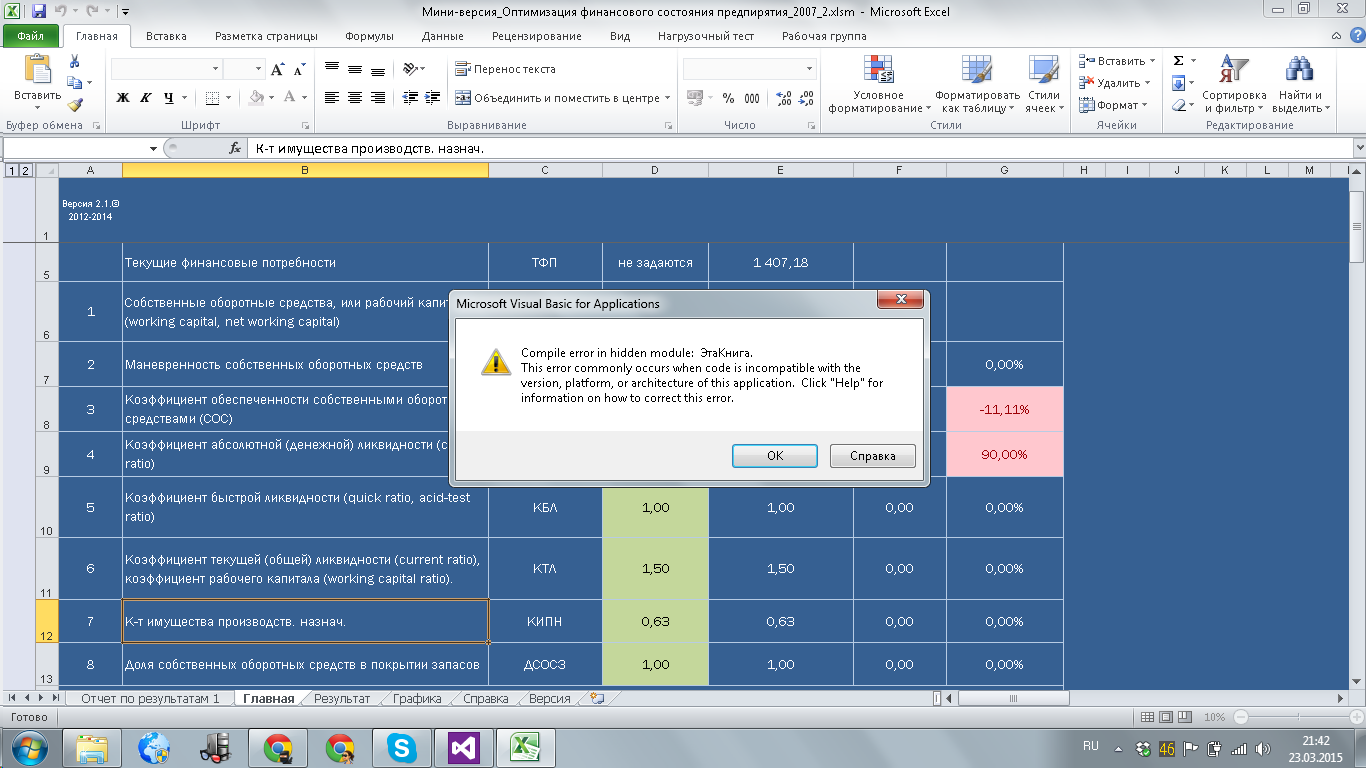
**Программа для финансового анализа**

1. **Требования к программному обеспечению.**

Исходя из технического задания, программа должна удовлетворять следующим требованиям:

* ПО должно быть выполнено в виде отдельного приложения, работающего в среде Windows 7 и выше;
* ПО должно корректно работать в 32-разрядных или 64-разрядных версиях Windows? Это имеет значение для выбора методов реализации обработки данных. Или независимо от платформы, что требует подхода к универсальности ПО. Уточнение потому что версия Windows 7 и выше априори предполагает 64 bit разрядность, но бывает и 32 bit, что может помешать работе модуля работы с данными если это не предусмотреть. Плюс к этому, ваша мини-версия у меня запустилась с ошибкой. Равно как и на любое другое действие, кроме справки. У меня ОС Windows 7 Максимальная 64 bit, MS Office 2010 Professional 64 bit. Макросы, которые реализуют функционал, пришлось включать вручную, автоматически они не подтянулись.



