**МИНОБРНАУКИ РОССИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

Факультет компьютерных наук

Кафедра информационных систем и технологий

«Система учёта личной бухгалтерии»

Курсовая работа по дисциплине

«Технологии программирования»

Обучающийся \_\_\_\_\_\_\_\_\_\_ *И.С. Господарикова, 3 курс, д/о*

Обучающийся \_\_\_\_\_\_\_\_\_\_ *И.В. Гражданкин, 3 курс, д/о*

Руководитель \_\_\_\_\_\_\_\_\_\_\_ *А.С. Миронова, ассистент*

Воронеж, 2018

[1 Анализ предметной области 3](#_Toc511638103)

[1.1 Обзор предметной области задачи 3](#_Toc511638104)

[1.2 Терминология предметной области 5](#_Toc511638105)

[2 Постановка задачи 6](#_Toc511638106)

[3 Анализ работы 7](#_Toc511638107)

[3.1 Анализ структуры разрабатываемой системы 7](#_Toc511638108)

[3.2 Анализ процессов в разрабатываемой системе 10](#_Toc511638109)

[3.3 Разработка макета структуры хранимых данных 12](#_Toc511638110)

[3.4 Макет интерфейса 14](#_Toc511638111)

[3.5 Анализ структуры занятости и планирование временного деления 16](#_Toc511638112)

1. Анализ предметной области
   1. Обзор предметной области задачи

Для понимания цели работы, стоит обратиться к рассмотрению закономерностей и особенностей предметной области. Ведение личной или семейной бухгалтерии – это то, с чем мы все, в том или ином виде, сталкиваемся ежедневно, поэтому представить специфику будет не так сложно.

Ведение любой бухгалтерии всегда предполагало работу с большими объёмами данных, трудных к запоминанию человеком. Ещё в древности люди придумывали различные системы записи, приспособления для облегчения вычислений, и в первую очередь это было обусловлено необходимостью совершенствования счёта денег. Сегодня же существует масса средств.

Также характерной чертой задач подобного рода являются достаточно высокие требования к точности и быстроте вычислений. Стоит отметить, что с течением времени они только растут, и если когда-то для нужд торговой фирмы хватало производительности отдела из нескольких бухгалтеров, то теперь может не доставать сети компьютеров со специализированным программным обеспечением.

Ведение же бухгалтерии личного характера имеет ряд особенностей, которые тоже следует рассмотреть в контексте анализа данной области. Для начала, следует отметить, что массивы данных о семейных тратах, хотя и являются большими, на порядки меньше, чем в задачах с бухучётом фирм. Так как современные вычислительные механизмы обладают достаточно большими значениями тактовой частоты, требования относительно быстроты вычислений несколько теряют значимость. Взамен этого обозначаются новые критерии качества программного продукта. Обратим на них особенное внимание в стремлении выявления главных аспектов задачи и обозначения целей разработки.

Одним из таких критериев можно назвать интуитивно понятный интерфейс программы. Программное обеспечение данного типа должно ориентироваться на широкую аудиторию и быть доступным для человека с самым общим уровня познаний в бухгалтерском деле. Также, в целях повышения эффективности работы с приложением, ввод данных должен занимать как можно меньше времени, поэтому при проектировании интерфейса следует постараться сделать его как можно более удобным.

Другой чертой программы для учёта личного бюджета или бухгалтерии семьи считается представленная пользователям функциональность. Чтобы удовлетворять потребности целевой аудитории, а, следовательно, пользоваться большим спросом среди продуктов такого характера, приложение должно иметь сферой своего применения нечто, большее простого подсчёта разницы между доходом и расходом денежных средств. Именно поэтому очень важно ещё на стадии планирования разработки проекта задать широкий спектр полезных для пользователя возможностей. К примеру, многие уже известные приложения этой области реализуют такие возможности как:

* учёт разнообразных статей дохода семьи;
* возможность задания разного рода статей расхода;
* пересчёт статей бюджета в разных валютах с учётом курсов;
* возможность календарного планирования и прогнозирования;
* калькулятор для расчёта кредитов и вкладов с учётом процентной ставки;
* широкие возможности анализа вводимых данных;
* построение отчётов (в том числе с графической визуализацией).

