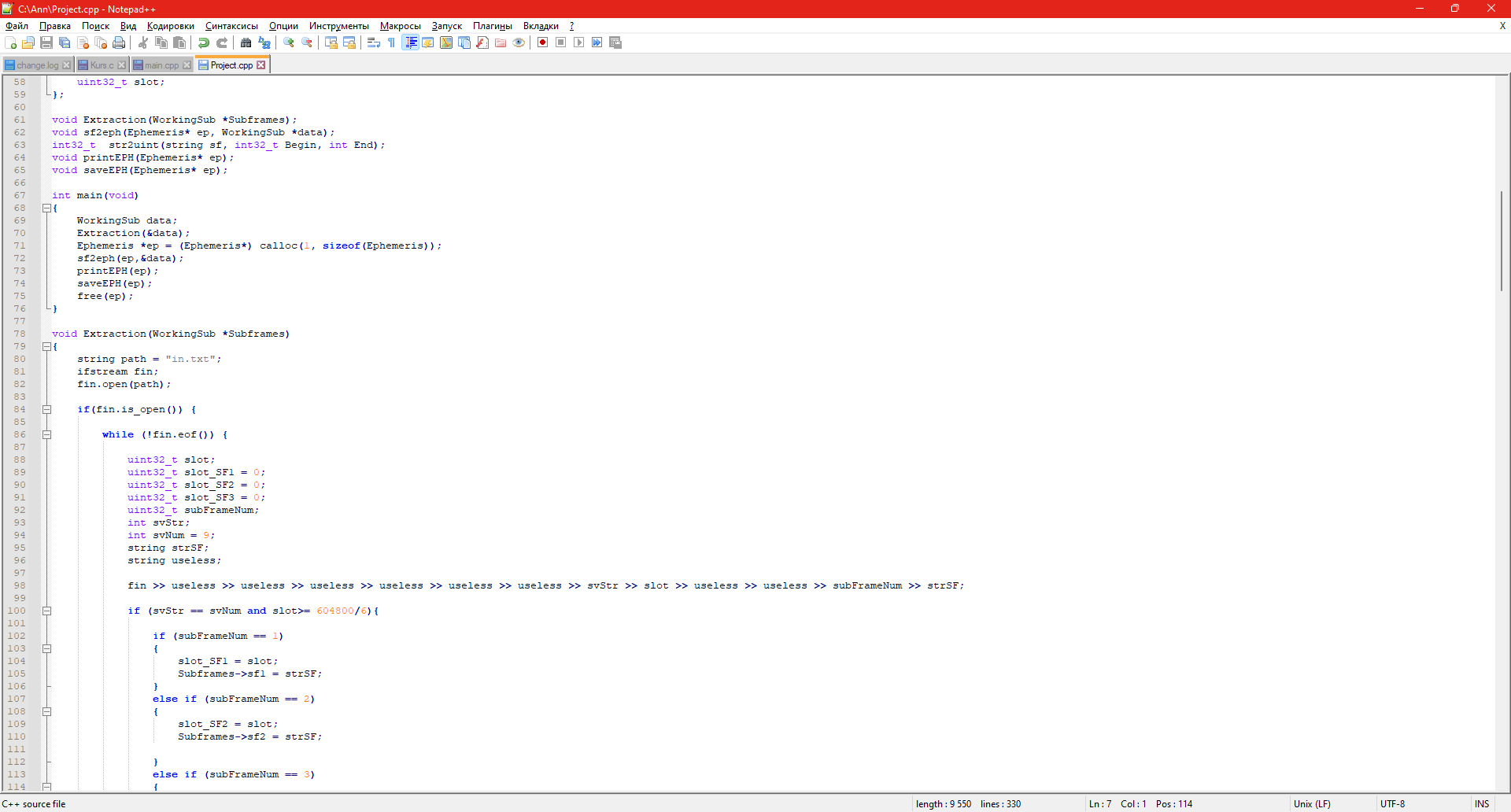
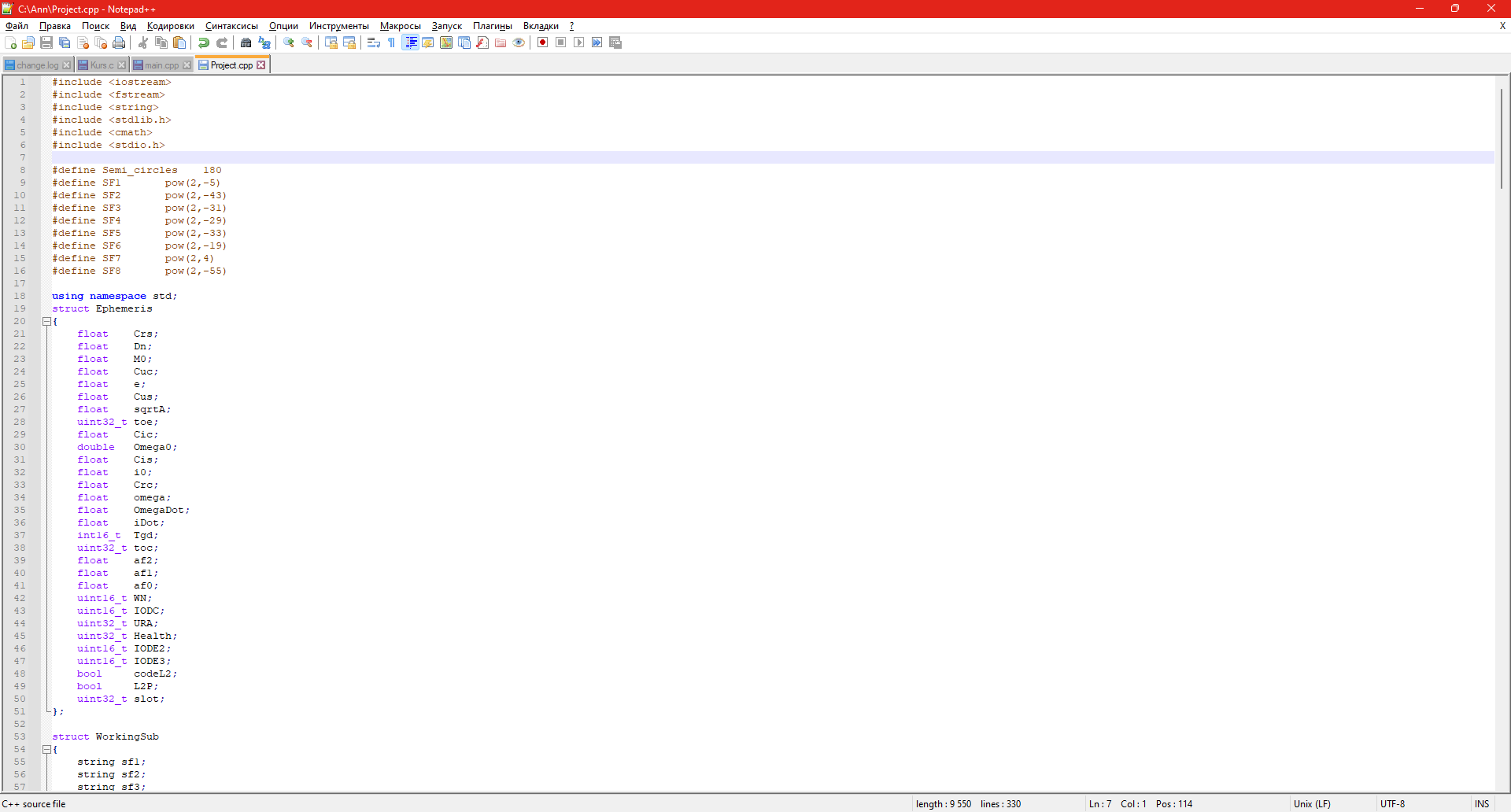
На третьем этапе курсового проекта была написана программа на языке C++, что целесообразно при разработке встраиваемого ПО. Было обнаружено расхождение в результатах, полученных на 2 и 3 этапе курсового проекта, составляющее 0,0236991 м, обусловленное различным количеством байт, выделяемом для хранения данных в Matlab и C++. Объем потребляемой памяти составил 4111,85 КБ, утечка памяти не была обнаружена.

