Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Национальный исследовательский университет «МЭИ»

|  |  |  |  |
| --- | --- | --- | --- |
| Институт: | *ИРЭ* | Кафедра: | *Радиотехнических систем* |
| Специальность: | | *11.05.01 Радиоэлектронные системы и комплексы* | |

**ОТЧЕТ по практике**

|  |  |
| --- | --- |
| **Наименование практики:** | Производственная практика: научно-исследовательская работа |

**СТУДЕНТ**



|  |  |
| --- | --- |
|  | / Тасканов В.Е. / |
| *(подпись )* | (*Фамилия и инициалы*) |

|  |  |
| --- | --- |
| Группа | ЭР-15-16 |
|  | *(номер учебной группы)* |

**ПРОМЕЖУТОЧНАЯ АТТЕСТАЦИЯ ПО ПРАКТИКЕ**

|  |
| --- |
| **ХОРОШО** |
| *(отлично, хорошо, удовлетворительно, неудовлетворительно, зачтено, не зачтено)* |

|  |  |
| --- | --- |
|  | / Куликов Р.С. / |
| *(подпись )* | (*Фамилия и инициалы члена комиссии*) |

|  |  |
| --- | --- |
|  | / Шатилов А.Ю. / |
| *(подпись )* | (*Фамилия и инициалы члена комиссии*) |

**Москва**

**2021**

СПИСОК ИСПОЛНИТЕЛЕЙ

студент Тасканов В.Е.

**Содержание**

[**ГЛАВА 1 ДОБАВЛЕНИЕ РАСЧЕТА КООРДИНАТ НС** 5](#_Toc74479281)

[1.1. Алгоритм расчета для ГНСС GPS 6](#_Toc74479282)

[1.1.2. Алгоритм расчета координат 8](#_Toc74479283)

[1.3. Алгоритм расчета для ГНСС ГЛОНАСС 10](#_Toc74479284)

[1.3.2. Алгоритм расчета координат 12](#_Toc74479285)

[1.4. Алгоритм расчета ионосферной погрешности 15](#_Toc74479286)

[**ГЛАВА 2 РЕАЛИЗАЦИЯ АЛГОРИТМОВ В ПРОГРАММЕ** 18](#_Toc74479287)

[2.1. Скачивание файла 18](#_Toc74479288)

[2.2. Обработка файла 18](#_Toc74479289)

[2.2.1. Обработка файла GPS 18](#_Toc74479290)

[2.2.1. Обработка файла ГЛОНАСС 18](#_Toc74479291)

[2.3. Расчет координат 19](#_Toc74479292)

[2.4. Расчет времени 19](#_Toc74479293)

[2.3. Изменение интерфейса программы 20](#_Toc74479294)

[2.4. Необходимые файлы для сборки проекта 21](#_Toc74479295)

[**ГЛАВА 3 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ** 25](#_Toc74479296)

[**ЗАКЛЮЧЕНИЕ** 31](#_Toc74479297)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 31](#_Toc74479298)

[**ПРИЛОЖЕНИЕ** 32](#_Toc74479299)

Введение

Спутниковые радионавигационные системы (СРНС) являются самыми точными системами по определению координат потребителя. Они стали важной частью в различных сферах нашей жизни. Наиболее распространенными являются системы ГЛОНАСС (Россия), GPS (США), Galileo (Евросоюз).

В 10 семестре стояла **цель работы** – откорректировать предыдущую программу, добавить функцию расчета координат НКА по данным альманахов систем ГЛОНАСС и GPS, добавить функцию расчета ионосферных погрешностей по углу возвышении спутника и текущему календарному времени года, и с учетом всех изменений произвести оценку координаты потребителя с учетом ошибок SISRE.

В рамках данной цели решаются следующие задачи:

1. Изменения предыдущего алгоритма определения координат НКА
2. Теоретическое изучение вносимых погрешностей от ионосферы
3. Добавления алгоритма расчета ионосферных погрешностей по углу возвышении спутника и текущему календарному времени года
4. Нахождения ошибки оценивания координаты потребителя с учетом ошибок SISRE и ионосферных погрешностей.

# **ГЛАВА 1 ДОБАВЛЕНИЕ РАСЧЕТА КООРДИНАТ НС**

В данной работе изменим рассчет координата НС, теперь они рассчитываются по данным альманахам соответствующих группировок для созвездий ГЛОНАСС и GPS.

Алгоритм расчета координат в программе будет следующий:

* Скачиваем файл с данными альманаха,
* Обрабатываем файл с помощью новой функции обработки,
* Рассчитываем координаты

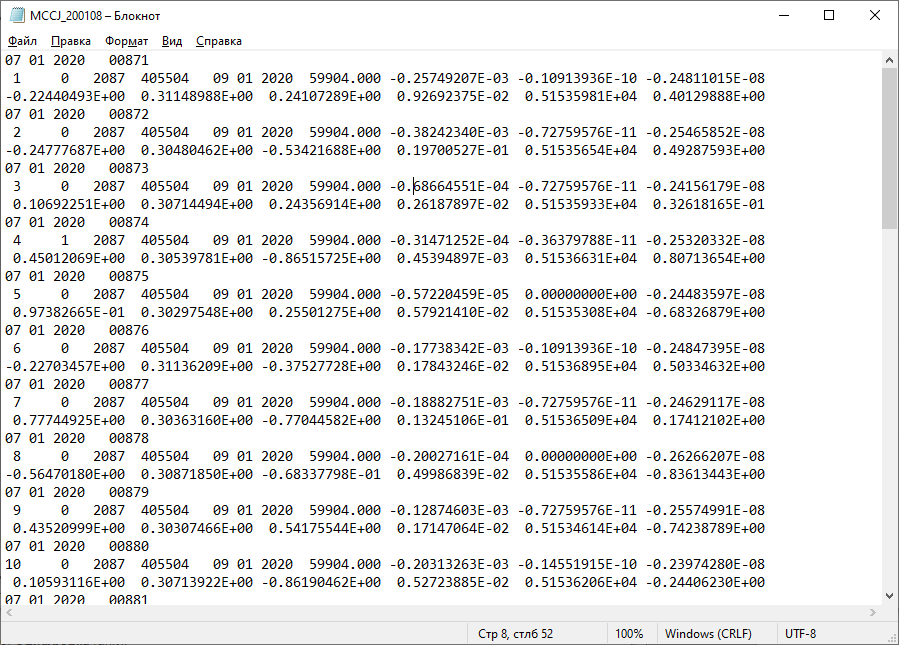
Файл будет скачивать с сервера «инфомационно-аналитечкского центра коррдинатно-временного и навигационного обеспечения, по адресу: «ftp://ftp.glonass-iac.ru/MCC/ALMANAC/», где далее следует выбор года и даты нужного альманаха. На сервере содержится два файла с разными расширениями – agp ( для ГНСС GPS) и agl (для ГНСС ГЛОНАСС).

Скачав файл необходимо его оцифровать (перенести нужные данные в программу для реализации последующих алгоритмов).

Оцифровка файлов для GPS и ГЛОНАСС значительно отличается, поэтому разделим их на разные функции.

## 1.1. Алгоритм расчета для ГНСС GPS

В файле с расширением – agp, содержатся альманахи, записанные в виде строк:



1. Пример скаченного файла с расширением agp

Где, строка 1, соответствует:

|  |  |
| --- | --- |
| 1 | Число получения альманаха |
| 2 | месяц получения альманаха |
| 3 | год получения альманаха |
| 4 | время получения альманаха от начала суток, с UTC |

Строка 2

|  |  |
| --- | --- |
| 1 | номер PRN |
| 2 | обобщенный признак здоровья (0 - здоров) |
| 3 | неделя GPS (альманаха) (номер недели полный) |
| 4 | время недели GPS, с (альманаха) (количество секунд от начала недели) |
| 5 | число |
| 6 | месяц |
| 7 | год |
| 8 | время альманаха, с |
| 9 | поправка времени КА GPS относительно системного времени, с, |
| 10 | скорость поправки времени КА GPS относительно системного времени, с/с |
| 11 | Om0 - скорость долготы узла, полуциклы/c, |

Строка 3

|  |  |
| --- | --- |
| 1 | Om0 - долгота узла, полуциклы, |
| 2 | I - наклонение, полуциклы, |
| 3 | w - аргумент перигея, полуциклы, |
| 4 | E – эксцентриситет, |
| 5 | SQRT(A) - корень из большой полуоси, м\*\*0.5, sqrt(A0) |
| 6 | M0 - средняя аномалия, полуциклы, |

1.1.2. Алгоритм расчета координат

Далее полученные значения подставляются в алгоритм расчета координат, который возьмем из ИКД GPS:

1. Определим время, отсчитываемое от опорной эпохи эфемерид:



1. Определим среднее движение:



1. Определим скорректированное среднее движение:



1. Определим среднюю аномалию:



1. Решим уравнение Кеплера минимум 3-мя итерациями и определим :



1. Определим истинную аномалию:



1. Определим скорректированный радиус орбиты спутника:





1. Определим аргумент широты:





1. Определим координаты НС в орбитальной плоскости:



1. Определим скорректированную долготу восходящего узла  определяется из соотношения:





1. Определим скорректированное наклонение орбиты спутника

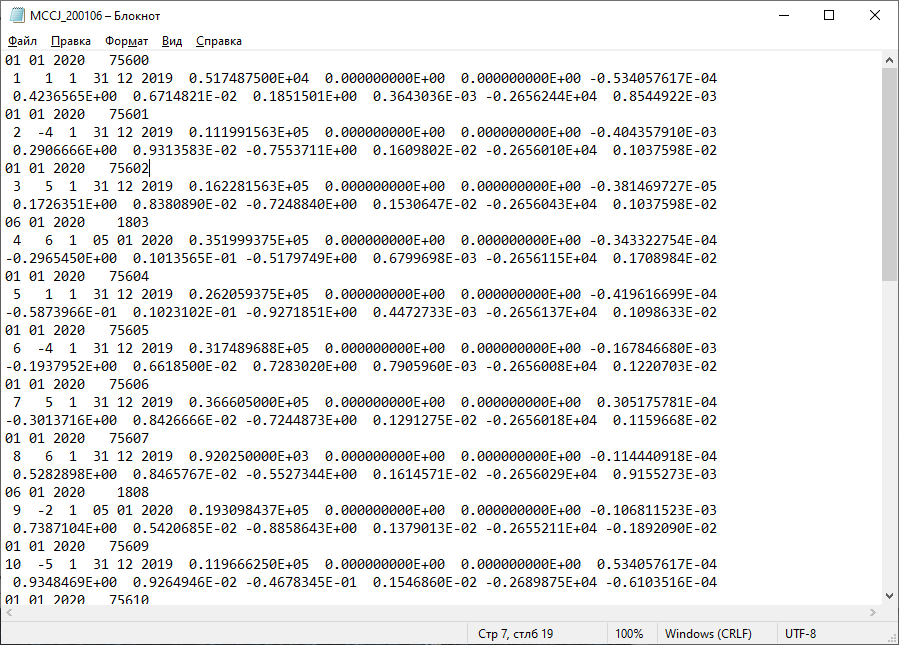


1. Определим координаты НС в геоцентрической системе координат:



1.3. Алгоритм расчета для ГНСС ГЛОНАСС

В файле с расширением – agl, содержатся альманахи, записанные в виде строк:



1. Пример скаченного файла с расширением agl

Где, строка 1, соответствует:

|  |  |
| --- | --- |
| 1 | Число получения альманаха |
| 2 | месяц получения альманаха |
| 3 | год получения альманаха |
| 4 | время получения альманаха от начала суток, с UTC |

Строка 2

|  |  |
| --- | --- |
| 1 | номер КА в группировке |
| 2 | номер частотного слота (-7 - 24) |
| 3 | признак здоровья по альманаху (0 - 1) |
| 4 | число |
| 5 | месяц |
| 6 | год |
| 7 | время прохождения первого узла, на которое все дано, с |
| 8 | поправка ГЛОНАСС-UTC, с |
| 9 | поправка GPS-ГЛОНАСС, с |
| 10 | поправка времени КА ГЛОНАСС относительно системного времени, с |

Строка 3

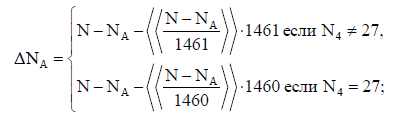
|  |  |
| --- | --- |
| 1 | Lam - долгота узла, полуциклы |
| 2 | dI - коррекция наклонения, полуциклы |
| 3 | w - аргумент перигея, полуциклы |
| 4 | E - эксцентриситет |
| 5 | dT - поправка к драконическому периоду, с |
| 6 | dTT - поправка к драконическому периоду, с/виток |

1.3.2. Алгоритм расчета координат

Далее полученные значения подставляются в алгоритм расчета координат, который возьмем из ИКД ГЛОНАСС:

1. Определяется интервал прогноза в секундах:



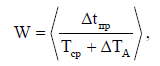
Где: 

N – календарный номер суток внутри четырехлетнего периода, начиная с високосного года, на которых находится заданный момент времени ti в секундах по шкале МДВ;

NA – календарный номер суток по шкале МДВ внутри четырехлетнего интервала, передаваемый НКА в составе неоперативной информации;

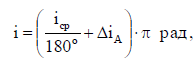
 – вычисление целого, ближайшего к x.