Отдельное внимание уделяется большинством пользователей графическому оформлению аналогичных программ. Такие возможности как назначение статьям бюджета графических маркеров в виде разнообразных значков и пользовательский выбор цветовых тем могут сыграть едва ли не решающую роль в выборе приложения для учёта личной бухгалтерии. Отсюда следует, что эти возможности тоже следует не упускать из вида и реализовывать в приятном глазу формате.

* 1. Терминология предметной области

1. Бюджет – это смета ухода и прихода денежных средств из собственности и в собственность того или иного объекта, расписанная на определённый срок. В данном контексте будем говорить о бюджете семьи или отдельного физического лица. Так же словом «бюджет» часто обозначается сумма денег в собственности оговариваемого объекта.
2. Статья бюджета – это некоторый пункт бюджета. Статьи бюджета, для удобства реализации, разделим на регулярные (такие как заработная плата или ежемесячное погашение кредита) и нерегулярные (как покупка машины или продажа дачного дома).
3. Статьёй расхода называется пункт бюджета, на который денежные средства тратятся.
4. Статьёй дохода называется пункт, который их в бюджет привносит.
5. Трата по статье расхода – это конкретная выбывшая из бюджета на некоторые нужды сумма.
6. Сбережения – это разница между доходом и расходом, не использованная в период времени, когда была получена, и предназначенная для удовлетворения потребностей в будущем.
7. Постановка задачи

Целью выполнения данной работы является создание приложения для учёта личной (семейной) бухгалтерии, ориентированного на персональный компьютер под управлением операционной системы Windows[]. Продукт должен будет обеспечивать возможность реализации следующих задач:

1. Составление личного (семейного) бюджета – задание статей сметы на некоторый период, которое включает в свою очередь реализацию таких функций как:

* добавление и удаление статей бюджета;
* редактирование статей бюджета.

1. Календарное распределение статей бюджета – методы, реализующие возможности, связанные с установкой дат:

* непосредственное назначение даты;
* возможность создания статьи, повторяющейся периодически.

1. Реализация методов работы со сберегаемыми средствами – должна быть предусмотрена возможность откладывания некоторых средств на различные цели в будущем.
2. Возможности для анализа данных бюджета за прошедшие периоды бухгалтерского учёта, для чего предполагается осуществить следующие возможности:

* подведение общей статистики и статистики за период;
* представление данных в виде графиков и диаграмм.

1. Возможности отслеживания курсов мировых валют, пересчёты бюджета для разных валют.
2. Реализация интуитивно понятного эргономичного интерфейса.
3. Анализ работы
   1. Анализ структуры разрабатываемой системы

Для анализа требований обратимся диаграмме вариантов использования данной системы на рисунке 1.