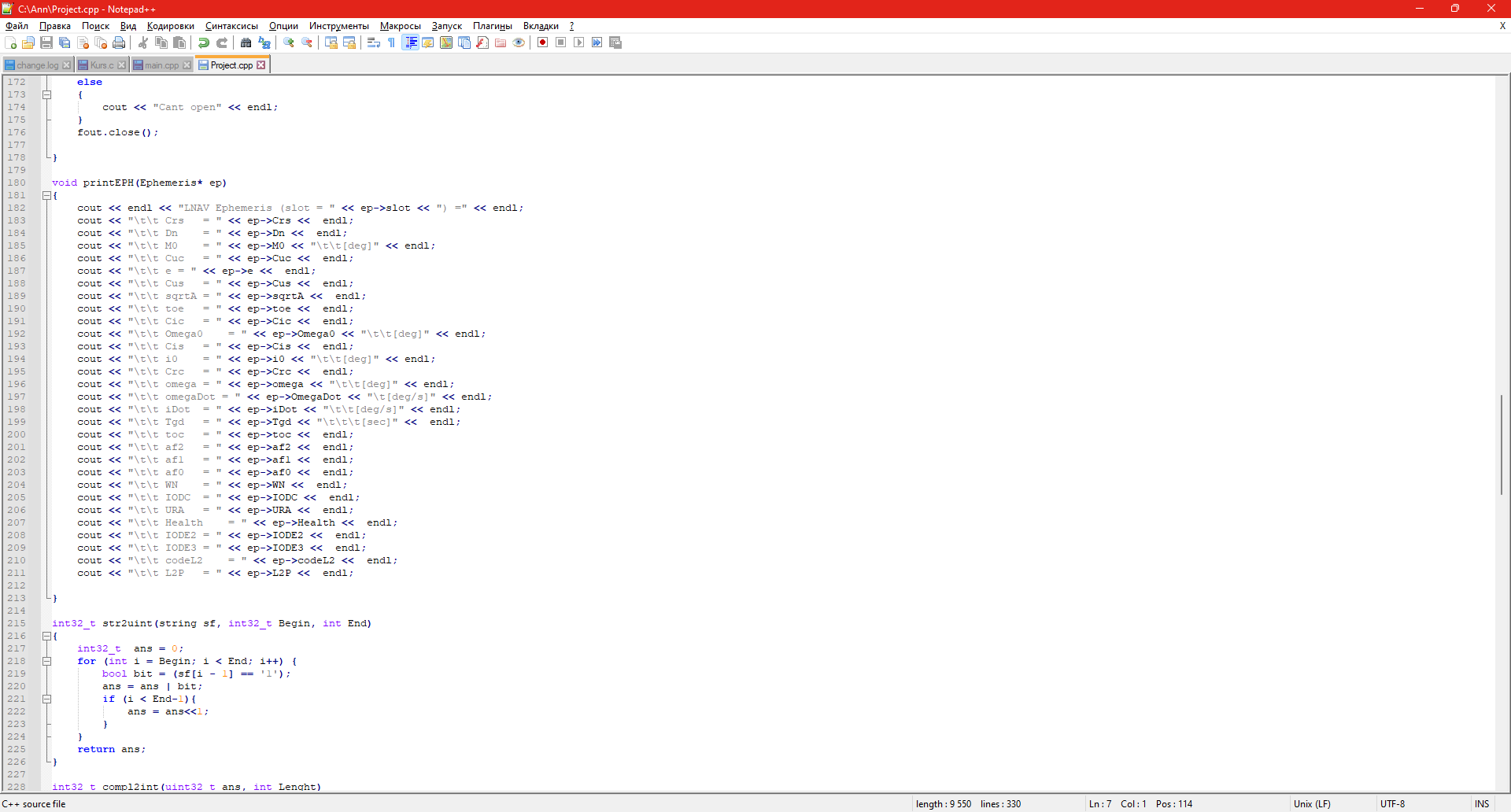
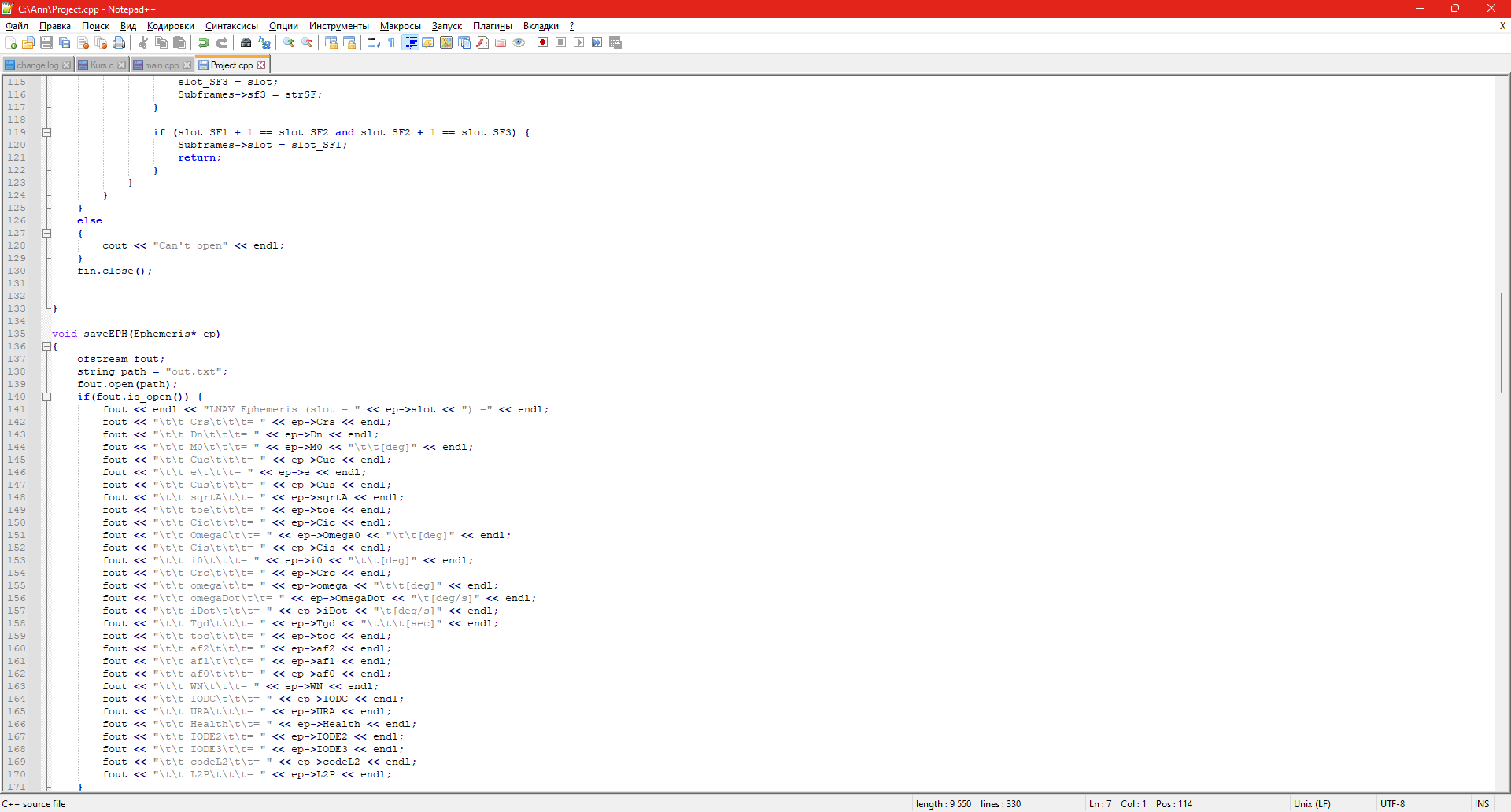
# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