1. Рассчитывается количество целых витков W на интервале прогноза:

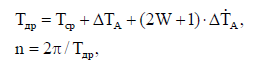


где  выделение целой части x;

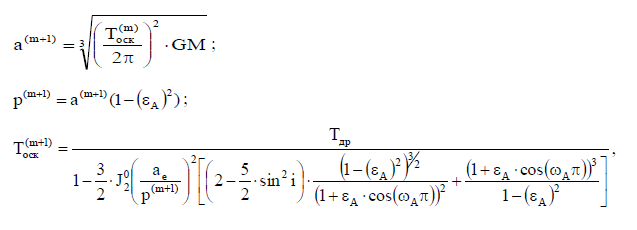
1. Определяется текущее наклонение:



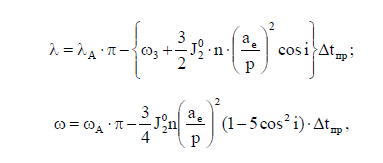
1. Определяются средний драконический период на витке W+1 и среднее движение:



1. Методом последовательных приближений m = 0, 1, 2… рассчитывается большая полуось орбиты a:



1. Определяются текущие значения долготы восходящего узла орбиты и аргумента перигея с учетом их векового движения под влиянием сжатия Земли:



1. Рассчитывается значение средней долготы на момент прохождения текущего восходящего узла:



Где 

1. Определяется текущее значение средней долготы НКА:



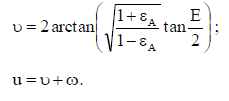
1. Определяется эксцентрическая аномалия путем решения уравнения Кеплера



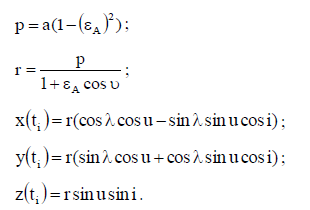
Как правило, используется схема последовательных приближений m = 0, 1, 2, и т.д.:



1. Вычисляются истинная аномалия и аргумент широты НКА u:



1. Рассчитываются координаты центра масс НКА в геоцентрической прямоугольной пространственной системе координат:



## 1.4. Алгоритм расчета ионосферной погрешности

Воспользуемся алгоритмом расчета из ИКД GPS

Модель коррекции модели ионосферы



Определим 



Где  - коэффициенты кубического уравнения, представляющие амплитуду вертикальной задержки

Определим фазу



Определим 



Где  - коэффициенты кубического уравнения, представляющие период модели

Определим коэффициент наклона



Где  - угол возвышения между пользователем и спутником

Определим геомагнитную широту земной проекции точки пересечения ионосферы (средняя высота ионосферы, предполагаемая 350 км)

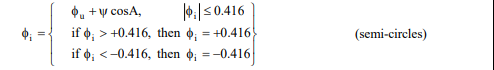


Определим геодезическая долгота земной проекции точки пересечения ионосферы



Определим геодезическая широта земной проекции точки пересечения ионосферы





Определим центральный угол Земли между положением пользователя и проекцией на землю точки пересечения ионосферы



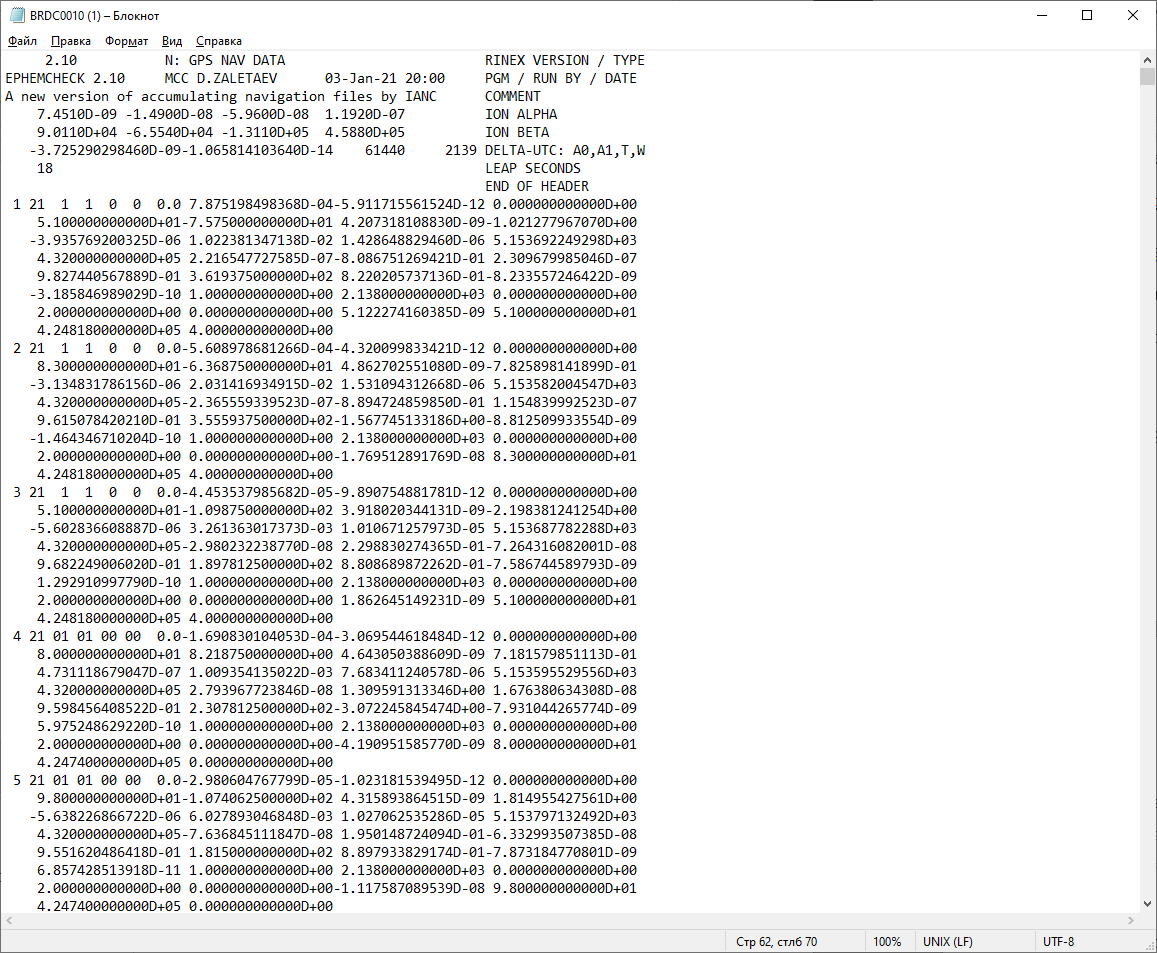
Определим локальное время:



Где ,

 - вычисленное системное время приемника

Значения коэффициентов ,берется из файла, который будем скачивать с сервера «инфомационно-аналитечкского центра коррдинатно-временного и навигационного обеспечения, по адресу: «ftp://ftp.glonass-iac.ru/MCC/BRDC/», расширение файла 21n



1. Пример скаченного файла с расширением 21n

Необходимые нам коэффициенты расположены на 4 и 5 строчках.

# **ГЛАВА 2 РЕАЛИЗАЦИЯ АЛГОРИТМОВ В ПРОГРАММЕ**

2.1. Скачивание файла

Для скачивания файлов модернизируем раннее созданный алгоритм «download» и для удобства последующих вызовов перенесем его в отдельный файл функции, который назовем: «FTPdownload», на вход которой подается разные пути и названия файла.

Функция содержит заголовочный файл – «FTPdownload.h», в котором хранятся применяемые классы и методы, а также файл с кодом реализации скачивания файла – «FTPdownload.CPP»

## 2.2. Обработка файла

Для обработки файлов также создадим отдельные функции, для ГЛОНАСС – «parserGLNS», а для GPS – «parserGPS»

2.2.1. Обработка файла GPS

Алгоритм обработки файла строится на методе «fscanf», которая обрабатывает последовательно каждое заданное значение, далее переносим полученные значения в массив значений «almanax\_GPS».

Функция содержит заголовочный файл – «parserGPS.h», в котором хранятся применяемые классы и методы, а также файл с кодом реализации обработки файла – «parserGPS.С»

### 2.2.1. Обработка файла ГЛОНАСС

Алгоритм обработки файла строится на методе «fscanf», которая обрабатывает последовательно каждое заданное значение, далее переносим полученные значения в массив значений «almanax\_GLNS».

Функция содержит заголовочный файл – «parserGLNS.h», в котором хранятся применяемые классы и методы, а также файл с кодом реализации обработки файла – «parserGLNS.С»

## 2.3. Расчет координат

Расчет координат для ГЛОНАСС и GPS выведем также в отдельные функции.

Для ГЛОНАСС функция принимает название – «ephemeridsGLNS», расчет соответствует формулам из п.1.3.2;

Функция содержит заголовочный файл – «ephemeridsGLNS.h», в котором хранятся применяемые классы и методы, а также файл с кодом реализации обработки файла – «ephemeridsGLNS.cpp»

Для GPS функция имеет название – «ephemerids», расчет соответствует формулам из п.1.1.2;

Функция содержит заголовочный файл – «ephemerids.h», в котором хранятся применяемые классы и методы, а также файл с кодом реализации обработки файла – «ephemerids.cpp»

## 2.4. Расчет времени

В процессе расчета координат возникнет проблема – получения времени расчета на которое нужно спрогнозировать координаты.

Для этого запишем класс – «timeCalc», в котором будет производиться перерасчет времени в нужный формат для трех ГНСС – ГЛОНАСС, GPS и GALILEO.

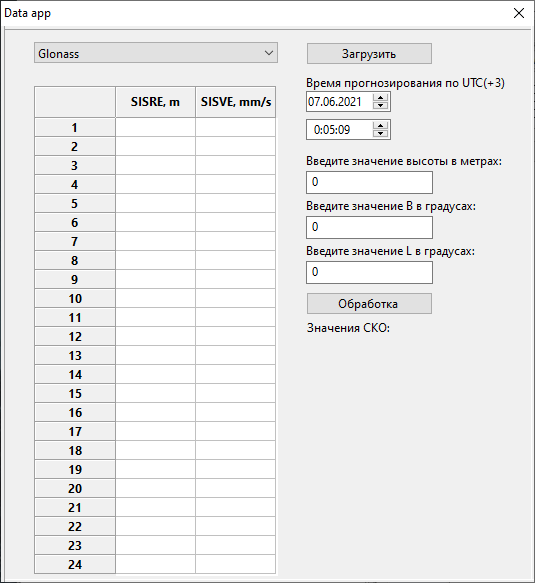
Для создания класса необходимо подать начальные значения: число, месяц, год, часы, минуты, секунды и миллисекунды.

Далее начальные значения преобразуются в секунды, с помощью встроенной библиотеки «ctime», а также подсчитывается количество поправок ко времени, для расчета в системе GPS и GALILEO.

В классе имеется три функции расчета времени:

* «timeGLNS» - для расчета времени в системе ГЛОНАСС,
* «timeGPS» - для расчета времени в системе GPS,
* «timeGLL» - для расчета времени в системе GALILEO.

2.3. Изменение интерфейса программы



1. Изменённый интерфейс программы

Добавил две новых формы:

* Ввод даты

Ввод даты производится с помощью встроенной формы в библиотеку wxWidgets – wxDatePickerCtrl, который реализован в виде небольшого окна, показывающего текущую дату, элемент управления можно редактировать с помощью клавиатуры, а также с помощью мышки

* Ввод времени

Ввод даты производится с помощью встроенной формы в библиотеку wxWidgets – wxTimePicerCtrl, который реализован в виде небольшого окна, показывающего текущее время, элемент управления можно редактировать с помощью клавиатуры, а также с помощью мышки

2.4. Необходимые файлы для сборки проекта

К отчету прикреплены 22 приложения, в которых содержатся основные файлы кода программы, необходимые для сборки проекта. Так файл: “dataMain.cpp”, код описан в приложение 1, содержит в себе основной алгоритм настройки окна приложения, с помощью библиотек wxWidgets, также содержит алгоритм скачивания файла с сервера и заполнения таблицы данными SISRE и SISVE, путь к файлу: “…\data\dataMain.cpp”.

Файл: “parser.c”, код описан в приложение 2, содержит в себе обработку скаченного файла с сервера, с помощью алгоритма из приложения 1 [2]. Под обработкой подразумевается фильтрование нужной нам информации – значения SISRE и SISVE для определенного спутника, путь к файлу: “…\data\parser.c”.

Файл: “parser.h”, код описан в приложение 3, содержит в себе обработчик массива SISerr, для использования этого массива в приложение 1 [2], данный обработчик необходим, так как приложение 2 написано на языке «С», а приложение 1 на языке «С++», путь к файлу: “…\data\parser.h”.

Файл: “data.cbp”, код описан в приложение 4 – это необходимый файл для сборки проекта, в котором прописан используемый компилятор, библиотеки, а также все необходимые заголовочные файлы, путь к файлу: “…\data \data.cbp”.

Файл: “dataMain.h”, код описан в приложение 5 – это заголовочный файл, в котором хранятся применяемые классы и методы, путь к файлу: “…\data\dataMain.h”.

Файл “xyz2enu.cpp”, код описан в приложение 6, в файле реализована функция перевода из геодезической системы координат в топоцентрические координаты (ENU), путь к файлу: “…\data \xyz2enu.cpp”

Файл “xyz2enu.h”, код описан в приложение 7 - это заголовочный файл, в котором объявляются применяемые классы и методы для функции перевода из геодезической системы координат в топоцентрические координаты (ENU) : “…\data \xyz2enu.h”

Файл “ephemeridsGLNS.cpp”, код описан в приложение 8, в файле реализована функция описание движения спутников ГЛОНАСС по орбитам и нахождения координат спутников ГЛОНАСС в определенный момент, путь к файлу: “…\data \ephemeridsGLNS.cpp ”

Файл “ ephemeridsGLNS.h”, код описан в приложение 9 - это заголовочный файл, в котором объявляются применяемые классы и методы для функции “ephemerids”, путь к файлу: “…\data \ ephemeridsGLNS.h ”

Файл “angle.cpp”, код описан в приложение 10, в файле реализована функция расчета видимости спутников, путь к файлу: “…\data \angle.cpp ”

Файл “ angle.h ”, код описан в приложение 11 - это заголовочный файл, в котором объявляются применяемые классы и методы расчета видимости спутников, путь к файлу: “…\data \ angle.h ”

Файл: “datadiaslog.wxs”, код описан в приложение 12 – это файл описания графического пользовательского интерфейса для плагина wxSmith, путь к файлу: “…\data\ wxsmith\datadiaslog.wxs”.

Файл “timeCalc.cpp”, код описан в приложение 13, в файле реализован класс перевода времени для трех систем СРНС, путь к файлу: “…\data \timeCalc.cpp ”

Файл “timeCalc.h”, код описан в приложение 14 - это заголовочный файл, в котором объявляются применяемые классы и методы для класса перевода времени для трех систем СРНС : “…\data \timeCalc.h”

Файл “FTPdownload.cpp”, код описан в приложение 15, в файле реализована функция скачивания файлов с сервера, путь к файлу: “…\data \FTPdownload.cpp ”

Файл “ FTPdownload.h ”, код описан в приложение 16 - это заголовочный файл, в котором объявляются применяемые классы и методы скачивания файлов с сервера, путь к файлу: “…\data \ FTPdownload.h ”

Файл: “parserGLNS.c”, код описан в приложение 17, содержит в себе обработку скаченного файла с сервера. Под обработкой подразумевается фильтрование нужной нам информации – значения SISRE и SISVE для определенного спутника, путь к файлу: “…\data\parserGLNS.c”.

Файл: “ parserGLNS.h”, код описан в приложение 18, содержит в себе обработчик массива SISerr, данный обработчик необходим, так как приложение 2 написано на языке «С», а приложение 1 на языке «С++», путь к файлу: “…\data\parserGLNS.h”.

Файл: “parserGPS.c”, код описан в приложение 19, содержит в себе обработку скаченного файла с сервера. Под обработкой подразумевается фильтрование нужной нам информации – значения SISRE и SISVE для определенного спутника, путь к файлу: “…\data\parserGPS.c”.

Файл: “ parserGPS.h”, код описан в приложение 20, содержит в себе обработчик массива SISerr, данный обработчик необходим, так как приложение 2 написано на языке «С», а приложение 1 на языке «С++», путь к файлу: “…\data\parserGPS.h”.

Файл “ephemerids.cpp”, код описан в приложение 21, в файле реализована функция описание движения спутников GPS по орбитам и нахождения координат спутников GPS в определенный момент, путь к файлу: “…\data \ephemerids.cpp ”

Файл “ephemerids.h”, код описан в приложение 22 - это заголовочный файл, в котором объявляются применяемые классы и методы для функции “ephemerids”, путь к файлу: “…\data \ ephemerids.h ”

**ГЛАВА 3 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

*Минимальные требования:*

Для запуска программы необходимо иметь windows 7/10 и подключенное устройство к интернету.

*Инструкция:*

1. Запустите программу “Data.exe” от имени администратора

Если программа не запустится отключите антивирус.