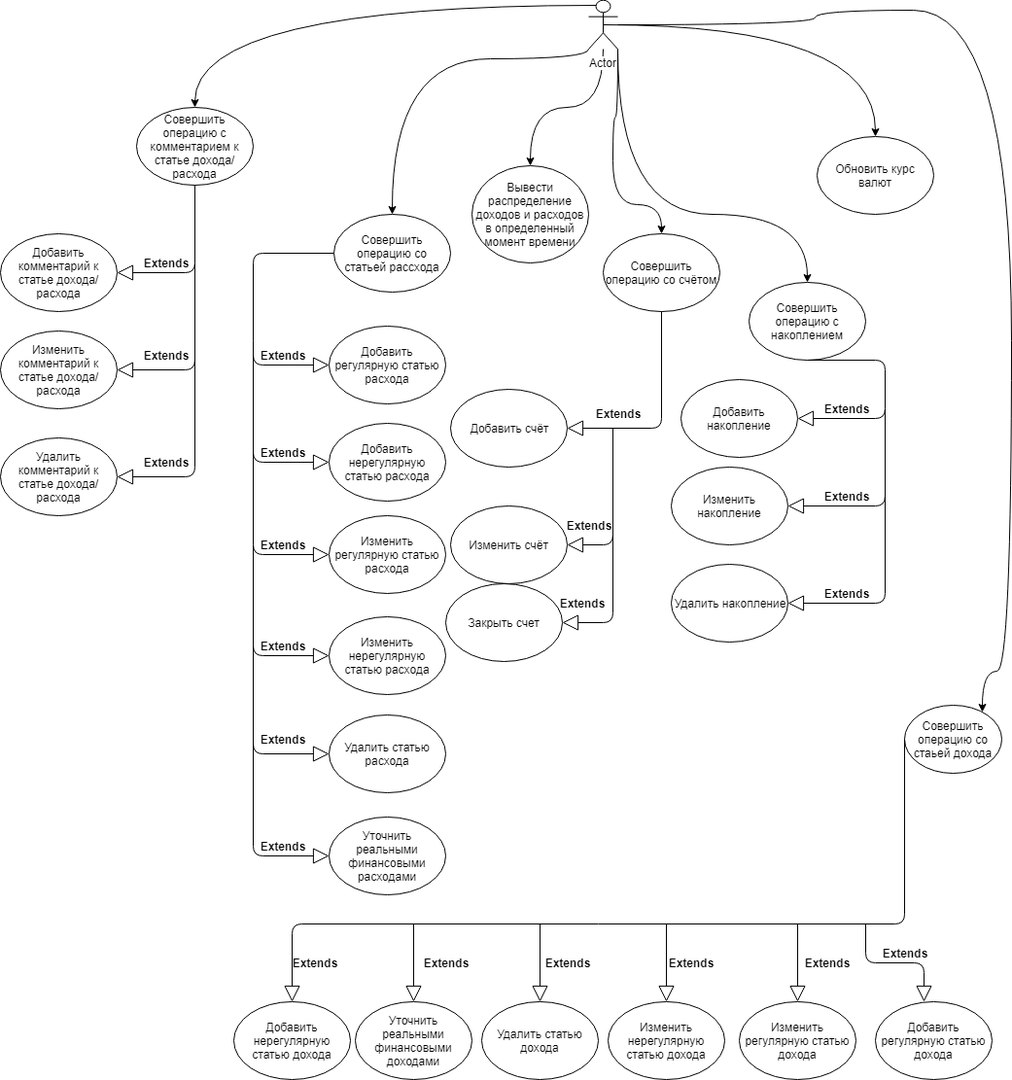


Рисунок 1 – Диаграмма вариантов использования.

На диаграмме находится единственный актор — это пользователь. Пользователь — это человек, который имеет учётную запись в системе и использует программу для учёта движения личных и/или семейных финансов.

Рассмотрим подробнее варианты использования, представленные на рисунке 1:

1. «Совершить операцию со статьей дохода» - расширяется следующими вариантами использования:

* «Добавить регулярную статью дохода» - пользователь добавляет статью дохода, относящуюся к категории регулярных статей, таких, например, как зарплата.
* «Добавить нерегулярную статью дохода» - пользователь добавляет статью дохода, для которой неизвестна или отсутствует регулярность её возникновения, например, премия или вырученные за продажу чего-либо деньги.
* «Изменить регулярную статью дохода» - пользователь при возникновении такой необходимости изменяет информацию о статье дохода, н-р, название, величину дохода, регулярность.
* «Изменить нерегулярную статью дохода» - пользователь изменяет информацию о нерегулярной статье дохода.
* «Удалить статью дохода» - пользователь удаляет статью дохода (не имеет значения, какую — регулярную или нерегулярную)