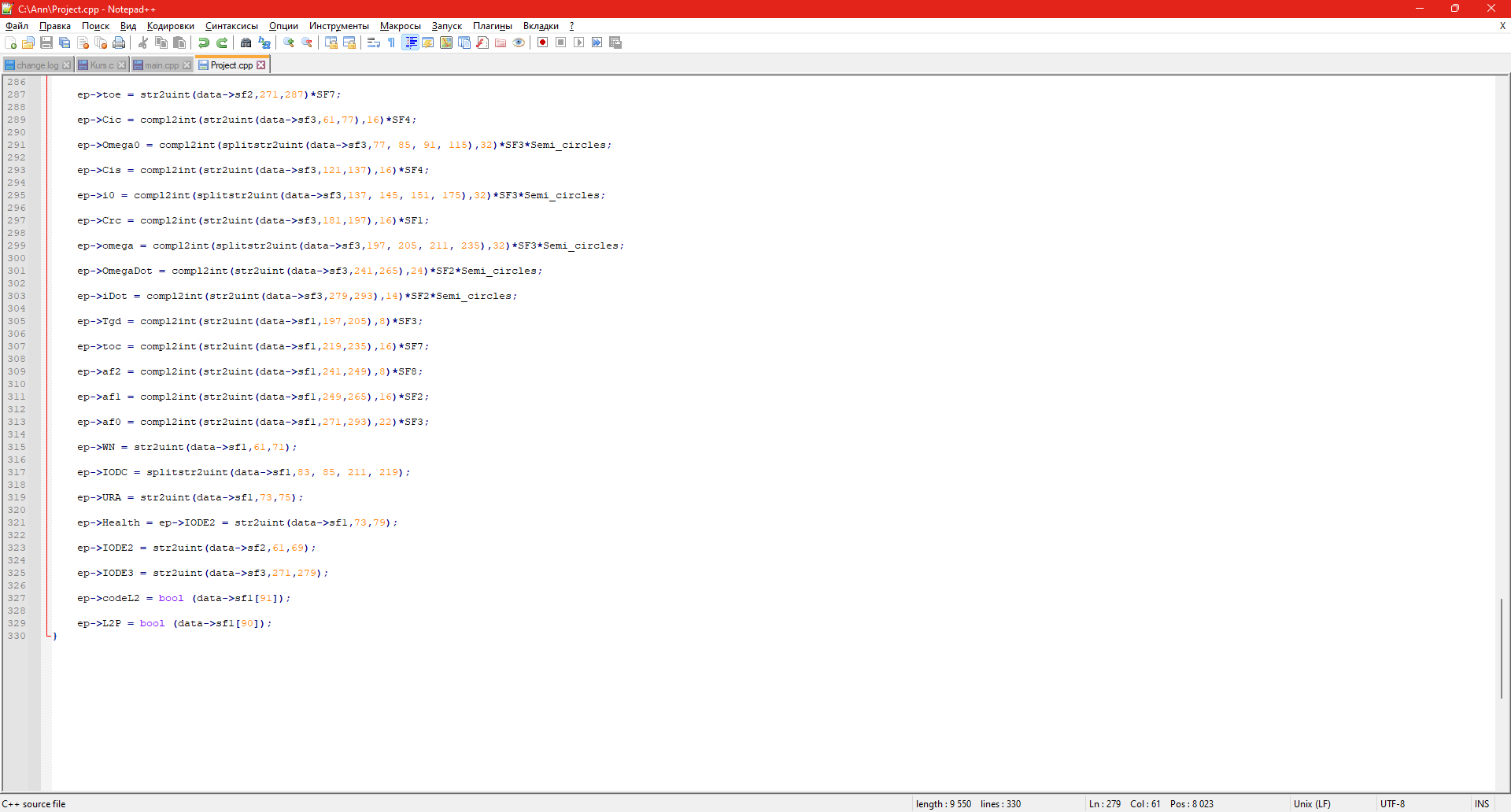
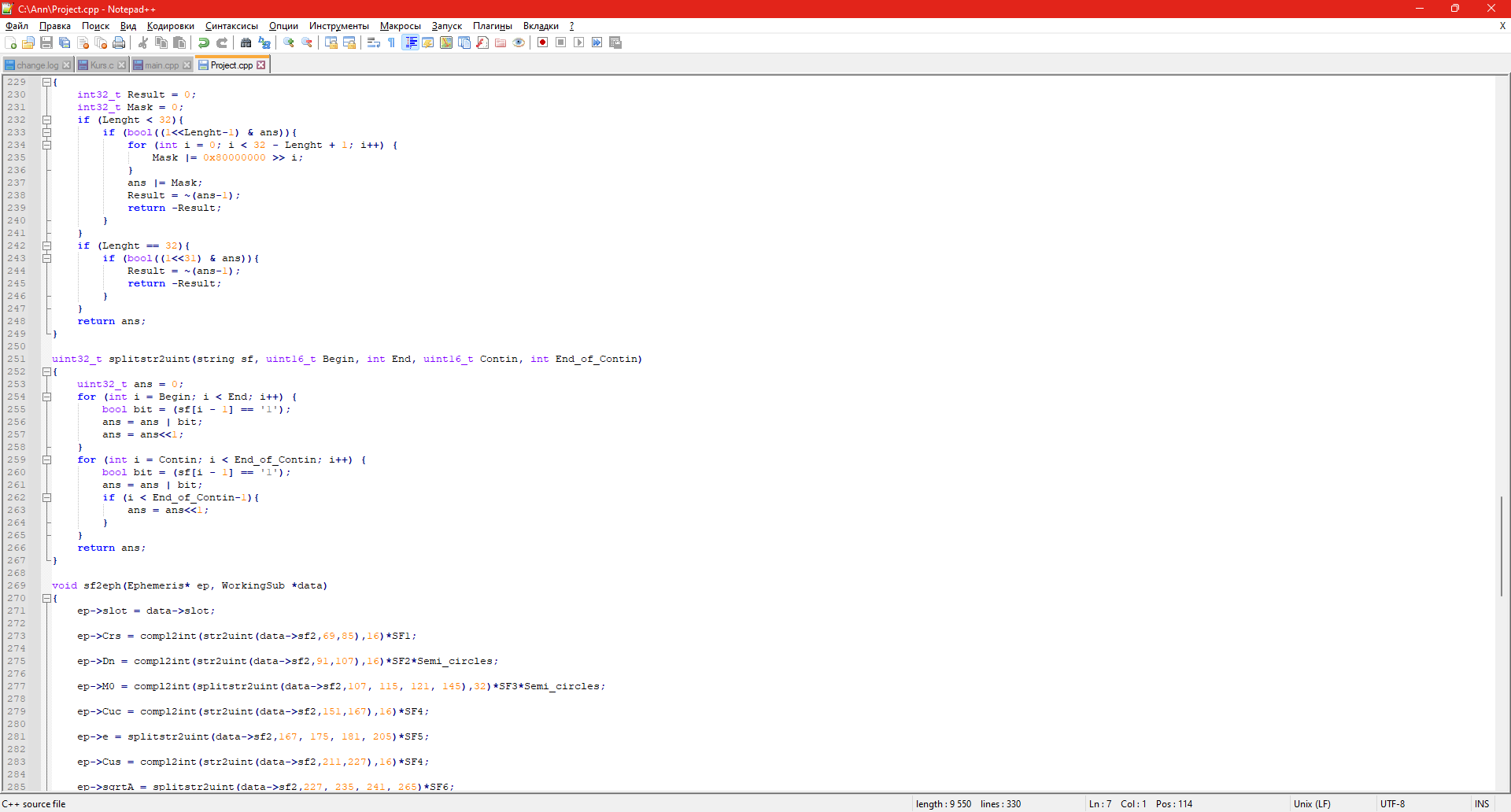
1. ГОСТ 7.32-2017. Структура и правила оформления отчета о научно-исследовательской работе.
2. Р. Лафоре Объектно-ориентированное программирование в С++. - 4-е изд. - Москва: Питер, 2004. - 923 с.
3. Interface Control Contractor: SAIC (GPS SEI) 200 N. Pacific Coast Highway, Suite 1800 El Segundo, CA 90245.
4. [Trimble GNSS Planning Online](http://www.gnssplanningonline.com/). [Электронный ресурс]. (Дата обращения: 24.04.2022г. 20:00).