У вас появится диалоговое окно:

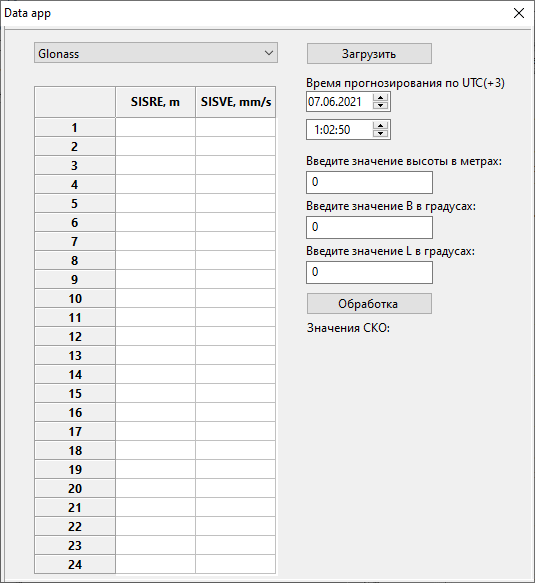


Рисунок 3.1. Интерфейс программы

1. Выберите необходимую вам НС:

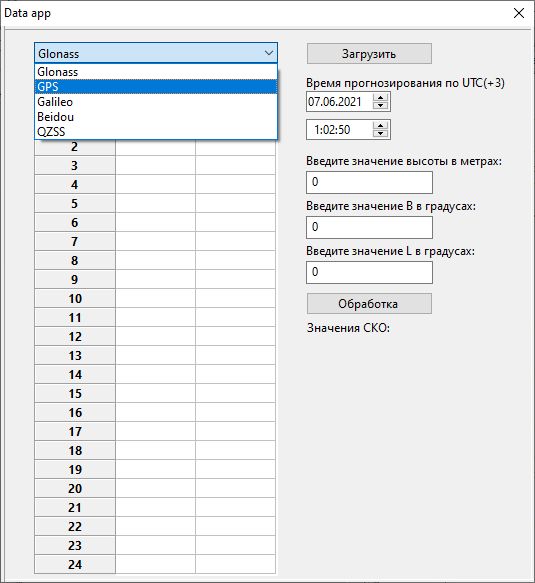


Рисунок 3.2. Интерфейс выбора НС

1. Нажмите кнопку «Загрузить»

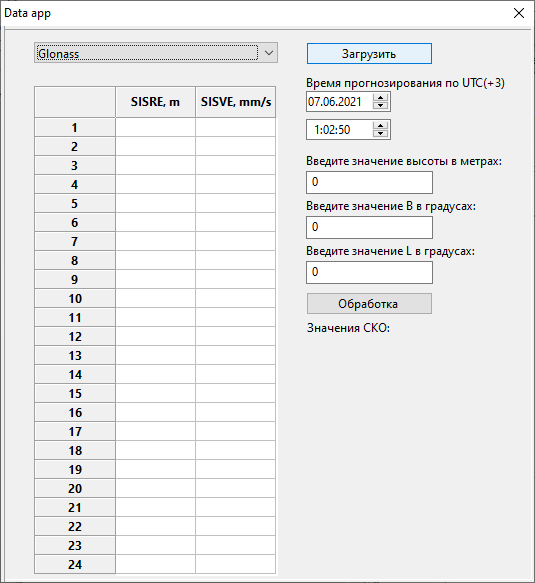


Рисунок 3.3. Интерфейс выбранной НС

Получили значения SISRE и SISVE для каждого спутника, если

значения равны 0.00, то данный спутник отсутствует.

Если при загрузке возникла ошибка существует два варианта решения ее:

* Отключите антивирус,
* Включите брандмауэр.

1. Для того, чтобы скачать данные для других НС, перейдите к п. 2 инструкции.

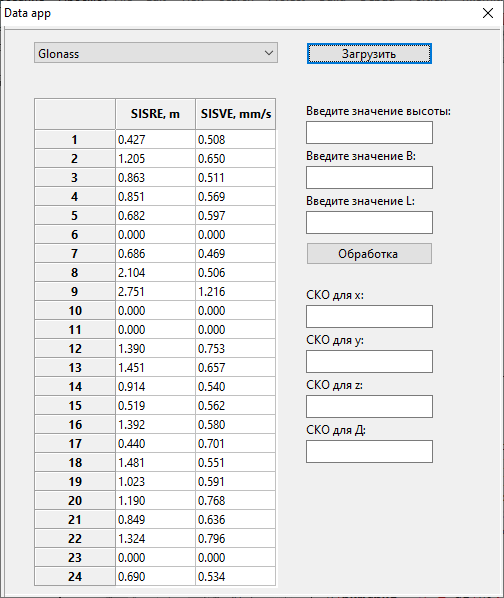


Рисунок 3.4. Полученные значения

1. Далее вводим значения:

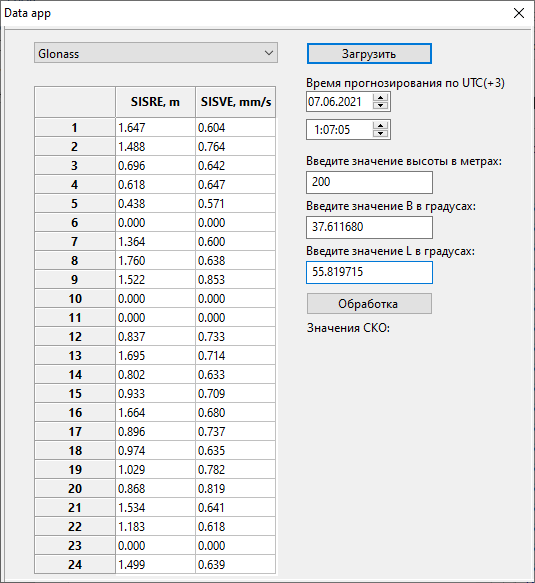


Рисунок 3.5. Ввод значений в формы

1. Нажимаем кнопку обработка:

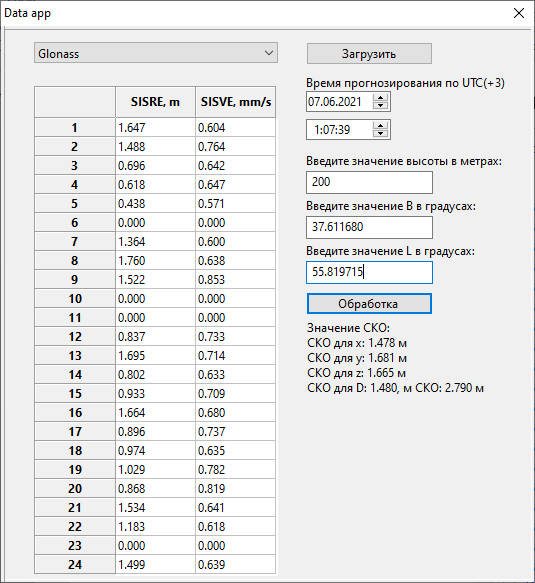


Рисунок 3.6. Полученные значения

Получаем значения СКО.

1. Чтобы закрыть программу нажмите крестик в диалоговом окне

**ЗАКЛЮЧЕНИЕ**

Получил значения СКО для x,y,z - координаты и погрешности временной шкалы, которые рассчитываются по данным альманахам соответствующих группировок для созвездий ГЛОНАСС и GPS.

Не все задачи удалось выполнить: добавить алгоритм расчета ионосферных и тропосферных погрешностей, из-за затеявшегося изменения функции расчета координат НКА по данным альманахов, так как для изменения ее пришлось написать и переписать ряд функций: функции обработки файлов с альманахами, перерасчет времени, алгоритм скачивания файлов с сервера.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1]. Сервер «инфомационно-аналитечкского центра коррдинатно-временного и навигационного обеспечения «ftp://glonass-iac.ru» »