* Какая средняя конфигурация локальной машины? Средняя загруженность и скорость работы? Это информация будет полезна при учете быстродействия ПО;
* Какой основной половозрастной состав пользователей системы и каково их количество? Эта информация важна для цветовой гаммы программы с учетом психофизических факторов;
* Может ли программа требовать установки дополнительного программного обеспечения для корректной работы? К примеру, если рассмотреть возможность использования SQL – сервера;
* Должен быть реализован удобный и интуитивно понятный графический интерфейс для пользователя;
* Для разработки программы будет использоваться объектно-ориентированный язык программирования C# в среде Microsoft Visual Studio 2012 Ultimate. Обоснование выбора заключается в функциональности MS Visual Studio, т. к. система включает в себя интегрированную среду разработки, которая позволяет максимально удобно и эффективно создавать приложение с поддержкой графического интерфейса.
* Самый основной вопрос к требованиям, что не отменяет концепции отдельного Windows – приложения – это форма реализации. Программа условно делится на 3 модуля: ввод многочисленных данных и показателей (или выбор данных из существующих справочников), основной расчетный модуль (включая, разумеется, симплекс-метод) и вывод результатов в графической интерпретации (графики, диаграммы) и обычной текстовой. Т.к., данные (задаваемые или результаты) должны храниться, то требуется механизм по работе с данными. Есть 3 варианта:

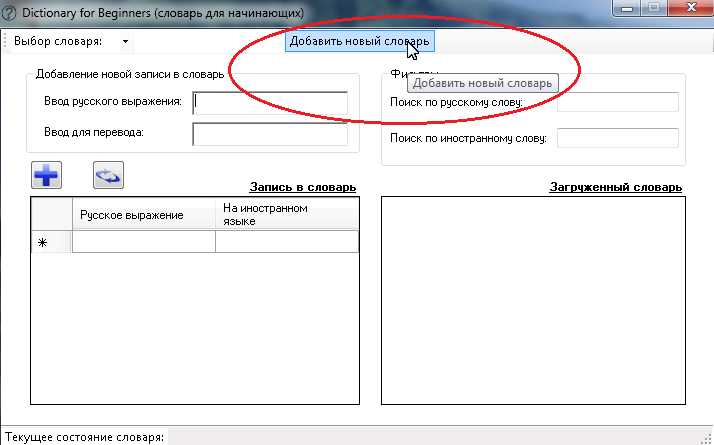
1. Загрузка данных в программу, запись в заданный файл, сохранение в отдельный файл, удаление, сохранение базы, построение визуализации и т. д, т. е, все стандартные механизмы работы с данными с привлечением MS Office Excel или Access. Т. е, зарезервированная база Access или книга Excel выступают в качестве источника данных (если требуются, конечно, хранить какие-то постоянные показатели во внешнем файле, а не в программе, сами данные программы не хранят), но по большей части в качестве места сохранения результатов. К данным впоследствии можно обращаться в любой удобный момент, выбрав явно название из диалогового окна выбора файла или задав диапазон дат или же указав конкретную дату, за которую должен открыться (или загрузиться в программу) отчет. Плюс данного метода – это относительная простота в реализации, широкий диапазон возможностей (которые уже реализовано, главное правильно применить, в этом и плюс ООП), отсутствие доп. программных средств, которые могут усложнить разработку и работу приложения в целом. Это *основной метод* работы с данными, применяемый к объектам MS Office, который используют, если решают отойти от вопроса формирования базы данных доп. средствами. Реализуется обобщенно двумя технологиями. Минус использования стандартных методов для работы с объектами office (методы определены самим Microsoft) – это невысокие показатели по быстродействию. Эффективно повлиять на эти показатели особой нет возможности, обусловлено самими методами. Поэтому нужно понимать, что если объем данных существенный (пусть не сразу, но мере накопления) и характеристики машины не самые высокие, то в скорости операций с данными – это будет заметно. Все-таки есть разница с загрузкой в 1-2 секунды и 10-20 к примеру. Но еще раз повторюсь, без базы данных, доп. средств, которые тоже могут нести нагрузку и с учетом простоты реализации – это самый оптимальный вариант. К тому как указано выше очень широкий механизм инструментов, можно получить практически любой механизм оffice в своей программе, не используя офис, это очень удобно для визуализации графики, поскольку привлекать сторонние графические библиотеки не всегда эффективно с точки зрения и кода, и производительности.