# ПРИЛОЖЕНИЕ А

Ниже приведен листинг программы, разработанной в ходе выполнения первого этапа данного курсового проекта.







# ПРИЛОЖЕНИЕ Б

Ниже приведен листинг программы, разработанной в ходе выполнения второго этапа данного курсового проекта.

clc

clear all

close all

% Константы из ИКД

pi = 3.1415326535898;

mu = 3.986005e+14; % Гравитационная постоянная

Omega\_e\_dot = 7.2921151467e-5; % Скорость вращения Земли

% Эфемериды

C\_rs = -22.9688; % Амплитуда поправочного члена синусоидальной гармоники к радиусу орбиты

delta\_n = deg2rad(2.55652e-7); % Среднее отклонение движения от вычисленного значения

M\_0 = deg2rad(-94.6511); % Средняя аномалия

C\_uc = -1.21817e-6; % Амплитуда поправочного члена косинусной гармоники к аргументу широты

eks = 0.00225987; % Эксцентриситет

C\_us = 1.06804e-5; % Амплитуда поправочного члена синусоидальной гармоники к аргументу широты

rootA = 5153.62; % Корень из большой полуоси

t\_oe = 93600; % Опорное время эфемерид

C\_ic = -2.42144e-8; % Амплитуда поправочного члена косинусной гармоники к углу наклона

Omega\_0 = deg2rad(8.79464); % Долгота узла

C\_is = 1.04308e-7; % Амплитуда поправочного члена синусоидальной гармоники к углу наклона

i\_0 = deg2rad(54.6911); % Наклонение

C\_rc = 170.656; % Амплитуда поправочного члена косинусной гармоники к радиусу орбиты

omega = deg2rad(107.109); % Аргумент перигея

Omega\_dot = deg2rad(-4.49402e-7); % Скорость прямого восхождения

IDOT = deg2rad(2.53544e-8); % Скорость угла наклона

% Расчет значений элементов Кеплера

A = rootA^2; % Большая полуось

n\_0 = sqrt(mu/A^3); % Расчетное среднее движение

for i = 1:86400 % Суточный интервал рассмотрения

t = 86400+18 + i; % Количество секунд от начала текущей недели при начале рассмотрения в 00:00:01

% Время от опорной эпохи эфемерид

t\_k = t - t\_oe;

if t\_k>302400

t\_k = t\_k - 604800;

elseif t\_k<-302400

t\_k = t\_k + 604800;

end

n = n\_0 + delta\_n; % Скорректированное среднее движение

M\_k = M\_0 + n \* t\_k; % Средняя аномалия

% Эксцентрическая аномалия

E\_0 = M\_k;

E\_k = 0;

k = 0;

while abs(E\_0 - E\_k)>1e-8

E\_k = E\_0;

E\_0 = E\_0 + (M\_k - E\_0 + eks \* sin(E\_0))/(1 - eks \* cos(E\_0));

k = k + 1;

end

nu\_k = atan2((sqrt(1 - eks^2) \* sin(E\_k)/(1 - eks \* cos(E\_k))), ((cos(E\_k) - eks)/(1 - eks \* cos(E\_k)))); % Истинная аномалия