[2]. Отчет НИР за 9 семестр

**ПРИЛОЖЕНИЕ**

Приложение 1

|  |
| --- |
| #include "dataMain.h"  #include "FTPdownl.h"  #include "parser.h"  #include "parserGPS.H"  #include "angle.h"  #include "ephemerids.h"  #include "xyz2enu.h"  #include "parserGLNS.H"  #include "ephemeridsGLNS.h"  #include "timeCalc.h"  #include <wx/msgdlg.h>  #include <windows.h>  #include <wininet.h>  #include <iostream>  #include <string>  #include <stdio.h>  #include <armadillo>  #include <fstream>  #include <wx/string.h>  #include <wx/textfile.h>  #include <wx/dialog.h>  #include <wx/msgdlg.h>  #include <wx/spinctrl.h>  #include <wx/intl.h>  #include <wx/settings.h>  #define SQUARE(val) val \* val  using namespace std;  using namespace arma;  //(\*InternalHeaders(dataDialog)  #include <wx/intl.h>  #include <wx/settings.h>  #include <wx/string.h>  //\*)  //helper functions  enum wxbuildinfoformat  {  short\_f, long\_f  };  wxString wxbuildinfo(wxbuildinfoformat format)  {  wxString wxbuild(wxVERSION\_STRING);  if (format == long\_f )  {  #if defined(\_\_WXMSW\_\_)  wxbuild << \_T("-Windows");  #elif defined(\_\_UNIX\_\_)  wxbuild << \_T("-Linux");  #endif  #if wxUSE\_UNICODE  wxbuild << \_T("-Unicode build");  #else  wxbuild << \_T("-ANSI build");  #endif // wxUSE\_UNICODE  }  return wxbuild;  }  //(\*IdInit(dataDialog)  const long dataDialog::ID\_DATEPICKERCTRL1 = wxNewId();  const long dataDialog::ID\_CHOICE1 = wxNewId();  const long dataDialog::ID\_BUTTON2 = wxNewId();  const long dataDialog::ID\_NOTEBOOK1 = wxNewId();  const long dataDialog::ID\_BUTTON1 = wxNewId();  const long dataDialog::ID\_TEXTCTRL1 = wxNewId();  const long dataDialog::ID\_TEXTCTRL2 = wxNewId();  const long dataDialog::ID\_TEXTCTRL3 = wxNewId();  const long dataDialog::ID\_STATICTEXT1 = wxNewId();  const long dataDialog::ID\_STATICTEXT2 = wxNewId();  const long dataDialog::ID\_STATICTEXT3 = wxNewId();  const long dataDialog::ID\_STATICTEXT4 = wxNewId();  const long dataDialog::ID\_TIMEPICKERCTRL1 = wxNewId();  const long dataDialog::ID\_BUTTON4 = wxNewId();  const long dataDialog::ID\_STATICTEXT5 = wxNewId();  const long dataDialog::ID\_SASHWINDOW1 = wxNewId();  //\*)  const long dataDialog::ID\_GRID = wxNewId();  BEGIN\_EVENT\_TABLE(dataDialog,wxDialog)  //(\*EventTable(dataDialog)  //\*)  END\_EVENT\_TABLE()  /\*  bool download(LPCSTR server, LPCSTR login, LPCSTR pass, LPCSTR local\_file, LPCSTR remote\_file)  {  bool status;  HINTERNET hOpen, hConnection;  hOpen = InternetOpen(NULL, INTERNET\_OPEN\_TYPE\_DIRECT, NULL, NULL, 0);  if (hOpen == NULL)  return false;  hConnection = InternetConnectA(hOpen, server, 21, login, pass, INTERNET\_SERVICE\_FTP, INTERNET\_FLAG\_PASSIVE, 0 );  if (hConnection == NULL)  {  InternetCloseHandle(hOpen);  return false;  }  status=FtpGetFileA(hConnection, local\_file, remote\_file, true, 0, FTP\_TRANSFER\_TYPE\_UNKNOWN, 0);  InternetCloseHandle(hConnection);  InternetCloseHandle(hOpen);  return status;  }  \*/  dataDialog::dataDialog(wxWindow\* parent,wxWindowID id)  {  //(\*Initialize(dataDialog)  Create(parent, wxID\_ANY, \_("Data app"), wxDefaultPosition, wxDefaultSize, wxDEFAULT\_DIALOG\_STYLE, \_T("wxID\_ANY"));  SetClientSize(wxSize(533,556));  SetMinSize(wxSize(-1,-1));  SetMaxSize(wxSize(-1,-1));  SetBackgroundColour(wxSystemSettings::GetColour(wxSYS\_COLOUR\_INACTIVEBORDER));  SashWindow1 = new wxSashWindow(this, ID\_SASHWINDOW1, wxPoint(56,40), wxSize(480,504), wxSW\_3D|wxCLIP\_CHILDREN, \_T("ID\_SASHWINDOW1"));  SashWindow1->SetForegroundColour(wxSystemSettings::GetColour(wxSYS\_COLOUR\_WINDOWTEXT));  SashWindow1->SetBackgroundColour(wxSystemSettings::GetColour(wxSYS\_COLOUR\_MENUBAR));  DatePickerCtrl1 = new wxDatePickerCtrl(SashWindow1, ID\_DATEPICKERCTRL1, wxDefaultDateTime, wxPoint(305,65), wxSize(85,21), wxDP\_DEFAULT|wxDP\_SHOWCENTURY, wxDefaultValidator, \_T("ID\_DATEPICKERCTRL1"));  DatePickerCtrl1->SetForegroundColour(wxSystemSettings::GetColour(wxSYS\_COLOUR\_ACTIVEBORDER));  DatePickerCtrl1->SetBackgroundColour(wxSystemSettings::GetColour(wxSYS\_COLOUR\_HIGHLIGHT));  Choice1 = new wxChoice(SashWindow1, ID\_CHOICE1, wxPoint(33,16), wxSize(244,21), 0, 0, 0, wxDefaultValidator, \_T("ID\_CHOICE1"));  Choice1->SetSelection( Choice1->Append(\_("Glonass")) );  Choice1->Append(\_("GPS"));  Choice1->Append(\_("Galileo"));  Choice1->Append(\_("Beidou"));  Choice1->Append(\_("QZSS"));  Button2 = new wxButton(SashWindow1, ID\_BUTTON2, \_("Загрузить"), wxPoint(305,16), wxSize(127,23), 0, wxDefaultValidator, \_T("ID\_BUTTON2"));  Notebook1 = new wxNotebook(SashWindow1, ID\_NOTEBOOK1, wxPoint(124,214), wxDefaultSize, 0, \_T("ID\_NOTEBOOK1"));  Down = new wxButton(SashWindow1, ID\_BUTTON1, \_("Обработка "), wxPoint(305,266), wxSize(127,23), 0, wxDefaultValidator, \_T("ID\_BUTTON1"));  TextCtrlH = new wxTextCtrl(SashWindow1, ID\_TEXTCTRL1, \_("0"), wxPoint(305,145), wxSize(127,-1), 0, wxDefaultValidator, \_T("ID\_TEXTCTRL1"));  TextCtrlB = new wxTextCtrl(SashWindow1, ID\_TEXTCTRL2, \_("0"), wxPoint(305,190), wxSize(127,-1), 0, wxDefaultValidator, \_T("ID\_TEXTCTRL2"));  TextCtrlL = new wxTextCtrl(SashWindow1, ID\_TEXTCTRL3, \_("0"), wxPoint(305,235), wxSize(127,-1), 0, wxDefaultValidator, \_T("ID\_TEXTCTRL3"));  StaticText1 = new wxStaticText(SashWindow1, ID\_STATICTEXT1, \_("Введите значение высоты в метрах:"), wxPoint(305,127), wxDefaultSize, 0, \_T("ID\_STATICTEXT1"));  StaticText2 = new wxStaticText(SashWindow1, ID\_STATICTEXT2, \_("Введите значение B в градусах:"), wxPoint(305,172), wxDefaultSize, 0, \_T("ID\_STATICTEXT2"));  StaticText3 = new wxStaticText(SashWindow1, ID\_STATICTEXT3, \_("Введите значение L в градусах:"), wxPoint(305,217), wxDefaultSize, 0, \_T("ID\_STATICTEXT3"));  StaticText4 = new wxStaticText(SashWindow1, ID\_STATICTEXT4, \_("Значения СКО:"), wxPoint(306,294), wxDefaultSize, 0, \_T("ID\_STATICTEXT4"));  TimePickerCtrl1 = new wxTimePickerCtrl(SashWindow1, ID\_TIMEPICKERCTRL1, wxDateTime::Now(), wxPoint(305,93), wxSize(85,21), 0, wxDefaultValidator, \_T("ID\_TIMEPICKERCTRL1"));  Button3 = new wxButton(SashWindow1, ID\_BUTTON4, \_("для отладки/test"), wxPoint(303,463), wxDefaultSize, 0, wxDefaultValidator, \_T("ID\_BUTTON4"));  StaticText5 = new wxStaticText(SashWindow1, ID\_STATICTEXT5, \_("Время прогнозирования по UTC(+3)"), wxPoint(305,49), wxDefaultSize, 0, \_T("ID\_STATICTEXT5"));  SashWindow1->SetSashVisible(wxSASH\_TOP, true);  SashWindow1->SetSashVisible(wxSASH\_BOTTOM, true);  SashWindow1->SetSashVisible(wxSASH\_LEFT, true);  SashWindow1->SetSashVisible(wxSASH\_RIGHT, true);  Connect(ID\_DATEPICKERCTRL1,wxEVT\_DATE\_CHANGED,(wxObjectEventFunction)&dataDialog::OnDatePickerCtrl1Changed);  Connect(ID\_BUTTON2,wxEVT\_COMMAND\_BUTTON\_CLICKED,(wxObjectEventFunction)&dataDialog::OnButton2Click);  Connect(ID\_BUTTON1,wxEVT\_COMMAND\_BUTTON\_CLICKED,(wxObjectEventFunction)&dataDialog::OnButton1Click1);  Connect(ID\_TEXTCTRL1,wxEVT\_COMMAND\_TEXT\_UPDATED,(wxObjectEventFunction)&dataDialog::OnTextCtrl1Text1);  Connect(ID\_TIMEPICKERCTRL1,wxEVT\_DATE\_CHANGED,(wxObjectEventFunction)&dataDialog::OnTimePickerCtrl1Changed);  Connect(ID\_BUTTON4,wxEVT\_COMMAND\_BUTTON\_CLICKED,(wxObjectEventFunction)&dataDialog::OnButton3Click2);  Connect(ID\_SASHWINDOW1,wxEVT\_SASH\_DRAGGED,(wxObjectEventFunction)&dataDialog::OnSashWindow1SashDragged);  Connect(wxID\_ANY,wxEVT\_INIT\_DIALOG,(wxObjectEventFunction)&dataDialog::OnInit);  //\*)  Grid = new wxGrid(SashWindow1, ID\_GRID, wxPoint(33,60), wxSize(244,490), 0, \_T("ID\_GRID"));  Grid->CreateGrid(24,2);  Grid->SetColLabelValue(0, \_("SISRE, m"));  Grid->SetColLabelValue(1, \_("SISVE, mm/s"));  Grid->SetDefaultCellFont( Grid->GetFont() );  Grid->SetDefaultCellTextColour( Grid->GetForegroundColour() );  }  dataDialog::~dataDialog()  {  //(\*Destroy(dataDialog)  //\*)  }  void dataDialog::OnQuit(wxCommandEvent& event)  {  Close();  }  void dataDialog::OnInit(wxInitDialogEvent& event) {};  void dataDialog::OnAbout(wxCommandEvent& event)  {  wxString msg = wxbuildinfo(long\_f);  wxMessageBox(msg, \_("Welcome to..."));  }  void dataDialog::OnChoice1Select(wxCommandEvent& event)  {  }  void dataDialog::OnSashWindow1SashDragged(wxSashEvent& event)  {  }  void dataDialog::OnButton2Click(wxCommandEvent& event)  {  wxString s;  const char\* File1 ;  const char\* file ;  //wxMessageBox(Choice1->GetString(Choice1->GetSelection()), \_(""));  if ((Choice1->GetString(Choice1->GetSelection()))== "GPS"s)  {  File1 = "/MCC/PRODUCTS/LATEST/MERMS-GSC\_C.ete";  file = "MERMS-GSC\_C.ete";  wxTextFile file11(wxT("MERMS-GSC\_C.ete"));  if (file11.Exists())  {  wxRemoveFile(file);  }  }  if ((Choice1->GetString(Choice1->GetSelection()))== "Glonass"s)  {  File1 = "/MCC/PRODUCTS/LATEST/MERMS-RSC\_C.ete";  file = "MERMS-RSC\_C.ete";  wxTextFile file11(wxT("MERMS-RSC\_C.ete"));  if (file11.Exists())  {  wxRemoveFile(file);  }  }  if ((Choice1->GetString(Choice1->GetSelection()))== "Galileo"s)  {  File1 = "/MCC/PRODUCTS/LATEST/MERMS-ESC\_C.ete";  file = "MERMS-ESC\_C.ete";  wxTextFile file11(wxT("MERMS-ESC\_C.ete"));  if (file11.Exists())  {  wxRemoveFile(file);  }  }  if ((Choice1->GetString(Choice1->GetSelection()))== "Beidou"s)  {  File1 = "/MCC/PRODUCTS/LATEST/MERMS-CSC\_C.ete";  file = "MERMS-CSC\_C.ete";  wxTextFile file11(wxT("MERMS-CSC\_C.ete"));  if (file11.Exists())  {  wxRemoveFile(file);  }  }  if ((Choice1->GetString(Choice1->GetSelection()))== "QZSS"s)  {  File1 = "/MCC/PRODUCTS/LATEST/MERMS-JSC\_C.ete";  file = "MERMS-JSC\_C.ete";  wxTextFile file11(wxT("MERMS-JSC\_C.ete"));  if (file11.Exists())  {  wxRemoveFile(file);  }  }  bool down = download( "glonass-iac.ru", NULL, NULL, File1, file);  if (!down)  {  wxMessageBox(\_("Error"), \_("Error"));  return;  }  Gridd(file);  }  void dataDialog::Gridd(const char\* file)  {  int k=0;  int sizeY;  if (Grid != NULL)  {  delete Grid;  }  memset(&SISerr,0, sizeof(SISerr));  int max\_sats = parse(file);  sizeY=490;  Grid = new wxGrid(SashWindow1, ID\_GRID, wxPoint(33,60), wxSize(244,sizeY), 0, \_T("ID\_GRID"));  wxString s;  Grid->CreateGrid(max\_sats,2);  Grid->SetColLabelValue(0, \_("SISRE, m"));  Grid->SetColLabelValue(1, \_("SISVE, mm/s"));  Grid->SetDefaultCellFont( Grid->GetFont() );  Grid->SetDefaultCellTextColour( Grid->GetForegroundColour() );  for (k=0; k<max\_sats; k++ )  {  Grid->SetCellValue((k), 0, wxString::Format("%.3f", SISerr[k].SISRE));  Grid->SetCellValue((k), 1, wxString::Format("%.3f", SISerr[k].SISVE));  }  }  void dataDialog::OnChoice1Select3(wxCommandEvent& event)  {  }  void dataDialog::OnTextCtrl1Text1(wxCommandEvent& event)  {  }  void dataDialog::OnSpinCtrl1Change(wxSpinEvent& event)  {  }  void dataDialog::OnChoice1Select4(wxCommandEvent& event)  {  }  void dataDialog::OnButton1Click1(wxCommandEvent& event)  {  StaticText4 ->ClearBackground();  // Вводим значения h,B,L  double h;  double Bgrad;  double Lgrad;  TextCtrlH->GetValue().ToDouble(&h);  TextCtrlB->GetValue().ToDouble(&Bgrad);  TextCtrlL->GetValue().ToDouble(&Lgrad);  double PI = M\_PI;  double B; //Latitude  double L; //Longitude  B=Bgrad\*PI/180;  L=Lgrad\*PI/180;  double N;  double e=0;  double a=6378136; // радиус З  // Получение координат потребителя  N=a/sqrt(1-(e\*e)\*(sin(B))\*(sin(B)));  double Coord\_x;  double Coord\_y;  double Coord\_z;  Coord\_x = (N+h)\*cos(B)\*cos(L);  Coord\_y = (N+h)\*cos(B)\*sin(L);  Coord\_z = ((1-e\*e)\*N+h)\*sin(B);  double Coord\_user[3];  Coord\_user[0]=(N+h)\*cos(B)\*cos(L);  Coord\_user[1]= (N+h)\*cos(B)\*sin(L);  Coord\_user[2]= ((1-e\*e)\*N+h)\*sin(B);  double Coord\_sput[3];  double alpha;  //передаем в класс определения времени  //Т.е получили время на которое необходимо предсказать  //Далее "найдем" файл от которого будет высчитывать само предсказание  //если от сегодн. дня, то день -1; тк файл загружается ~ в 18 00;  // Считаем сегодняшн. дату  int year\_predsk;  int month\_predsk;  int day\_predsk;  int hour\_predsk;  int min\_predsk;  int sec\_predsk;  //время от которого скачиваем  int year\_down;  int month\_down;  int day\_down;  //int hour\_down;  //int min\_down;  //int sec\_down ;  wxDateTime T;  T = DatePickerCtrl1->GetValue();  day\_predsk = T.GetDay(); //для скачивания файла  month\_predsk = T.GetMonth()+1; // тк 1 месяц равен 0;  year\_predsk = T.GetYear();  //получаем дату и время  // Дата и время от которой предсказывать:  TimePickerCtrl1->GetTime(&hour\_predsk, &min\_predsk, &sec\_predsk);  time\_t nowsec = time(0);  tm \*ltm = localtime(&nowsec);  int yeartoday = 1900+ltm->tm\_year;  int monthtoday = 1 + ltm->tm\_mon;  int daytoday = ltm->tm\_mday;  int hourrtoday = ltm->tm\_hour;  int mintoday = 1 + ltm->tm\_min;  int sectoday = 1 + ltm->tm\_sec;  // Если предсказание в прошлом, то год ии месяц предсказания остется тот же, но день - прошлый  day\_down = day\_predsk-1;  year\_down = year\_predsk;  month\_down = month\_predsk;  ofstream f;  f.open("test/test2.txt");  f<< "year\_predsk="<< year\_predsk<<endl;  f<< "month\_predsk="<< month\_predsk<<endl;  f<< "day\_predsk="<< day\_predsk<<endl;  f<< "hour\_predsk="<< hour\_predsk<<endl;  f<< "min\_predsk="<< min\_predsk<<endl;  f<< "sec\_predsk="<< sec\_predsk<<endl;  //если предсказание уже на будущее, то год,месяц остается сегодняшний, а день минус 1  //если год = году сейчас, но день больше или равен, то день минус 1  // по сути можно облегчить и сделать 1 цикл через "или"  if (year\_predsk>yeartoday)  {  day\_down = daytoday -1;  year\_down = yeartoday;  month\_down = monthtoday;  }  if (year\_predsk == yeartoday)  {  if ( day\_predsk >= daytoday)  {  day\_down = daytoday -1;  year\_down = yeartoday;  month\_down = monthtoday;  }  }  f<< "day\_download="<< day\_down<<endl;  f<< "month\_download="<< month\_down<<endl;  f<< "year\_download="<< year\_down<<endl;  timeCalc calc(day\_predsk,month\_predsk,year\_predsk,hour\_predsk,min\_predsk,sec\_predsk,00);  //  //преобразование в слово для скачивания  string textYear = to\_string(year\_down);  string text5 = "MCCJ\_";  string text2 = to\_string(year\_down -2000);  string text3 ;  if (month\_down<10)  {  text3 = "0"s + to\_string(month\_down);  }  else  {  text3 = to\_string(month\_down);  }  string text4;  if (day\_down<10)  {  text4 = "0"s +to\_string(day\_down);  }  else  {  text4 = to\_string(day\_down);  }  string text1;  string text0;  mat sko;  if ((Choice1->GetString(Choice1->GetSelection()))== "GPS")  {  text1 = text5+text2+text3+text4+".agp"s;  text0 = "/MCC/ALMANAC/"+ textYear +"/"+text1;  f<< " day\_down="<< day\_down<<endl;  f<< " text4="<< text4<<endl;  const char\* File1 ;  const char\* file ;  File1 = text0.c\_str();//"/MCC/ALMANAC/2015/MCCJ\_150307.agp"//перевод строки с строку Си  file = text1.c\_str();  //! добавить если файла нет, искать ближайший!  f<< "const char\* File1"<< File1<<endl;  f<< " file"<< file<<endl;  bool down = download( "glonass-iac.ru", NULL, NULL, File1, file);  int max\_sats = parseGPS(file);  // Расчет матрицы Dn, Hn, SKO  int numberSput = 32;  int vsb[numberSput] ;  int sumvsb = 0;  vector<int> Visibles; //вектор из кол-во элементов - visibles  calc.timeGPS();  double toe=calc.sec\_since\_week;  f<< "GPS:"<<endl;  f<<"toe()calc.sec\_since\_week="<<toe<<endl;  f<<"week="<<calc.week<<endl;  Coordinates Coord\_sp; // можно потом заменить в 482 строке и ниже.  for (int i=1; i<=numberSput; i++)  {  // Получение коорд спутников  //ephemerids(double toe,int t\_almanax, double M0, double sqrtA, double E, double I, double Om0, double time\_week ))  Coord\_sp = ephemerids(toe,  almanax\_GPS[i-1].t\_almanax,  almanax\_GPS[i-1].M0,  almanax\_GPS[i-1].sqrtA,  almanax\_GPS[i-1].E,  almanax\_GPS[i-1].I,  almanax\_GPS[i-1].Om0,  almanax\_GPS[i-1].time\_week);  f <<"i-1 (номер спут)"<<i-1<<endl;  f<<"Coord\_sp.X="<<Coord\_sp.X<<endl;  f <<"Coord\_sp.Y =" <<Coord\_sp.Y<<endl;  f <<"Coord\_sp.Z =" <<Coord\_sp.Z<<endl;  Coord\_sput[0] = Coord\_sp.X;  Coord\_sput[1] = Coord\_sp.Y;  Coord\_sput[2] = Coord\_sp.Z;  // Определение угла  alpha = 90 - (angle(Coord\_sput, Coord\_user, B, L)\*180/PI);  // определение видимости спутника  vsb[i]=0;  if ((alpha) >5)  {  vsb[i]=1;  sumvsb++;  Visibles.push\_back(i); // добавление элемента в конец вектора  }  }  // получение матрицы Dn  int i = 0;  mat Dn;  Dn.zeros(sumvsb, sumvsb);  for (int k=1; k<=numberSput; k++)  {  if ((vsb[k]) == 1)  {  Dn(i,i) = SISerr[i].SISRE;  i++;  }  }  double max\_val\_Dn = Dn.max();  for (int i= 0; i<sumvsb; i++)  {  if ( Dn(i,i) == 0)  {  Dn(i,i) = max\_val\_Dn;  }  }  // получение матрицы H  double dx;  double dy;  double dz;  double Ri;  mat H(sumvsb, 4);  H.zeros();  int numsput = 0;  for (int k=1; k<=numberSput; k++)  {  if ((vsb[k]) == 1)  {  Coord\_sp = ephemerids(toe,  almanax\_GPS[k-1].t\_almanax,  almanax\_GPS[k-1].M0,  almanax\_GPS[k-1].sqrtA,  almanax\_GPS[k-1].E,  almanax\_GPS[k-1].I,  almanax\_GPS[k-1].Om0,  almanax\_GPS[k-1].time\_week);  dx=(Coord\_sp.X-Coord\_x);  dy=(Coord\_sp.Y-Coord\_y);  dz=(Coord\_sp.Z- Coord\_z);  // Ri = sqrt (SQUARE(dx)+SQUARE(dy)+SQUARE(dz));  Ri = sqrt (pow(dx,2)+pow(dy,2)+pow(dz,2));  H(numsput, 0 ) = dx/Ri;  H(numsput, 1) = dy/Ri;  H(numsput, 2) = dz/Ri;  H(numsput, 3) = 1;  numsput++ ;  }  }  mat Htr = H.t();  sko = sqrt((inv(Htr\*inv(Dn)\*H)).t());  }  else if ((Choice1->GetString(Choice1->GetSelection()))== "Glonass")  {  text1 = text5+text2+text3+text4+".agl"s;  text0 = "/MCC/ALMANAC/"+ textYear +"/"+text1;  f<< " day\_down="<< day\_down<<endl;  f<< " text4="<< text4<<endl;  const char\* File1 ;  const char\* file ;  File1 = text0.c\_str();//"/MCC/ALMANAC/2015/MCCJ\_150307.agl"//перевод строки с строку Си  file = text1.c\_str();  //! добавить если файла нет, искать ближайший!  f<< "const char\* File1"<< File1<<endl;  f<< " file"<< file<<endl;  bool down = download( "glonass-iac.ru", NULL, NULL, File1, file);  int max\_sats = parseGLNS(file);  int numberSput = 24;  int vsb[numberSput] ;  int sumvsb = 0;  vector<int> Visibles; //вектор из кол-во элементов - visibles  //double toe=44271.777;//время  f<< "Glns:"<<endl;  f<<"calc.GLNS\_numb\_fouryear\_period (N4)=" << calc.GLNS\_numb\_fouryear\_period<<endl;;  f <<"calc.GLNS\_sec\_since\_week=" <<calc.GLNS\_sec\_since\_week<<endl;;  GlonassCoordinates Coord\_sp;  for (int i=1; i<=numberSput; i++)  {  // Получение коорд спутников  //ephemerids(double toe,int t\_almanax, double M0, double sqrtA, double E, double I, double Om0, double time\_week ))  calc.timeGLNS();  timeCalc GLNSephemTime( almanax\_GLNS[i-1].date,almanax\_GLNS[i-1].month, almanax\_GLNS[i-1].year,0,0,0,0);  Coord\_sp = ephemeridsGLNS(calc.GLNS\_numb\_fouryear\_period, //N4  calc.GLNS\_day\_after\_vis\_year,  calc.GLNS\_sec\_since\_week,  GLNSephemTime.GLNS\_numb\_fouryear\_period, //Na берем из расчета даты альманаха  almanax\_GLNS[i-1].tLA,  almanax\_GLNS[i-1].dT,  almanax\_GLNS[i-1].dT,  almanax\_GLNS[i-1].dTT,  almanax\_GLNS[i-1].E,  almanax\_GLNS[i-1].w,  almanax\_GLNS[i-1].Lam);  f <<"i-1 (номер спутн) ="<<i-1<<endl;  f << "GLNS\_numb\_fouryear\_period (Na)"<< GLNSephemTime.GLNS\_numb\_fouryear\_period<<endl;  f<<"Coord\_sp.X="<<Coord\_sp.X<<endl;  f <<"Coord\_sp.Y =" <<Coord\_sp.Y<<endl;  f <<"Coord\_sp.Z =" <<Coord\_sp.Z<<endl;  Coord\_sput[0] = Coord\_sp.X;  Coord\_sput[1] = Coord\_sp.Y;  Coord\_sput[2] = Coord\_sp.Z;  // Определение угла  alpha = 90 - (angle(Coord\_sput, Coord\_user, B, L)\*180/PI);  // определение видимости спутника  vsb[i]=0;  if ((alpha) >5)  {  vsb[i]=1;  sumvsb++;  Visibles.push\_back(i); // добавление элемента в конец вектора  }  }  // получение матрицы Dn  int i = 0;  mat Dn;  Dn.zeros(sumvsb, sumvsb);  for (int k=1; k<=numberSput; k++)  {  if ((vsb[k]) == 1)  {  Dn(i,i) = SISerr[i].SISRE;  i++;  }  }  double max\_val\_Dn = Dn.max();  for (int i= 0; i<sumvsb; i++)  {  if ( Dn(i,i) == 0)  {  Dn(i,i) = max\_val\_Dn;  }  }  // получение матрицы H  double dx;  double dy;  double dz;  double Ri;  mat H(sumvsb, 4);  H.zeros();  int numsput = 0;  for (int k=1; k<=numberSput; k++)  {  if ((vsb[k]) == 1)  {  timeCalc GLNSephemTime( almanax\_GLNS[k-1].date,almanax\_GLNS[k-1].month, almanax\_GLNS[k-1].year,0,0,0,0);  Coord\_sp = ephemeridsGLNS(calc.GLNS\_numb\_fouryear\_period, //N4  calc.GLNS\_day\_after\_vis\_year,  calc.GLNS\_sec\_since\_week,  GLNSephemTime.GLNS\_numb\_fouryear\_period, //Na берем из расчета даты альманаха  almanax\_GLNS[k-1].tLA,  almanax\_GLNS[k-1].dT,  almanax\_GLNS[k-1].dT,  almanax\_GLNS[k-1].dTT,  almanax\_GLNS[k-1].E,  almanax\_GLNS[k-1].w,  almanax\_GLNS[k-1].Lam);  dx=(Coord\_sp.X-Coord\_x);  dy=(Coord\_sp.Y-Coord\_y);  dz=(Coord\_sp.Z- Coord\_z);  // Ri = sqrt (SQUARE(dx)+SQUARE(dy)+SQUARE(dz));  Ri = sqrt (pow(dx,2)+pow(dy,2)+pow(dz,2));  H(numsput, 0 ) = dx/Ri;  H(numsput, 1) = dy/Ri;  H(numsput, 2) = dz/Ri;  H(numsput, 3) = 1;  numsput++ ;  }  }  //для ион  /\* text1 = "BRDC1510.21n"s;  text\_0 = "/MCC/BRDC/" +textYear +"/" + text\_1";  const char\* File11 ;  const char\* file1 ;  File11 = text0.c\_str();//""//перевод строки с строку Си  file1 = text1.c\_str();  bool down = download( "glonass-iac.ru", NULL, NULL, File11, file1);  \*/  mat Htr = H.t();  sko = sqrt((inv(Htr\*inv(Dn)\*H)).t());  }  else  {  wxMessageBox(\_("Выберите другую ГНСС"), \_("Error"));  }  wxString s;  s.Printf("Значение СКО:\nСКО для x: %.3f м\nСКО для y: %.3f м\nСКО для z: %.3f м\nСКО для D: %.3f, м СКО: %.3f м",  sko(0,0), sko(1,1), sko(2,2), sko(3,3), sqrt (pow(sko(0,0),2)+pow(sko(1,1),2)+pow(sko(2,2),2) ));  StaticText4->SetLabel(s);  f.close();  }  void dataDialog::OnDatePickerCtrl1Changed(wxDateEvent& event)  {  }  void dataDialog::OnTimePickerCtrl1Changed(wxDateEvent& event)  {  }  void dataDialog::OnButton3Click1(wxCommandEvent& event)  {  /\*int \*hour;  int \*minn;  int \*sec;  TimePickerCtrl1->GetTime(hour, minn, sec);  wxString s;  s.Printf("Значение max\_sats: %f\nЗначение PRN: %f\nЗначение t\_almanax: %f\nЗначение v0m0:", hour, minn,sec);  StaticText5->SetLabel(s);  \*/  } |