Установка происходит локально, никто друг от друга не зависит.

1. Установка дополнительного ПО для работы с базами данных и создание базы данных. Речь идет от Sql – сервере. Полноценный сервер (бесплатный express версии) или различные упрощенные аналоги, наподобие **sql compact**. Плюс очевиден. Полноценная база, проиндексированная, с встроенным механизмом защиты и контроля доступа и самое основное высокая скорость работы, ни секунды, а миллисекунды. Но минус, это усложняет разработку, предъявляет требования к установке специализированных программных средств на каждую локальную машину, что не является эффективным. И лучше базу данных делать одну, хранить на одном сервере и настраивать доступы пользователей к базе. Самый логичный и правильный вариант, но в то же время при небольшом количестве пользователей неэффективно. Да и разработка существенно осложнится. Если настраивать на каждом компьютере отдельно как локальный сервер со своей локальной базой, то такой вариант возможен, но требует больше времени для настройки и установки.
2. Если не использовать сервер и создавать на нем базу данных, но методы работы и скорость как на сервере все-таки хочется, то можно использовать компромиссное решение специальных библиотек, которые реализуют весь функционал БД, но устанавливать при этом ничего не требуется, поскольку все зашито в одном небольшом файле. И вся программа, по сути, состоит из 3х файлов: самого приложения, dll – библиотеки для базы и локальной базы, которая создана этой библиотекой. Самый оптимальный вариант [SQLite](https://ru.wikipedia.org/wiki/SQLite). Одна из лучших в своем роде, полностью бесплатна и легальна, что немаловажно, т.к., хорошие функциональные библиотеки и расширения обычно платные. В этом случае потребуется больше времени, чтобы применить к вашей программе. Т.к., на стандартных функциях работы с базами данных, функционал этой библиотеки заканчивается. Чтобы на основе ее получать данные, совершать с ними дополнительные действия, в том числе визуализировать, то придется разработать свои методы, а не использовать те, которые предоставляет среда разработки с объектами того же офиса. На самом деле, это, пожалуй, лучший вариант исполнения с точки зрения возможностей, уникальности, масштабируемости и самое главное быстродействия. Но это серьезно увеличит срок. Практически все с нуля создавать.

Вопрос к вам, какой вариант ближе вам. Чего сами хотите или требования от пользователей, или руководства, как видите программу.

1. Все остальные блоки пояснений не требуют. Все обосновано четко и ясно, ввод данных, расчет, проверка параметров перед расчетом, получения результатов, дополнительные расчеты, визуализация, прогноз. Должна быть реализована гибкая система отклика программы на действия пользователя во время всего цикла работы: обработка ошибок, контроль ввода данных, контроль использования инструментов. Использование всплывающих сообщений с пояснениями (пример ниже).



Вопросы, которые возникают:

* Вы хотели предоставить специальный алгоритм оценки. Вы имели в виду блок-схемы, алгоритм решения задачи, исходный код модулей (которые у вас под паролем, и половина не доступны к использованию в мини-версии) или же просто алгоритм – это перечень действий по вводу данных и последовательного нажатия нескольких кнопок. Тогда это не алгоритм, а обычная логика любой расчетной программы. Интерпретировать ваш рабочий метод расчета симплекс метода на C# и по возможности оптимизировать, как минимум для времени решения и обработки ошибок – это одно, равно как и другие модули, если сохранить используемую логику. Разработать независимо от этого используя только перечень данных, а поток также протестировать и оптимизировать, и разработать свою логику программы – это другое. Будут ли предоставлены какие-то дополнительные сведения.
* Есть ли возможность получить исходные данные перечень всех переменных, поставку и описания задачи решения симплекс метода с ограничениями в формализованном виде, а не по листу Екселя, плюс описания остальных функций? Очевидно, изначально задача ставилась не в виде разбросанных данных на листе. Крайне неудобно работать. В любом случае придется описывать для себя в требуемом виде. Если это уже есть, то тем проще.