Phi\_k = nu\_k + omega; % Аргумент широты

delta\_u\_k = C\_us\*sin(2\*Phi\_k) + C\_uc\*cos(2\*Phi\_k); % Аргумент поправки на широту

delta\_r\_k = C\_rs\*sin(2\*Phi\_k) + C\_rc\*cos(2\*Phi\_k); % Коррекция радиуса

delta\_i\_k = C\_is\*sin(2\*Phi\_k) + C\_ic\*cos(2\*Phi\_k); % Коррекция наклона

u\_k = Phi\_k + delta\_u\_k; % Скорректированный аргумент широты

r\_k = A \* (1 - eks \* cos(E\_k)) + delta\_r\_k; % Скорректированный радиус

i\_k = i\_0 + delta\_i\_k + IDOT \* t\_k; % Скорректированное наклонение

% Позиция в орбитальной плоскости

x\_k\_sh = r\_k \* cos(u\_k);

y\_k\_sh = r\_k \* sin(u\_k);

Omega\_k = Omega\_0 + (Omega\_dot - Omega\_e\_dot) \* t\_k - Omega\_e\_dot \* t\_oe; % Скорректированная широта восходящего узла

% Расчет координат спутника

x\_k = x\_k\_sh \* cos(Omega\_k) - y\_k\_sh \* cos(i\_k) \* sin(Omega\_k);

y\_k = x\_k\_sh \* sin(Omega\_k) + y\_k\_sh \* cos(i\_k) \* cos(Omega\_k);

z\_k = y\_k\_sh \* sin(i\_k);

x\_current(1,i) = x\_k;

y\_current(1,i) = y\_k;

z\_current(1,i) = z\_k;

% Инерциальная система координат

theta = Omega\_e\_dot \* t\_k;

x\_k\_1 = x\_k \* cos(theta) - y\_k \* sin(theta);

y\_k\_1 = x\_k \* sin(theta) + y\_k \* cos(theta);

z\_k\_1 = z\_k;

x\_converted(1,i) = x\_k\_1;

y\_converted(1,i) = y\_k\_1;

z\_converted(1,i) = z\_k\_1;

% Данные из RTKNAVI (координаты приемника)

B = deg2rad(44.09363261); % Широта

L = deg2rad(39.00130546); % Долгота

H = 1.247; % Высота

[x(i), y(i), z(i)] = ecef2enu(x\_current(1,i),y\_current(1,i),z\_current(1,i),B,L,H,wgs84Ellipsoid,'radians'); % Отображение координат спутника относительно точки приема

% Полярная система координат

if z(i) > 0

rho(i) = sqrt(x(i)^2 + y(i)^2 + z(i)^2);

theta\_1(i) = acos(z(1,i)/rho(i));

if x(i) > 0

phi(i) = -atan(y(i)/x(i))+pi/2;

elseif (x(i)<0)&&(y(i)>0)

phi(i) = -atan(y(i)/x(i))+3\*pi/2;

elseif (x(i)<0)&&(y(i)<0)

phi(i) = -atan(y(i)/x(i))-pi/2;

end

else theta\_1(i) = NaN;

rho(i) = NaN;

phi(i) = NaN;

end

end

% Вывод значений координат

out = fopen('../out.txt', 'w+');

for i = 1:length(x\_current)

fprintf(out, '%6.0f %10.10f %10.10f %10.10f\n', i, x\_current(1,i), y\_current(1,i), z\_current(1,i));

end

fclose(out);

% Вывод графиков

R = 6378136; % Радиус Земли

[x\_sphere, y\_sphere, z\_sphere] = sphere(100);

x\_Earth=R\*x\_sphere;

y\_Earth=R\*y\_sphere;

z\_Earth=R\*z\_sphere;

figure (1)

subplot (1,1,1)

surf(x\_Earth,y\_Earth,z\_Earth);

hold on;

plot3(x\_current(1,:), y\_current(1,:), z\_current(1,:));

xlabel('x, м','FontSize',16);

ylabel('y, м','FontSize',16);

zlabel('z, м','FontSize',16);

figure (2)

subplot (1,1,1)

surf(x\_Earth,y\_Earth,z\_Earth);

hold on;

plot3(x\_converted(1,:), y\_converted(1,:), z\_converted(1,:));

xlabel('x, м','FontSize',16);

ylabel('y, м','FontSize',16);

zlabel('z, м','FontSize',16);

figure (3)

axes = polaraxes;

polarplot(axes,phi,rad2deg(theta\_1))