Приложение 2

|  |
| --- |
| /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Includes  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Подключение необходимого минимума заголовочных файлов  // Первым должен подключаться интерфейсный файл модуля  #include <stdio.h>  #include "parser.h"  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Macro Definitions  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Локальные макроопределения  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Extern Data  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Объявления экземпляров экспортируемых данных  data\_t SISerr[75];  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Local Data  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Локальные объявления типов и данных  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Local Function Prototypes  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Прототипы локальных функций (без комментариев)  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Function Pointers  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Объявление указателей на функции (без комментариев)  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Function Definitions  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Реализация функций  // Сначала реализация интерфейсных функций, далее реализация локальных функций  // Все функции должны иметь описание  int parse(const char\* file)  {  int i;  FILE\* fd;  char systype;  char dummy[30];  fd = fopen(file, "r");  while ( !feof(fd) )  {  fscanf(fd, "%c", &systype);  fscanf(fd, "%c", &systype);  /\* if (systype !=('G'|| 'R'))  {  printf("Sync error\n");  break;  }\*/  fscanf(fd, "%d", &i); //01 123 - орбитальный слот или PRN (R01-R24 для ГЛОНАСС, G01-G32 для GPS), либо  SISerr[i-1].systype = systype;  fscanf(fd, "%d", (int \*)dummy); //02 1 - учет признака пригодности (0-только пригодные КА по данным навигационных сообщений,  // 1 - возможно использование непригодных по данным эфемерид,  // 2 - возможно использование непригодных по данным альманахов,  // 3 - возможно использование любых непригодных эфемерид)  fscanf(fd, "%u", (unsigned int \*)dummy); //03 12 - число  fscanf(fd, "%c", dummy);  fscanf(fd, "%u", (unsigned int \*)dummy); //04 12 - месяц  fscanf(fd, "%c", dummy);  fscanf(fd, "%u", (unsigned int \*)dummy);//05 12 - год  fscanf(fd, "%u", (unsigned int \*)dummy);//06 12 - часы UTC  fscanf(fd, "%c", dummy);  fscanf(fd, "%u", (unsigned int \*)dummy); //07 12 - минуты UTC  fscanf(fd, "%c", dummy);  fscanf(fd, "%u", (unsigned int \*)dummy); //08 12 - секунды UTC  fscanf(fd, "%lf", (double \*)dummy); //09 123.123 - длительность интервала оценки, сутки  fscanf(fd, "%u", (unsigned int \*)dummy); //10 123456 - общее число обработанных точек  fscanf(fd, "%u", (unsigned int \*)dummy); //11 123 - % вошедших в оценку точек R, N, B  fscanf(fd, "%u", (unsigned int \*)dummy); //12 123 - % вошедших в оценку точек часы  fscanf(fd, "%u", (unsigned int \*)dummy); //13 123 - % вошедших в оценку точек SISRE  fscanf(fd, "%u", (unsigned int \*)dummy); //14 123 - % вошедших в оценку точек Vr, Vn, Vb  fscanf(fd, "%u", (unsigned int \*)dummy); //15 123 - % вошедших в оценку точек частота  fscanf(fd, "%u", (unsigned int \*)dummy); //16 123 - % вошедших в оценку точек SISVE  fscanf(fd, "%lf", (double \*)dummy);// 17 123.123 - МО ошибки R, м МIN ошибки R, м Med модуля ошибки R, м  fscanf(fd, "%lf", (double \*)dummy);// 18 123.123 - МО ошибки N, м МIN ошибки N, м Med модуля ошибки N, м  fscanf(fd, "%lf", (double \*)dummy);//19 123.123 - МО ошибки B, м МIN ошибки B, м Med модуля ошибки B, м  fscanf(fd, "%lf", (double \*)dummy);//20 123.123 - МО ошибки часов, нс МIN ошибки часов, нс Med модуля ошибки часов, нс  fscanf(fd, "%lf", (double \*)dummy);//21 123.123 - МО ошибки SISRE, м МIN ошибки SISRE, м Med модуля ошибки SISRE, м  fscanf(fd, "%lf", (double \*)dummy);//22 123.123 - СКП R, м МAX ошибки R, м 95% модуля ошибки R, м  fscanf(fd, "%lf", (double \*)dummy);//23 123.123 - СКП N, м МAX ошибки N, м 95% модуля ошибки N, м  fscanf(fd, "%lf", (double \*)dummy);//24 123.123 - СКП B, м МAX ошибки B, м 95% модуля ошибки B, м  fscanf(fd, "%lf", (double \*)dummy);//25 123.123 - СКП часов, нс МAX ошибки часов, нс 95% модуля ошибки часов, нс  fscanf(fd, "%lf", &(SISerr[i-1].SISRE));//26 123.123 - СКП SISRE, м МAX ошибки SISRE, м 95% модуля ошибки SISRE, м  fscanf(fd, "%lf", (double \*)dummy);//27 123.123 - МО ошибки VR, мм/с МIN ошибки VR, мм/с Med ошибки VR, мм/с  fscanf(fd, "%lf", (double \*)dummy);//28 123.123 - МО ошибки VN, мм/с МIN ошибки VN, мм/с Med ошибки VN, мм/с  fscanf(fd, "%lf", (double \*)dummy);//29 123.123 - МО ошибки VB, мм/с МIN ошибки VB, мм/с Med ошибки VB, мм/с  fscanf(fd, "%lf", (double \*)dummy);//30 123.123 - МО ошибки частоты, нс/с МIN ошибки частоты, нс Med ошибки частоты, нс  fscanf(fd, "%lf", (double \*)dummy);//31 123.123 - МО ошибки SISVE, м МIN ошибки SISVE, м Med ошибки SISVE, мм/с  fscanf(fd, "%lf", (double \*)dummy);//32 123.123 - СКП VR, мм/с МAX ошибки VR, мм/с 95% ошибки VR, мм/с  fscanf(fd, "%lf", (double \*)dummy);//33 123.123 - СКП VN, мм/с МAX ошибки VN, мм/с 95% ошибки VN, мм/с  fscanf(fd, "%lf", (double \*)dummy);//34 123.123 - СКП VB, мм/с МAX ошибки VB, мм/с 95% ошибки VB, мм/с  fscanf(fd, "%lf", (double \*)dummy);//35 123.123 - СКП частоты, нс/с МAX ошибки часов, нс/c 95% ошибки часов, нс/c  fscanf(fd, "%lf", &(SISerr[i-1].SISVE));//36 123.123 - СКП SISVE, мм/с МAX ошибки SISVE, мм/с 95% ошибки SISVE, мм/с  fscanf(fd, "%c", &systype);  }  int imax=i;  fclose(fd);  return imax;  } |

Приложение 3

|  |
| --- |
| #ifndef PARSER\_H  #define PARSER\_H  #ifdef \_\_cplusplus  extern "C" {  #endif  typedef struct  {  char systype;  double SISRE;  double SISVE;  } data\_t;  extern data\_t SISerr[75];  int parse(const char\* file);  #ifdef \_\_cplusplus  }  #endif  #endif |