1. «Совершить операцию со статьей расхода» - расширяется следующими вариантами использования:

* «Добавить регулярную статью расхода» - пользователь добавляет статью расхода, возникающую регулярно, с определённой частотой. Пример: еда, проезд на общественном транспорте.
* «Добавить нерегулярную статью расхода» - пользователь добавляет статью расхода, которая возникает неожиданно для пользователя или не имеет частоты возникновения. Пример: лекарства, замена сломавшейся мебели.
* «Изменить регулярную статью расхода» - пользователь изменяет информацию о статье расхода, такую, как название, величина расходов, регулярность расходов по данной статье.
* «Изменить нерегулярную статью расхода» - пользователь изменяет информацию о нерегулярной статье расхода.
* «Удалить статью расхода» - пользователь удаляет регулярную или нерегулярную статью расхода.

1. «Совершить операцию с комментарием к статье дохода/расхода» - расширяется следующими вариантами использования:

* «Добавить комментарий к статье дохода/расхода» - пользователь может добавить комментарий к какой-либо статье в его бюджете для дополнительного описания этой статьи.
* «Изменить комментарий к статье дохода/расхода» - пользователь может изменить комментарий к статье бюджета в случае, если не удовлетворён имеющимся комментарием.
* «Удалить комментарий к статье дохода/расхода» - пользователь может удалить имеющийся комментарий в случае отсутствия необходимости в нём.

1. «Вывести распределение доходов и расходов за определённый момент времени» - задавая промежуток времени, пользователь может посмотреть соотношение доходов и расходов за этот промежуток, а также соотношение регулярных/нерегулярных доходов и расходов.
2. «Совершить операцию со счётом» - расширяется следующими вариантами использования:

* «Добавить счёт» - пользователь добавляет счёт — именованный набор определённых статей дохода и расхода, для удобства наблюдения за ними.
* «Изменить счёт» - пользователь изменяет данные счёта — название, набор относящихся к нему статей дохода и расхода.
* «Закрыть счёт» - пользователь удаляет счёт.

1. «Совершить операцию с накоплением» - расширяется следующими вариантами использования:

* «Добавить накопление» - пользователь добавляет счёт особого типа, с которым может совершать операции только добавления статей дохода или удаления.
* «Изменить накопление» - пользователь изменяет информацию о накоплениях, такую, как название и назначение счёта, срок закрытия (при наличии).
* «Закрыть накопление» - пользователь закрывает накопления при достижении цели открытия счёта или при возникновении необходимости каким-либо иным образом распорядиться средствами этого счёта.

1. «Совершить операцию с валютой» - расширяется следующими вариантами использования:

* «Обновить курс валюты» - пользователь может для актуализации информации обновить курс валюты вручную или автоматически (при наличии подключения к сети Интернет).
  1. Анализ процессов в разрабатываемой системе

Для более детального видения процессов, происходящих в разрабатываемой системе, построим диаграмму последовательностей для некоторых вариантов использования.

На рисунке 2 мы можем увидеть диаграмму последовательностей для процесса создания и добавления статьи в бюджет и выведения сведений о доходах и расходах.

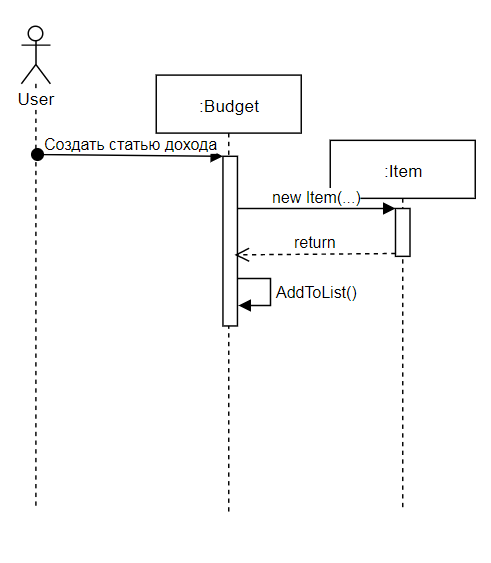


Рисунок 2 – Диаграмма последовательностей для варианта использования «Создать статью дохода».