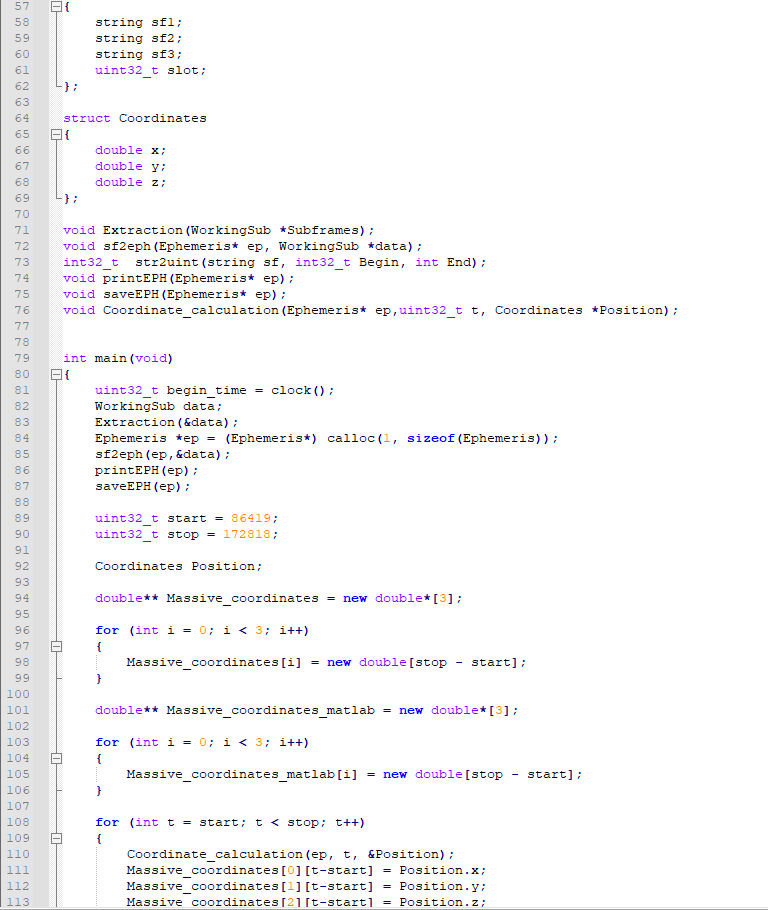
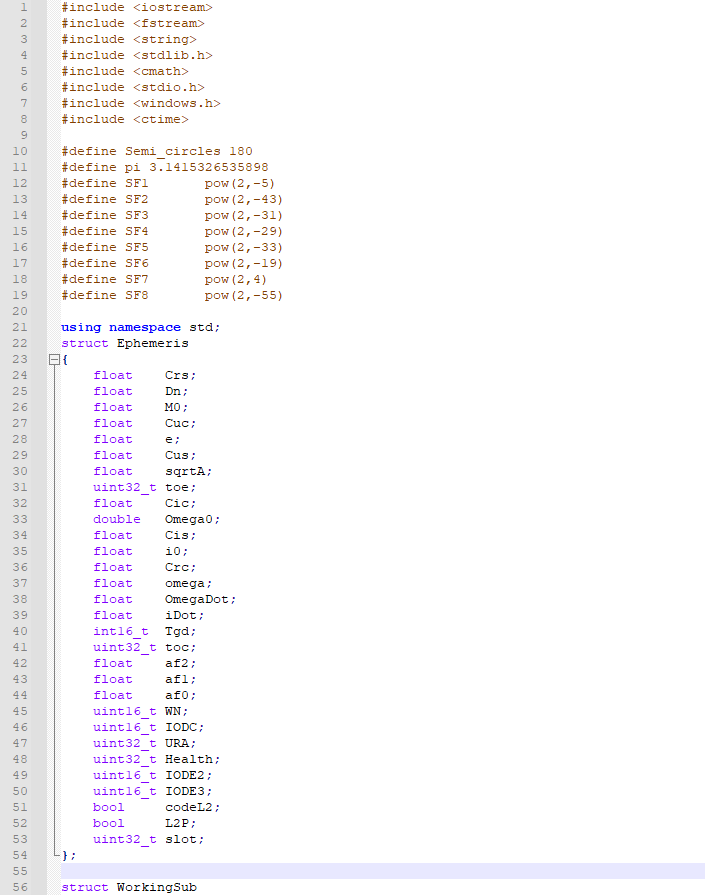
axes.ThetaDir = 'clockwise';

axes.ThetaZeroLocation = 'top';

rlim([0 80]);

# ПРИЛОЖЕНИЕ В

Ниже приведен листинг программы, разработанной в ходе выполнения третьего этапа данного курсового проекта.

****