1. **По дополнительным требованиям.**

* Возможность внесения данных в ограничения, отключения расчетных модулей и т.д., могу предложить завести в программу администратора, который будет управлять всеми ограничениями, которые будут недоступны в обычном использовании. Или банально на уровне паролей, будут неактивные элементы или действия, те же кнопочки, вызов окна для изменения данных и т.д. Т.е., чтобы активировать нужно, ввести пароль администратора.
* Касаемо привязки программы к конкретной машине. Можно при запуске программы делать двухэтапную проверку: логин пароль пользователя, hostname или ip – адрес конкретной машины. Для программы заполнения бланков по ПДД, если вы просматривали мое портфолио, я так и делал. Работает на ура. Если данные меняются, то опять-таки под паролем администратора всегда можно изменить данные и продолжать работать дальше.

1. **Итого:**

Для реализации программы в ней должно быть решено:

1. Разработка интерфейса программы с учетом пожеланий к удобству, настройки цветовой гаммы и набора данных.
2. Ввод данных. С возможностью хранения или загрузки из файла постоянных показателей с возможностью их изменять при необходимости.
   1. Проверка на корректность заведения данных.
3. Расчет основной задачи симплекс методом, расчет дополнительных показателей на основании полученного решения.
   1. Интерпретация имеющихся алгоритмов на новый язык с сохранение бизнес – логики старой программы или же разработка своих алгоритмов с новой логикой.
   2. Тестирование алгоритмов, оптимизация.
   3. Проверка на заполнение необходимых полей с проверкой выполнения условий для корректного решения задачи.
4. Визуализация данных по найденному решению.
5. Реализовать алгоритм прогнозирования данных на основе существующих данных.
6. Блок работы с данными: сохранение, добавления, редактирования, удаления, загрузки в базу или из базы в программу.
7. Реализовать модуль печати.
8. Разработать модуль двухэтапной аутентификации через логин пароль (возможно в зашифрованном виде, если есть желание) и привязку к локальной машине, с возможностью совершения операций с полным доступом под паролем администратора впоследствии.
9. Тестирование и отладка программы помодульно на этапе разработки и при завершении основных работ.
10. Оптимизация по быстродействию, обработка исключений.
11. Повторное тестирование в разных средах Windows.
12. Написание документации (модуль help или FAQ) если потребуется.
13. Создание дистрибутивов для установки.
14. Заложить доп. время под риски:

* Исправление в случае непредвиденных обстоятельств. Банально, на вашей среде не будет корректно работать какой-то модуль, но при этом он протестирован и ошибок не содержит. Поиск проблемы и решения. В итоге, может дополнительные настройки, программы или скрипты влияют на программу. Далеко за примером ходить не надо, помимо ручного включения всех макросов, которые автоматически не запустились, пришлось еще отключить ряд екселевских надстроек, которые никакого отношения к вашему файлу не имели.
* В ходе работы появятся новые обстоятельства, изменение существующих механизмов, добавления новых.

1. **Примерная оценка по стоимости и времени выполнения.**

Конечную оценку окончательно можно сделать после выбора формата работы программы (из 3х описанных), согласования всех функций, заложенных в программе.

Предлагаю следующую оценку. Не учитывать характер уникальности, инвестирование на дальнейшую эксплуатацию и возможные доработки в сопровождении, сложность используемых объектов и механизмов, а учитывать только человеко-час в качестве подсчета трудозатрат. Наиболее объективная оценка. Для реализации вашей программы и всех механизмов, которые я описал моя оценка 125 часов. Это если заниматься программой все 25 дней, а не от случая к случаю, подходя качественно и фундаментально. После уточнения формата программы и функций количество часов может быть снижено или наоборот увеличено. Но в любом случае меньше 100 часов не будет при таком объеме.