Для варианта использования «Создать статью дохода» происходящие действия можно описать следующим образом:

* пользователь обращается к классу Budget, указывая некоторые параметры статьи;
* класс Budget инициирует экземпляр класса Item;
* класс Budget добавляет созданный Item в свой список статей.

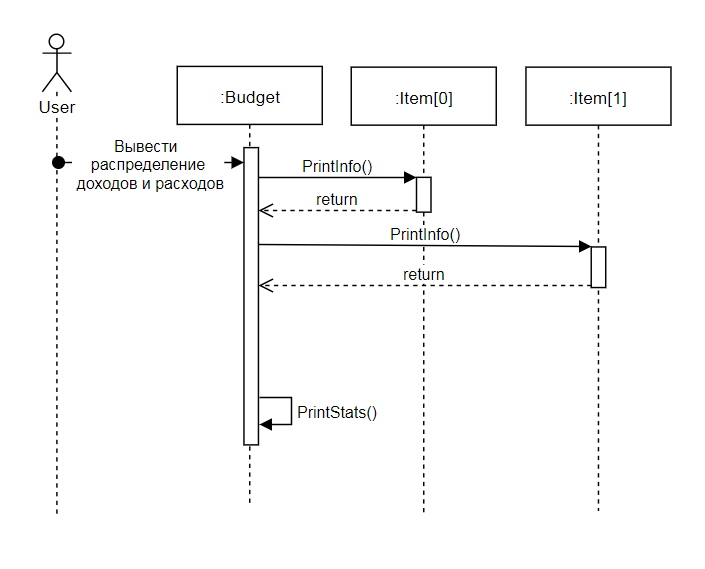


Рисунок 3 – Диаграмма варианта использования «Вывести распределение доходов и расходов».

Для варианта использования «Вывести распределение доходов и расходов» процесс будет выглядеть так:

* пользователь обращается к классу Budget, запрашивая вывод данных;
* класс Budget обращается по очереди ко всем доступным экземплярам класса Item;
* класс Budget выводит собранные данные.
  1. Разработка макета структуры хранимых данных

Специфика выполняемой работы предполагает взаимодействие программы с обширными массивами хранимой информации, для чего потребуется реализация базы данных. На рисунке 4 показана общая структура хранимых модулей.

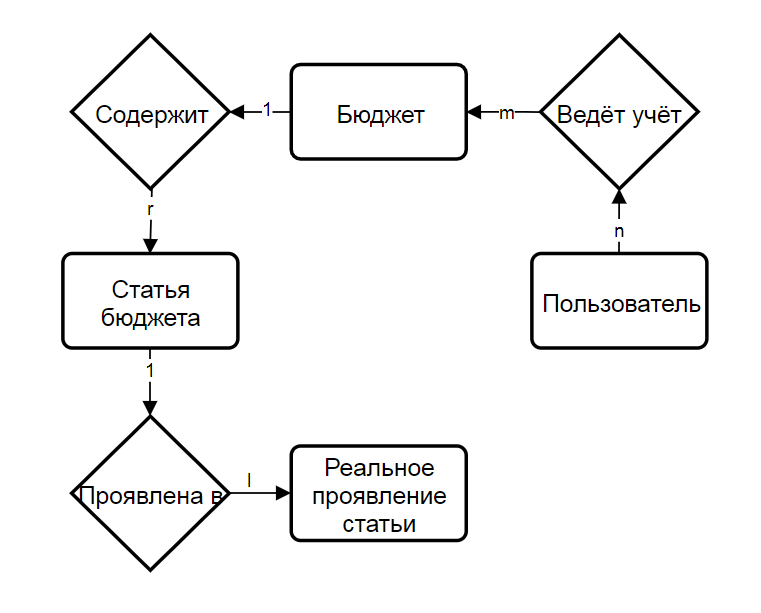


Рисунок 4 – Схема структуры хранимых данных.