Приложение 4

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" standalone="yes" ?>  <CodeBlocks\_project\_file>  <FileVersion major="1" minor="6" />  <Project>  <Option title="Data" />  <Option pch\_mode="2" />  <Option compiler="gcc" />  <Build>  <Target title="Debug">  <Option platforms="Windows;" />  <Option output="bin/Debug/Data" prefix\_auto="1" extension\_auto="1" />  <Option object\_output="obj/Debug/" />  <Option type="0" />  <Option compiler="gcc" />  <Option projectLinkerOptionsRelation="2" />  <Compiler>  <Add option="-g" />  <Add option="-D\_\_WXDEBUG\_\_" />  </Compiler>  <Linker>  <Add option="-static-libstdc++" />  <Add option="-static-libgcc" />  <Add option="-static" />  <Add library="libwxmsw30u.a" />  <Add library="libwxpng.a" />  <Add library="libwxjpeg.a" />  <Add library="libwxtiff.a" />  <Add library="libwxzlib.a" />  <Add library="libwininet.a" />  </Linker>  </Target>  </Build>  <Compiler>  <Add option="-Wall" />  <Add option="-pipe" />  <Add option="-mthreads" />  <Add option="-D\_\_GNUWIN32\_\_" />  <Add option="-D\_\_WXMSW\_\_" />  <Add option="-DwxUSE\_UNICODE" />  <Add directory="$(#wx)/include" />  <Add directory="$(#wx)/lib/gcc\_lib/mswud" />  <Add directory="." />  </Compiler>  <ResourceCompiler>  <Add directory="$(#wx)/include" />  <Add directory="$(#wx)/lib/gcc\_lib/mswud" />  </ResourceCompiler>  <Linker>  <Add option="-static-libstdc++" />  <Add option="-static-libgcc" />  <Add option="-static" />  <Add option="-mthreads" />  <Add library="libwxmsw30u.a" />  <Add library="libwxpng.a" />  <Add library="libwxjpeg.a" />  <Add library="libwxtiff.a" />  <Add library="libwxzlib.a" />  <Add library="libkernel32.a" />  <Add library="libuser32.a" />  <Add library="libgdi32.a" />  <Add library="libwinspool.a" />  <Add library="libcomdlg32.a" />  <Add library="libadvapi32.a" />  <Add library="libshell32.a" />  <Add library="libole32.a" />  <Add library="liboleaut32.a" />  <Add library="libuuid.a" />  <Add library="libcomctl32.a" />  <Add library="libwsock32.a" />  <Add library="libodbc32.a" />  <Add library="libuxtheme.a" />  <Add library="libopenblas.a" />  <Add library="libshlwapi.a" />  <Add library="libversion.a" />  <Add library="liboleacc.a" />  <Add directory="$(#wx)/lib/gcc\_lib" />  <Add directory="." />  </Linker>  <Unit filename="FTPdownl.cpp" />  <Unit filename="FTPdownl.h" />  <Unit filename="angle.cpp" />  <Unit filename="angle.h" />  <Unit filename="data.cbp" />  <Unit filename="dataApp.cpp" />  <Unit filename="dataApp.h" />  <Unit filename="dataMain.cpp" />  <Unit filename="dataMain.h" />  <Unit filename="ephemerids.cpp" />  <Unit filename="ephemerids.h" />  <Unit filename="ephemeridsGLNS.cpp" />  <Unit filename="ephemeridsGLNS.h" />  <Unit filename="parser.c">  <Option compilerVar="CC" />  </Unit>  <Unit filename="parser.h" />  <Unit filename="parserGLNS.H" />  <Unit filename="parserGLNS.c">  <Option compilerVar="CC" />  </Unit>  <Unit filename="parserGPS.H" />  <Unit filename="parserGPS.c">  <Option compilerVar="CC" />  </Unit>  <Unit filename="resource.rc">  <Option compilerVar="WINDRES" />  </Unit>  <Unit filename="timeCalc.cpp" />  <Unit filename="timeCalc.h" />  <Unit filename="wxsmith/datadialog.wxs" />  <Unit filename="xyz2enu.cpp" />  <Unit filename="xyz2enu.h" />  <Extensions>  <code\_completion />  <envvars />  <debugger />  <lib\_finder disable\_auto="1" />  <wxsmith version="1">  <gui name="wxWidgets" src="dataApp.cpp" main="dataDialog" init\_handlers="necessary" language="CPP" />  <resources>  <wxDialog wxs="wxsmith/datadialog.wxs" src="dataMain.cpp" hdr="dataMain.h" fwddecl="0" i18n="1" name="dataDialog" language="CPP" />  </resources>  </wxsmith>  </Extensions>  </Project>  </CodeBlocks\_project\_file> |

Приложение 5

|  |
| --- |
| #ifndef dataMAIN\_H  #define dataMAIN\_H  //(\*Headers(dataDialog)  #include <wx/button.h>  #include <wx/choice.h>  #include <wx/datectrl.h>  #include <wx/dateevt.h>  #include <wx/dialog.h>  #include <wx/notebook.h>  #include <wx/sashwin.h>  #include <wx/stattext.h>  #include <wx/textctrl.h>  #include <wx/timectrl.h>  //\*)  #include <wx/grid.h>  #include <wx/spinctrl.h>  //#include <wx/gdicmn.h>  class dataDialog: public wxDialog  {  public:  dataDialog(wxWindow\* parent,wxWindowID id = -1);  virtual ~dataDialog();  private:  //(\*Handlers(dataDialog)  void OnQuit(wxCommandEvent& event);  void OnAbout(wxCommandEvent& event);  void OnChoice1Select(wxCommandEvent& event);  void OnSashWindow1SashDragged(wxSashEvent& event);  void OnChoice1Select12(wxCommandEvent& event);  void OnChoice1Select2(wxCommandEvent& event);  void OnButton1Click(wxCommandEvent& event);  void OnButton2Click(wxCommandEvent& event);  void OnComboBox1Selected(wxCommandEvent& event);  void OnButton3Click(wxCommandEvent& event);  void OnChoice1Select1(wxCommandEvent& event);  void OnTextCtrl1Text(wxCommandEvent& event);  void OnInit(wxInitDialogEvent& event);  void OnGrid1CellLeftClick(wxGridEvent& event);  void OnChoice1Select3(wxCommandEvent& event);  void OnTextCtrl1Text1(wxCommandEvent& event);  void OnButton1Click1(wxCommandEvent& event);  void OnRichTextCtrl1Text(wxCommandEvent& event);  void OnSpinCtrl1Change(wxSpinEvent& event);  void OnCustom1Paint(wxPaintEvent& event);  void OnTextCtrlLText(wxCommandEvent& event);  void OnTextCtrlTEST1Text(wxCommandEvent& event);  void OnChoice1Select4(wxCommandEvent& event);  void OnTextCtrlZText(wxCommandEvent& event);  void OnButton1Click2(wxCommandEvent& event);  void OnDatePickerCtrl1Changed(wxDateEvent& event);  void OnButton3Click1(wxCommandEvent& event);  void OnButton3Click2(wxCommandEvent& event);  void OnTimePickerCtrl1Changed(wxDateEvent& event);  //\*)  void Gridd(const char\* file);  //(\*Identifiers(dataDialog)  static const long ID\_DATEPICKERCTRL1;  static const long ID\_CHOICE1;  static const long ID\_BUTTON2;  static const long ID\_NOTEBOOK1;  static const long ID\_BUTTON1;  static const long ID\_TEXTCTRL1;  static const long ID\_TEXTCTRL2;  static const long ID\_TEXTCTRL3;  static const long ID\_STATICTEXT1;  static const long ID\_STATICTEXT2;  static const long ID\_STATICTEXT3;  static const long ID\_STATICTEXT4;  static const long ID\_TIMEPICKERCTRL1;  static const long ID\_BUTTON4;  static const long ID\_STATICTEXT5;  static const long ID\_SASHWINDOW1;  //\*)  // static const long ID\_STATICTEXT4;  static const long ID\_GRID;  //(\*Declarations(dataDialog)  wxButton\* Button2;  wxButton\* Button3;  wxButton\* Down;  wxChoice\* Choice1;  wxDatePickerCtrl\* DatePickerCtrl1;  wxNotebook\* Notebook1;  wxSashWindow\* SashWindow1;  wxStaticText\* StaticText1;  wxStaticText\* StaticText2;  wxStaticText\* StaticText3;  wxStaticText\* StaticText4;  wxStaticText\* StaticText5;  wxTextCtrl\* TextCtrlB;  wxTextCtrl\* TextCtrlH;  wxTextCtrl\* TextCtrlL;  wxTimePickerCtrl\* TimePickerCtrl1;  //\*)  // wxStaticText\* StaticText4;  wxGrid\* Grid;  DECLARE\_EVENT\_TABLE()  };  #endif // dataMAIN\_H |

Приложение 6

|  |
| --- |
| #include <math.h>  void xyz2enu(const double lat, const double lon, double \* xyz2enu)  {  /\*%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  %\* Copyright c 2001 The board of trustees of the Leland Stanford \*  %\* Junior University. All rights reserved. \*  %\* This script file may be distributed and used freely, provided \*  %\* this copyright notice is always kept with it. \*  %\* \*  %\* Questions and comments should be directed to Todd Walter at: \*  %\* twalter@stanford.edu \*  %\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*33\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  %  %FINDXYZ2ENU find the rotation matrix to go from XYZ ECEF coordinates  % to a local East North Up frame  % [xyz2enu] = FINDXYZ2ENU(LAT, LON)  % LAT, LON specify the coordinates of the center of the local frame in radians  % XYZ2ENU is the rotation matrix such that DELTA\_ENU = XYZ2ENU\*DELTA\_XYZ \*/  xyz2enu[0] = sin(lon);  xyz2enu[1] = cos(lon);  xyz2enu[2] = 0.0;  xyz2enu[5] = cos(lat);  xyz2enu[8] = sin(lat);  xyz2enu[3] = -xyz2enu[1] \* xyz2enu[8];  xyz2enu[4] = -xyz2enu[0] \* xyz2enu[8];  xyz2enu[6] = xyz2enu[1] \* xyz2enu[5];  xyz2enu[7] = xyz2enu[0] \* xyz2enu[5];  xyz2enu[0] = -xyz2enu[0];  } |

Приложение 7

|  |
| --- |
| #ifndef xyz2enu\_H  #define xyz2enu\_H  void xyz2enu(const double lat, const double lon, double \* xyz2enu);  #endif // dataAPP\_H |

Приложение 8

|  |
| --- |
| #include <windows.h>  #include <wininet.h>  #include <iostream>  #include <string>  #include <stdio.h>  #include <wx/msgdlg.h>  #include <wx/string.h>  #include <wx/textfile.h>  #include <wx/dialog.h>  #include <wx/msgdlg.h>  #include <wx/spinctrl.h>  #include <wx/intl.h>  #include <wx/settings.h>  GlonassCoordinates ephemeridsGLNS(double N4 ,  double N,  double ti,  double N\_A,  double tlymbda\_A,  double dT,  double dI,  double dTT,  double Ee,  double omegaA,  double Lam)  {//toe - время на которое нуэно расчитать;  GlonassCoordinates Coordinates;  //1. Определяется интервал прогноза пр Δtпр в секундах:  double dN\_A;  double Tsr = 43200; //номинальное значение периода обращения НКА, в секундах (Tср определено в nинтерфейсе соответствующего сигнала)  double isr = 63; // Из указаний  if (N4 == 27)  dN\_A= N-N\_A-(floor((N-N\_A)/1461)\*1460); //() вычисление целого, ближайшего к x  else  dN\_A= N-N\_A-(floor((N-N\_A)/1461)\*1461);  double dtpr = dN\_A\*86400+(ti-tlymbda\_A);  //2.Рассчитывается количество целых витков W на интервале прогноза:  double W = floor(dtpr/(Tsr+dT)); // dTa = dT ? поправка к среднему значению драконического периода обращения.  //3 Определяется текущее наклонение:  double i = ((isr/180.0)+dI)\*M\_PI; //ΔiA = dI ? – поправка к среднему значению наклонения орбиты.  printf("i = %0.3f c\n", i );  //4 Определяются средний драконический период на витке W+1 и среднее движение:  double Tdr = Tsr+dT+(2\*W+1)\*dTT; //dT'a = dTT; половинная скорость изменения драконического периода.  printf("Tdr = %0.3f c\n", Tdr);  double n = 2\*M\_PI/Tdr;  printf("n = %0.9f c\n", n);  //5.Методом последовательных приближений m = 0, 1, 2… рассчитывается большая полуось орбиты a:  double GM = 398600441.8e6; //геоцентрическая константа гравитационного поля Земли с учетом атмосферы,  double ae = 6378136; // большая (экваториальная) полуось общеземного эллипсоида ПЗ-90  double J02 = 1082.62575e-6; //зональный гармонический коэффициент второй степени  double Tosk;  double p;  double a;  double epsA = Ee;  //стартовые значения  double a\_old = 0;  a = 1;  Tosk = Tdr;  p = 0;  //int ksh = 1;  /\*  while (abs(a-a\_old)>1e-2)  {a\_old = a;  a = pow((pow((Tosk/(2\*M\_PI)),2)\*GM),1.0/3); //Tock = Tdr;  p = a\*(1-pow(epsA,2));  Tosk = Tdr/((1-(3.0/2)\*J02\*pow((ae/p),2))\*((2-(5.0/2)\*pow(sin(i),2))\*((pow(1-  pow(epsA,2),3.0/2))/(1+epsA\*pow(cos(omegaA\*M\_PI),2)))+  (1+epsA\*pow(cos(omegaA\*M\_PI),3))/(1-pow(epsA,2))));  cout << "----"<<ksh<<endl;  printf("a = %0.3f c\n", a);  printf("p = %0.3f c\n", p);  printf("Tosk = %0.3f c\n", Tosk );  ksh++;  }  \*/  double eq1 = 2 - (5.0/2)\*pow(sin(i),2);  double eq2 = 1 - pow(epsA,2);  double eq3 = 1 + epsA\*cos(omegaA\*M\_PI);  double eq4 = pow(eq2,3.0/2);  double eq5 = pow(eq3,2);  double eq6 = eq4 / eq5;  double eq7 = pow(eq3,3);  double eq8 = eq7 / eq2;  double Big\_div = eq1 \* eq6 + eq8;  double eq9;  while (fabs(a - a\_old) > 1e-5)  {  a\_old = a;  a = pow((pow((Tosk/(2\*M\_PI)),2)\*GM),1.0/3); //err  p = a\*eq2;  eq9 = 1 - (3.0/2)\*J02\*(pow((ae/p),2));  Tosk = Tdr/(eq9\*Big\_div);  // ksh++;  }  //6Определяются текущие значения долготы восходящего узла орбиты и аргумента  //перигея с учетом их векового движения под влиянием сжатия Земли:  double omegaZ = 7.2921150e-5; //угловая скорость вращения Земли  double lymbda = Lam\*M\_PI-(omegaZ+(3.0/2)\*J02\*n\*pow((ae/p),2)\*cos(i))\*dtpr;  double omega = omegaA\*M\_PI-(3.0/4)\*J02\*n\*pow((ae/p),2)\*(1-5.0\*pow(cos(i),2))\*dtpr;  //7Рассчитывается значение средней долготы на момент прохождения текущего  //восходящего узла:  double E0 = -2\*atan((sqrt((1-epsA)/(1+epsA)))\*tan(omega/2.0));  double L1 =omega + E0 - epsA\*sin(E0);  //8Определяется текущее значение средней долготы НКА:  double L = L1+n\*(dtpr-(Tsr+dT)\*W-dTT\*pow(W,2));  //10Определяется эксцентрическая аномалия путем решения уравнения Кеплера  double E = L - omega;  double Eold = 0;  while (abs(E-Eold)>1e-9)  {  Eold = E;  ////////////////////////////////////////  E = L-omega+ epsA\*sin(E); // eps ?! !! стра 81, п10 икд  ////////////////////////////////////////  }  //11Вычисляются истинная аномалия  и аргумент широты НКА u:  double v = 2\*atan((sqrt((1-epsA)/(1+epsA)))\*tan(E/2));  double u = v + omega;  //12Рассчитываются координаты центра масс НКА в геоцентрической  //прямоугольной пространственной системе координат:  double r = p/(1+epsA\*cos(v));  Coordinates.X = r\*(cos(lymbda)\*cos(u)-sin(lymbda)\*sin(u)\*cos(i));  Coordinates.Y = r\*(sin(lymbda)\*cos(u)+cos(lymbda)\*sin(u)\*cos(i));  Coordinates.Z = r\*sin(u)\*sin(i);  return Coordinates;  } |