Как видно из схемы, хранимые данные представлены в четырёх сущностях:

* «Пользователь» – непосредственный потребитель программного продукта, имеющий логин и пароль в системе бухгалтерии;
* «Бюджет» – смета статей дохода и расхода;
* «Статья бюджета» – обобщённая единица бюджета (как, например, «Продукты», «Отдых», «Коммунальные платежи»);
* «Реальное проявление статьи» – то, на что деньги действительно были потрачены (к примеру, «Осенние туфли», «Мешок картошки»).

Рассматривая связи между перечисленными сущностями, заметим, что схема предполагает, что «Реальному проявлению статьи» может соответствовать одна и только одна «Статья бюджета», в то время как «Статья бюджета» может проявляться в сколько угодно «Реальных проявлений», но иметь только один «Бюджет». Ему, в свою очередь, может соответствовать произвольное число «Статей» и некоторое количество «Пользователей», которые в это время могут вести учёт любого числа «Бюджетов».

* 1. Макет интерфейса

Для более чёткого определения вида конечного продукта спроектируем макет интерфейса приложения. На рисунках 5-9 показаны схематически графические отображения программы.

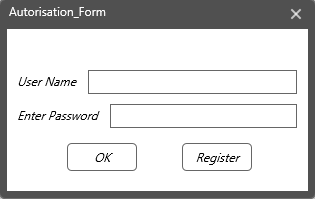


Рисунок 5 – Форма авторизации (форма #1).

Форма авторизации – первое, что видит пользователь, при начале работы с приложением. Ему предлагается ввести имя пользователя и пароль. При нажатии кнопки «OK» происходит проверка введённых данных, если данные корректны, происходит переход к основной форме (форма #3, рисунок 7), если же нет, отображается сообщение об этом. При нажатии кнопки «Register» открывается регистрационная форма (форма #2, рисунок 6), где по похожей логике происходит регистрация нового пользователя.

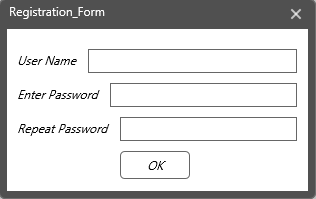


Рисунок 6 – Форма регистрации (форма #2).

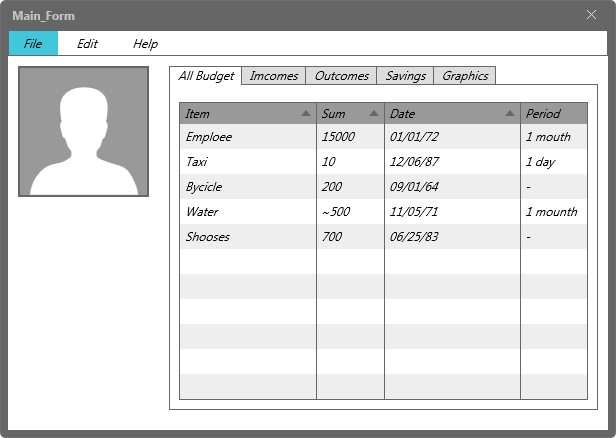


Рисунок 7 – Основная форма (форма #3).

Основная форма отображает все введённые данные о бюджете в данный период в табличном представлении. Также через основную форму происходит вся работа хранимыми данными, и реализуются возможности анализа бюджета. При необходимости добавить статью дохода вызывается форма создания статьи (форма#4, рисунок 8). При выборе на вкладке «Graphics» типа графика и периода отображения, составленный график выводится на форме отображения (форма #5, рисунок 9).

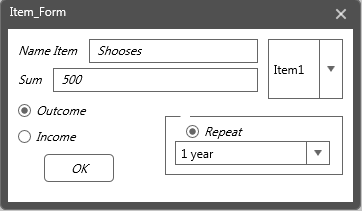


Рисунок 8 – Форма создания статьи (форма #4)

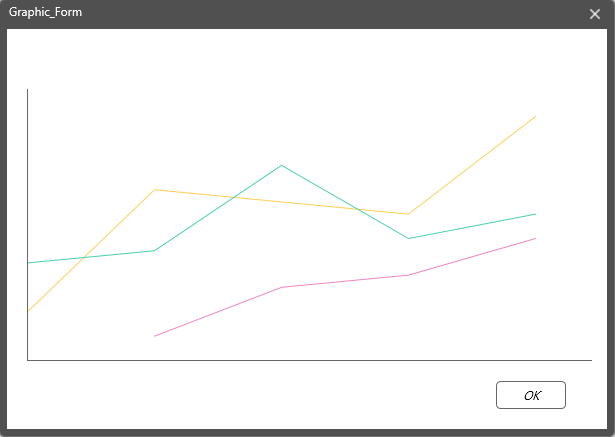


Рисунок 9 – Форма отображения (форма #5)

Таким образом, мы перечислили все основные элементы графического отображения, которые потребуются пользователю для эффективной работы с разрабатываемым приложением, и можем чётко представить себе процесс взаимодействия с ядром, а также вид пользовательской стороны реализации описанных ранее возможностей.

* 1. Календарный план

Для большей эффективности работы, попробуем составить план занятости в проекте, распределив временные издержки по данному периоду, и проанализировать занятость в разработке приложения.

В качестве инструмента отображения календарного распределения работ будем использовать диаграмму Ганта.