Приложение 9

|  |
| --- |
| #ifndef EPHEMERIDSGLNS\_H  #define EPHEMERIDSGLNS\_H  typedef struct  {  double X;  double Y;  double Z;  } GlonassCoordinates;  GlonassCoordinates ephemeridsGLNS(double N4 ,//номер текущего 4х летия  double N,//N – календарный номер суток внутри четырехлетнего периода, начиная с високосного года  //на которых находится заданный момент времени ti в секундах по шкале МДВ  double ti, //количество секунд от начала текущих суток. (берется из шкалы времени)  double N\_A, //календарный номер суток по шкале МДВ внутри четырехлетнего интервала, передаваемый НКА в составе неоперативной информации;  //будем рассчитывать в ручную исходя из строки 2 4-5-6 пункт.  double tlymbda\_A,//2 строка-7 - время прохождения первого узла, на которое все дано, с  double dT, //поправка к драконическому периоду, с  double dI,  double dTT,  double Ee,// эксцентриситет  double omegaA,//аргумент перигей )  double Lam); // долгота узла, полуциклы, она же лямбда A  #endif |

Приложение 10

|  |
| --- |
| #include "angle.h"  #include "xyz2enu.h"  #define \_USE\_MATH\_DEFINES  #include <math.h>  #define SQUARE(val) val \* val  double angle(double Coord\_sput[3],double Coord[3], double B, double L)  {  double ES[3];  double Renu[3];  double R[3];  double lengthES;  double C[9];  double a;  ES[0]=(Coord\_sput[0]-Coord[0]);  ES[1]=(Coord\_sput[1]-Coord[1]);  ES[2]=(Coord\_sput[2]- Coord[2]);  lengthES = sqrt (pow((Coord\_sput[0]- Coord[0]),2)+pow((Coord\_sput[1]-Coord[1]),2)+pow((Coord\_sput[2]- Coord[2]),2));  //lengthES = sqrt (SQUARE(Coord\_sput[0]- Coord[0])+SQUARE(Coord\_sput[1]-Coord[1])+SQUARE(Coord\_sput[2]- Coord[2]));  R[0]= ES[0]/lengthES; // вектор столбец  R[1]= ES[1]/lengthES;  R[2]= ES[2]/lengthES;  xyz2enu(B, L, C); // получение матрицы ENU в -> C  Renu[0]=C[0]\*R[0]+C[1]\*R[1]+C[2]\*R[2]; //вектор столбец  Renu[1]=C[3]\*R[0]+C[4]\*R[1]+C[5]\*R[2];  Renu[2]=C[6]\*R[0]+C[7]\*R[1]+C[8]\*R[2];  a = acos (Renu[2]); // rad  return a;  } |

Приложение 11

|  |
| --- |
| // Начнем с директив препроцессора. ADD\_H – это произвольное уникальное имя (обычно используется имя заголовочного файла)  #ifndef ANGLE\_H  #define ANGLE\_H  double angle(double Coord\_sput[3],double Coord[3], double B, double L);  #endif |

Приложение 12

|  |
| --- |
| <?xml version="1.0" encoding="utf-8" ?>  <wxsmith>  <object class="wxDialog" name="dataDialog">  <title>Data app</title>  <size>533,556</size>  <bg>wxSYS\_COLOUR\_INACTIVEBORDER</bg>  <minsize>-1,-1</minsize>  <maxsize>-1,-1</maxsize>  <id\_arg>0</id\_arg>  <handler function="OnInit" entry="EVT\_INIT\_DIALOG" />  <object class="wxSashWindow" name="ID\_SASHWINDOW1" variable="SashWindow1" member="yes">  <pos>56,40</pos>  <size>480,504</size>  <fg>wxSYS\_COLOUR\_WINDOWTEXT</fg>  <bg>wxSYS\_COLOUR\_MENUBAR</bg>  <handler function="OnSashWindow1SashDragged" entry="EVT\_SASH\_DRAGGED" />  <object class="wxDatePickerCtrl" name="ID\_DATEPICKERCTRL1" variable="DatePickerCtrl1" member="yes">  <pos>305,65</pos>  <size>85,21</size>  <fg>wxSYS\_COLOUR\_ACTIVEBORDER</fg>  <bg>wxSYS\_COLOUR\_HIGHLIGHT</bg>  <handler function="OnDatePickerCtrl1Changed" entry="EVT\_DATE\_CHANGED" />  </object>  <object class="wxChoice" name="ID\_CHOICE1" variable="Choice1" member="yes">  <content>  <item>Glonass</item>  <item>GPS</item>  <item>Galileo</item>  <item>Beidou</item>  <item>QZSS</item>  </content>  <selection>0</selection>  <pos>33,16</pos>  <size>244,21</size>  </object>  <object class="wxButton" name="ID\_BUTTON2" variable="Button2" member="yes">  <label>Загрузить</label>  <pos>305,16</pos>  <size>127,23</size>  <handler function="OnButton2Click" entry="EVT\_BUTTON" />  </object>  <object class="wxNotebook" name="ID\_NOTEBOOK1" variable="Notebook1" member="yes">  <pos>124,214</pos>  </object>  <object class="wxButton" name="ID\_BUTTON1" variable="Down" member="yes">  <label>Обработка </label>  <pos>305,266</pos>  <size>127,23</size>  <handler function="OnButton1Click1" entry="EVT\_BUTTON" />  </object>  <object class="wxTextCtrl" name="ID\_TEXTCTRL1" variable="TextCtrlH" member="yes">  <value>0</value>  <pos>305,145</pos>  <size>127,-1</size>  <handler function="OnTextCtrl1Text1" entry="EVT\_TEXT" />  </object>  <object class="wxTextCtrl" name="ID\_TEXTCTRL2" variable="TextCtrlB" member="yes">  <value>0</value>  <pos>305,190</pos>  <size>127,-1</size>  </object>  <object class="wxTextCtrl" name="ID\_TEXTCTRL3" variable="TextCtrlL" member="yes">  <value>0</value>  <pos>305,235</pos>  <size>127,-1</size>  </object>  <object class="wxStaticText" name="ID\_STATICTEXT1" variable="StaticText1" member="yes">  <label>Введите значение высоты в метрах:</label>  <pos>305,127</pos>  </object>  <object class="wxStaticText" name="ID\_STATICTEXT2" variable="StaticText2" member="yes">  <label>Введите значение B в градусах:</label>  <pos>305,172</pos>  </object>  <object class="wxStaticText" name="ID\_STATICTEXT3" variable="StaticText3" member="yes">  <label>Введите значение L в градусах:</label>  <pos>305,217</pos>  </object>  <object class="wxStaticText" name="ID\_STATICTEXT4" variable="StaticText4" member="yes">  <label>Значения СКО:</label>  <pos>306,294</pos>  </object>  <object class="wxTimePickerCtrl" name="ID\_TIMEPICKERCTRL1" variable="TimePickerCtrl1" member="yes">  <pos>305,93</pos>  <size>85,21</size>  <handler function="OnTimePickerCtrl1Changed" entry="EVT\_DATE\_CHANGED" />  </object>  <object class="wxButton" name="ID\_BUTTON4" variable="Button3" member="yes">  <label>для отладки/test</label>  <pos>303,463</pos>  <handler function="OnButton3Click2" entry="EVT\_BUTTON" />  </object>  <object class="wxStaticText" name="ID\_STATICTEXT5" variable="StaticText5" member="yes">  <label>Время прогнозирования по UTC(+3)</label>  <pos>305,49</pos>  </object>  </object>  </object>  </wxsmith> |

Приложение 13

|  |
| --- |
| #include "timeCalc.h"  #include <iostream>  #define \_USE\_MATH\_DEFINES  #include <math.h>  #include <ctime>  timeCalc::timeCalc(int date,  int month,  int year,  int hour,  int minutes,  int sec,  int m\_sec)  {  c\_date = date;  c\_month= month;  c\_year = year;  c\_hour = hour;  c\_minutes = minutes;  c\_sec = sec;  c\_m\_sec = m\_sec;  tm tm;  tm.tm\_year=year-1900;  tm.tm\_mon = month-1;  tm.tm\_mday = date;  tm.tm\_hour = hour;  tm.tm\_min=minutes;  tm.tm\_sec=sec;  time\_t time = mktime(&tm);  timeSec =time;  int dt=0;  if ((tm.tm\_year) > 81)  dt++;  if ((tm.tm\_year) > 82)  dt++;  if ((tm.tm\_year) > 83)  dt++;  if ((tm.tm\_year) > 85)  dt++;  if ((tm.tm\_year) > 87)  dt++;  if ((tm.tm\_year) > 89)  dt++;  if ((tm.tm\_year) > 90)  dt++;  if ((tm.tm\_year) > 92)  dt++;  if ((tm.tm\_year) > 93)  dt++;  if ((tm.tm\_year) > 94)  dt++;  if ((tm.tm\_year) > 95)  dt++;  if ((tm.tm\_year) > 97)  dt++;  if ((tm.tm\_year) > 98)  dt++;  if ((tm.tm\_year) > 105)  dt++;  if ((tm.tm\_year) > 108)  dt++;  if ((tm.tm\_year) > 112)  dt++;  if ((tm.tm\_year) > 115)  dt++;  if ((tm.tm\_year) > 116)  dt++;  dT = dt;  }  timeCalc::~timeCalc()  {  }  void timeCalc::timeGLNS()  {  tm tmm;  tmm.tm\_year=c\_year-1900;  tmm.tm\_mon = c\_month-1;  tmm.tm\_mday = c\_date;  tmm.tm\_hour = c\_hour;  tmm.tm\_min=c\_minutes;  tmm.tm\_sec=c\_sec;  tm GlonassVis;  tm tmGlonass;  tmGlonass.tm\_year=96;//год (1900 год = 0)  tmGlonass.tm\_mon = 0; // месяц года (январь = 0) [0,11]  tmGlonass.tm\_mday = 1;// день месяца [1,31]  tmGlonass.tm\_hour = 0; // часы после полуночи [0,23]  tmGlonass.tm\_min=0; //минуты после часов [0,59]  tmGlonass.tm\_sec=0; // секунды после минут [0,59]  timee glonass;  tmm.tm\_hour = tmm.tm\_hour+3;  time\_t time22 = mktime(&tmm);  time\_t timebaseGlonass = mktime(&tmGlonass);  double timeGlonass = time22 - timebaseGlonass;  glonass.numb\_fouryear\_period = (static\_cast<double>(tmm.tm\_year) - static\_cast<double>(tmGlonass.tm\_year))/4;  int deltayear = tmm.tm\_year%4;  GlonassVis.tm\_year = tmm.tm\_year - deltayear;  GlonassVis.tm\_mon = 0; // месяц года (январь = 0) [0,11]  GlonassVis.tm\_mday = 1;// день месяца [1,31]  GlonassVis.tm\_hour = 0; // часы после полуночи [0,23]  GlonassVis.tm\_min=0; //минуты после часов [0,59]  GlonassVis.tm\_sec=0; // секунды после минут [0,59]  time\_t time\_aftet\_vis\_year = mktime(&GlonassVis);  glonass.delta\_day\_after\_vis\_year = time22 - time\_aftet\_vis\_year;  glonass.day\_after\_vis\_year =glonass.delta\_day\_after\_vis\_year/(60\*60\*24); //дней после висок года .. учесть 1 дей.  glonass.week = timeGlonass/(60\*60\*24\*7);  glonass.sec\_after\_week=fmod(timeGlonass,(60\*60\*24\*7));  glonass.sum\_sec = tmm.tm\_hour\*60\*60 + tmm.tm\_min\*60 + tmm.tm\_sec;  GLNS\_sec\_since\_week = glonass.sum\_sec;  GLNS\_numb\_fouryear\_period = ceil(glonass.numb\_fouryear\_period); //N4 -for GLONASS ;  GLNS\_day\_after\_vis\_year = ceil(glonass.day\_after\_vis\_year) ;  }  void timeCalc::timeGPS()  {  tm tmgps;  tmgps.tm\_year=80;//год (1900 год = 0)  tmgps.tm\_mon = 0; // месяц года (январь = 0) [0,11]  tmgps.tm\_mday = 6;// день месяца [1,31]  tmgps.tm\_hour = 0; // часы после полуночи [0,23]  tmgps.tm\_min=0; //минуты после часов [0,59]  tmgps.tm\_sec=0; // секунды после минут [0,59]  time\_t timebase = mktime(&tmgps);  timee gps;  double time2 = timeSec;  gps.sec = (time2 - timebase)+dT;  gps.week = gps.sec/(60\*60\*24\*7);  //gps.sec\_after\_week = static\_cast<int>(round(gps.sec))%(60\*60\*24\*7);  gps.sec\_after\_week=fmod(gps.sec,(60\*60\*24\*7));  sec\_since\_week = (gps.sec\_after\_week);  week = static\_cast<int>(gps.week);  }  void timeCalc::timeGLL()  {  tm tmgalileo;  tmgalileo.tm\_year=99;//год (1900 год = 0)  tmgalileo.tm\_mon = 7; // месяц года (январь = 0) [0,11]  tmgalileo.tm\_mday = 22;// день месяца [1,31]  tmgalileo.tm\_hour = 0; // часы после полуночи [0,23]  tmgalileo.tm\_min=0; //минуты после часов [0,59]  tmgalileo.tm\_sec=0; // секунды после минут [0,59]  timee galileo;  time\_t timebasegalileo = mktime(&tmgalileo);  double time2 = timeSec;  galileo.sec = (time2 - timebasegalileo)+dT;  galileo.week = galileo.sec/(60\*60\*24\*7);  galileo.sec\_after\_week = fmod(galileo.sec,(60\*60\*24\*7));  week = static\_cast<int>(galileo.week);  sec\_since\_week = galileo.sec\_after\_week;  } |