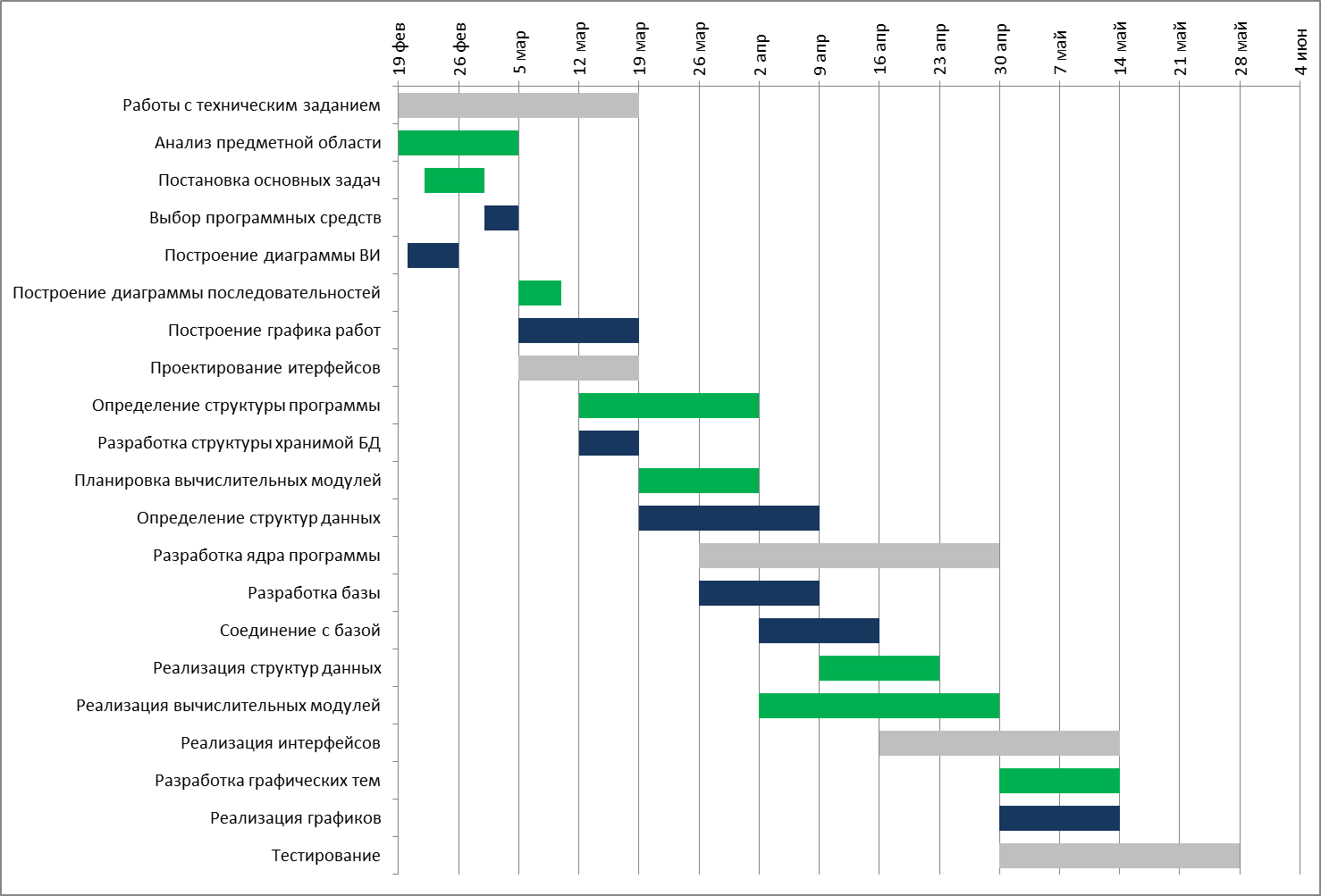


Рисунок 10 - Календарное планирование хода работ.

На диаграмме можно увидеть календарное распределение усилий по разработке. Ход работы разделён на этапы:

1. Работа с техническим заданием – этап включает в себя различную аналитику предметной области, установку основных требований и определение общей структуры разрабатываемого продукта.
2. Анализ предметной области – изучение предметной области.
3. Постановка основных задач – формулировка целей и задач будущего проекта.
4. Выбор программных средств – выбор ОС (Windows[]), базы данных(MySql[]), фреймворк (.Net v.4.5[]).
5. Построение диаграммы ВИ-концептуальное представление системы в процессе её проектирования и разработки.
6. Построение диаграммы последовательностей-моделирование взаимодействия объектов системы во времени, а так же обмена сообщения между ними.
7. Построение графика работ – моделирование диаграммы Ганта.
8. Проектирование интерфейса – проектирование пользовательского интерфейса для взаимодействия пользователя с проектом.
9. Определение структуры – раздел, подразумевающий работы по конкретному определению структур данных внутри программы.
10. Разработка структуры БД – логическое проектирование БД с учётом взаимосвязей между всеми элементами базы данных.
11. Планировка вычислительных модулей – формирование структуры считающего ядра программы.
12. Определение структур данных – выбор формата организации и хранения данных.
13. Разработка ядра программы – этап, на котором происходит реализация считающих механизмов, хранимых структур и взаимодействий между ними.
14. Разработка базы данных – физическое проектирование с учетом особенностей хранения данных, методов доступа, взаимосвязей.
15. Соединение с БД-подключение базы данных к проекту.
16. Реализация структур данных – непосредственное кодирование определённых ранее структур.
17. Реализация вычислительных модулей – реализация частей программы, производящих расчёт.
18. Реализация интерфейса – пункт, включающий в себя реализацию графических возможностей приложения.
19. Разработка графических тем – проектирование пользовательского интерфейса с учётом возможностей программы.
20. Реализация графиков – реализация возможности анализа данных при помощи построения графиков.
21. Тестирование – непосредственно проведение тестов над конечным продуктом.