Приложение 14

|  |
| --- |
| #ifndef TIMECALC\_H  #define TIMECALC\_H  struct timee{  double sec;  double week;  double sec\_after\_week;  double day\_after\_vis\_year;  double delta\_day\_after\_vis\_year ;  double sum\_sec;  double numb\_fouryear\_period;  double sec\_after\_week\_plus\_delta;  };  class timeCalc  {  public:  timeCalc(int date,  int month,  int year,  int hour,  int minutes,  int sec,  int m\_sec ); // msec пока не используется.  ~timeCalc(); // дописать  int dT; // поправки ко времени  int sec\_since\_week; //GPS,GALILEO  int week;//GPS,GALILEO  int GLNS\_sec\_since\_week;  int GLNS\_numb\_fouryear\_period; //N4 -for GLONASS ;  int GLNS\_day\_after\_vis\_year;//NT - for GLONASS  void timeGLNS();  void timeGPS();  void timeGLL();  double timeSec;  int c\_date;  int c\_month;  int c\_year; //в формате 2015  int c\_hour;  int c\_minutes;  int c\_sec;  int c\_m\_sec;  protected:  /\*double timeSec;  int c\_date;  int c\_month;  int c\_year; //в формате 2015  int c\_hour;  int c\_minutes;  int c\_sec;  int c\_m\_sec;\*/  };  #endif |

Приложение 15

|  |
| --- |
| #include "FTPdownl.h"  #include <windows.h>  #include <wininet.h>  #include <iostream>  #include <stdio.h>  bool download(LPCSTR server, LPCSTR login, LPCSTR pass, LPCSTR local\_file, LPCSTR remote\_file)  {  bool status;  HINTERNET hOpen, hConnection;  hOpen = InternetOpen(NULL, INTERNET\_OPEN\_TYPE\_DIRECT, NULL, NULL, 0);  if (hOpen == NULL)  return false;  hConnection = InternetConnectA(hOpen, server, 21, login, pass, INTERNET\_SERVICE\_FTP, INTERNET\_FLAG\_PASSIVE, 0 );  if (hConnection == NULL)  {  InternetCloseHandle(hOpen);  return false;  }  status=FtpGetFileA(hConnection, local\_file, remote\_file, true, 0, FTP\_TRANSFER\_TYPE\_UNKNOWN, 0);  InternetCloseHandle(hConnection);  InternetCloseHandle(hOpen);  return status;  } |

Приложение 16

|  |
| --- |
| #include <windows.h>  #ifndef FTPdonwl\_H  #define FTPdonwl\_H  bool download(LPCSTR server, LPCSTR login, LPCSTR pass, LPCSTR local\_file, LPCSTR remote\_file);  #endif |

Приложение 17

|  |
| --- |
| /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//\*\*  \* \file filename.с  \* \brief Краткое описание, назначение  \* \remarks Необязательная секция, для комментариев  \* \author Автор  \*  \* \if use\_svn\_keywords  \* :: $: Revision of last commit  \* :: $: Date of last commit  \* :: $: Author of last commit  \* \endif  \*  \* \b LICENSE \b INFORMATION \n  \* Copyright (c) Year, Company, City, Country  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Includes  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Подключение необходимого минимума заголовочных файлов  // Первым должен подключаться интерфейсный файл модуля  #include <stdio.h>  #include "parserGLNS.H"  //#include "parserGPS.h"  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Macro Definitions  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Локальные макроопределения  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Extern Data  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Объявления экземпляров экспортируемых данных  data\_almanax\_GLNS almanax\_GLNS[75];  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Local Data  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Локальные объявления типов и данных  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Local Function Prototypes  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Прототипы локальных функций (без комментариев)  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Function Pointers  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Объявление указателей на функции (без комментариев)  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Function Definitions  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Реализация функций  // Сначала реализация интерфейсных функций, далее реализация локальных функций  // Все функции должны иметь описание  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* END OF FILE  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  int parseGLNS(const char\* file)  {  int i;  int Numbb;  char systype;  char dummy[50];  FILE \*fd;  // string file = "MCCJ\_150306.AGP";  fd = fopen(file, "r");  //fd = fopen("test.txt", "r");  if (fd == NULL)  {  printf("ERORR");  }  else{ printf("OK"); }  while ( !feof(fd) ) // пока не конец файла  {  // almanax\_GLNS[i-1].systype = systype;  //Строка 1  fscanf(fd, "%u", (int \*)dummy); //1 - число пhttps://vk.com/doc165113148\_595394058?hash=f2e6d67d96e5f1a609&dl=ab5ddaf49d0048c0feолучения альманаха  fscanf(fd, "%u", (int \*)dummy); //2 - месяц получения альманаха  fscanf(fd, "%u", (int \*)dummy); //3 - год получения альманаха  fscanf(fd, "%u", &Numbb); //4 -  //Строка 2  fscanf(fd, "%u", &(i));//1 - номер PRN  almanax\_GLNS[i-1].PRN=i;  almanax\_GLNS[i-1].NN = Numbb;  fscanf(fd, "%u", (int \*)dummy);//2 - номер частотного слота (-7 - 24)  fscanf(fd, "%u", (int \*)dummy);//3 - признак здоровья по альманаху (0 - 1)  fscanf(fd, "%u", &(almanax\_GLNS[i-1].date));//4 - число  fscanf(fd, "%u", &(almanax\_GLNS[i-1].month));//5 - месяц  fscanf(fd, "%u", &(almanax\_GLNS[i-1].year));//6 - год  fscanf(fd, "%lf", &(almanax\_GLNS[i-1].tLA));//7 - время прохождения первого узла, на которое все дано, с  fscanf(fd, "%lf", (int \*)dummy);//8 - поправка ГЛОНАСС-UTC, с  fscanf(fd, "%lf", (int \*)dummy);//9- поправка GPS-ГЛОНАСС, с  fscanf(fd, "%lf", (int \*)dummy);//10 - поправка времени КА ГЛОНАСС относительно системного времени, с  //Строка 3  fscanf(fd, "%lf", &( almanax\_GLNS[i-1].Lam ));//1 - Lam - долгота узла, полуциклы  fscanf(fd, "%lf", &( almanax\_GLNS[i-1].dI));//2 - dI -коррекция наклонения, полуциклы  fscanf(fd, "%lf", &( almanax\_GLNS[i-1].w));//3 - w - аргумент перигея, полуциклы  fscanf(fd, "%lf", &( almanax\_GLNS[i-1].E));//4 - E - эксцентриситет  fscanf(fd, "%lf", &( almanax\_GLNS[i-1].dT));//5 - dT - поправка к драконическому периоду, с  fscanf(fd, "%lf", &( almanax\_GLNS[i-1].dTT));//6 - dTT - поправка к драконическому периоду, с/виток  almanax\_GLNS[i-1].PRN=i;  }  int imax=i;  almanax\_GLNS[0].mass = imax;  fclose(fd);  //printf("num\_week = %d " , almanax\_GPS[1].num\_week);  return imax;  } |

Приложение 18

|  |
| --- |
| #ifndef PARSERGLNS\_H  #define PARSERGLNS\_H  #ifdef \_\_cplusplus  extern "C" {  #endif  typedef struct  {  int NN;  char systype;  int date;  int month;  int year;  double t\_almanax;  unsigned int PRN;  double tLA; // строка 2 -п.7  double Lam ;  double dI;  double w;  double E;  double dT;  double dTT;  int mass;  } data\_almanax\_GLNS;  extern data\_almanax\_GLNS almanax\_GLNS[75];  int parseGLNS(const char\* file);  #ifdef \_\_cplusplus  }  #endif  #endif |

Приложение 19

|  |
| --- |
| /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//\*\*  \* \file filename.с  \* \brief Краткое описание, назначение  \* \remarks Необязательная секция, для комментариев  \* \author Автор  \*  \* \if use\_svn\_keywords  \* :: $: Revision of last commit  \* :: $: Date of last commit  \* :: $: Author of last commit  \* \endif  \*  \* \b LICENSE \b INFORMATION \n  \* Copyright (c) Year, Company, City, Country  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Includes  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Подключение необходимого минимума заголовочных файлов  // Первым должен подключаться интерфейсный файл модуля  #include <stdio.h>  #include "parserGPS.H"  //#include "parserGPS.h"  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Macro Definitions  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Локальные макроопределения  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Extern Data  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Объявления экземпляров экспортируемых данных  data\_almanax almanax\_GPS[75];  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Local Data  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Локальные объявления типов и данных  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Local Function Prototypes  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Прототипы локальных функций (без комментариев)  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Function Pointers  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Объявление указателей на функции (без комментариев)  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Function Definitions  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  // Реализация функций  // Сначала реализация интерфейсных функций, далее реализация локальных функций  // Все функции должны иметь описание  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* END OF FILE  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  int parseGPS(const char\* file)  {  int i;  int Numbb;  char systype;  char dummy[50];  FILE \*fd;  // string file = "MCCJ\_150306.AGP";  fd = fopen(file, "r");  //fd = fopen("test.txt", "r");  if (fd == NULL)  {  printf("ERORR");  }  else{ printf("OK"); }  while ( !feof(fd) ) // пока не конец файла  {  // almanax\_GPS[i-1].systype = systype;  //Строка 1  fscanf(fd, "%u", (int \*)dummy); //1 - число получения альманаха  fscanf(fd, "%u", (int \*)dummy); //2 - месяц получения альманаха  fscanf(fd, "%u", (int \*)dummy); //3 - год получения альманаха  fscanf(fd, "%u", &Numbb); //4 -  //Строка 2  fscanf(fd, "%u", &(i));//1 - номер PRN  almanax\_GPS[i-1].PRN=i;  almanax\_GPS[i-1].NN = Numbb;  fscanf(fd, "%u", (int \*)dummy);//2 - обобщенный признак здоровья  fscanf(fd, "%u", &( almanax\_GPS[i-1].num\_week));//3 - неделя GPS (альманаха)  fscanf(fd, "%u", &( almanax\_GPS[i-1].time\_week));//4 - время недели GPS, с (альманаха)  fscanf(fd, "%u", (int \*)dummy);//5 - число  fscanf(fd, "%u", (int \*)dummy);//6 - месяц  fscanf(fd, "%u", (int \*)dummy);//7 - год  fscanf(fd, "%lf", &( almanax\_GPS[i-1].t\_almanax));//8 - время альманаха, с  fscanf(fd, "%lf", (int \*)dummy);//9 - поправка времени КА GPS относительно системного времени, с  fscanf(fd, "%lf", (int \*)dummy);//10- скорость поправки времени КА GPS относительно системного времени, с/с  fscanf(fd, "%lf", &( almanax\_GPS[i-1].vOm0));//11- Om0 - скорость долготы узла, полуциклы/c  //Строка 3  fscanf(fd, "%lf", &( almanax\_GPS[i-1].Om0));//1 - Om0 - долгота узла, полуциклы  fscanf(fd, "%lf", &( almanax\_GPS[i-1].I));//2 - I - наклонение, полуциклы  fscanf(fd, "%lf", &( almanax\_GPS[i-1].w));//3 - w - аргумент перигея, полуциклы  fscanf(fd, "%lf", &( almanax\_GPS[i-1].E));//4 - E - эксцентриситет  fscanf(fd, "%lf", &( almanax\_GPS[i-1].sqrtA));//5 - SQRT(A) - корень из большой полуоси, м\*\*0.5  fscanf(fd, "%lf", &( almanax\_GPS[i-1].M0));//6 - M0 - средняя аномалия, полуциклы  almanax\_GPS[i-1].PRN=i;  }  int imax=i;  almanax\_GPS[0].mass = imax;  fclose(fd);  //printf("num\_week = %d " , almanax\_GPS[1].num\_week);  return imax;  } |

Приложение 20

|  |
| --- |
| #ifndef PARSERGPS\_H  #define PARSERGPS\_H  #ifdef \_\_cplusplus  extern "C" {  #endif  typedef struct  {  int NN;  char systype;  int num\_week;  int time\_week;  double t\_almanax;  // double dt;  // double vt;  unsigned int PRN;  double vOm0;  double Om0;  double I;  double w;  double E;  double sqrtA;  double M0;  int mass;  } data\_almanax;  extern data\_almanax almanax\_GPS[75];  int parseGPS(const char\* file);  #ifdef \_\_cplusplus  }  #endif  #endif |

Приложение 21

|  |
| --- |
| #include "ephemerids.h"  #define \_USE\_MATH\_DEFINES  #include <math.h>  #include <windows.h>  #include <wininet.h>  #include <iostream>  #include <string>  #include <stdio.h>  #include <wx/msgdlg.h>  #include <wx/string.h>  #include <wx/textfile.h>  #include <wx/dialog.h>  #include <wx/msgdlg.h>  #include <wx/spinctrl.h>  #include <wx/intl.h>  #include <wx/settings.h>  Coordinates ephemerids(double toe,  double t\_almanax,  double almanax\_M0,  double sqrtA,  double E,  double I,  double Om0,  double time\_week )  {//toe - время на которое нуэно расчитать;  Coordinates Coordinates;  double M0 = almanax\_M0\*M\_PI;  double A0 = pow((sqrtA),2);  double en = E;  double omegan = 0.16578805\*M\_PI;  double i0n = I\*M\_PI;  double Omega0n = Om0\*M\_PI;  double nu = 3.986005\*pow(10,14); //из икд  double OMEGA\_REF=(-2.6\*pow(10,-9))\*M\_PI;//из икд  double OMEGA\_e=7.2921151467\*pow(10,-5);//из икд  int week\_almanax = time\_week;  double t\_sec\_almanax = t\_almanax;  double tk = toe- t\_sec\_almanax;  if (tk > 302400)  {  tk = 604800-tk;  }  else if (tk<-302400)  {  tk = 604800+tk;  }  double n0 = sqrt(nu/pow(A0,3));  double nA = n0;  double Mk = M0 + nA\*tk;  double Ek;  double Ekold = 0;  Ek = en\*sin(0)+Mk;  while (abs(Ek- Ekold)>0.000000001 )  { Ekold = Ek;  Ek = en\*sin(Ek)+Mk;  }  double vk = atan2(sqrt(1-(pow(en,2)))\*sin(Ek), (cos(Ek)-en) );  double Ak = A0;  double rk = Ak\*(1-en\*cos(Ek));  double Fk = vk + omegan;  double uk = Fk;  double xkk = rk\*cos(uk);  double ykk = rk\*sin(uk);  double OMEGA = OMEGA\_REF;  double OMEGAk = Omega0n+(OMEGA-OMEGA\_e)\*tk-OMEGA\_e\*t\_sec\_almanax;  double ik = i0n;  double xk = xkk\*cos(OMEGAk)-ykk\*cos(ik)\*sin(OMEGAk);  double yk = xkk\*sin(OMEGAk)+ykk\*cos(ik)\*cos(OMEGAk);  double zk = ykk\*sin(ik);  Coordinates.X = xkk\*cos(OMEGAk)-ykk\*cos(ik)\*sin(OMEGAk);  Coordinates.Y = xkk\*sin(OMEGAk)+ykk\*cos(ik)\*cos(OMEGAk);  Coordinates.Z = ykk\*sin(ik);  return Coordinates;  } |

Приложение 22

|  |
| --- |
| #ifndef EPHEMERIDS\_H  #define EPHEMERIDS\_H  typedef struct  {  double X;  double Y;  double Z;  } Coordinates;  Coordinates ephemerids(double toe,  double t\_almanax,  double almanax\_M0,  double sqrtA,  double E,  double I,  double Om0,  double time\_week );  #endif |