**Профессиональное образовательное учреждение**

**«Челябинский юридический колледж»**

Предметно- цикловая комиссия

КУРСОВОЙ ПРОЕКТ

по основной образовательной программе подготовки специалистов

по специальности 09.02.03 – программирование в компьютерных системах

**Подсистема банковского расчета кредитов**

|  |  |  |
| --- | --- | --- |
| Студент гр. ПО-1-15  отделение права  и  информационных технологий | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | А.А. Гончаров |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2019 г. |  |
| Руководитель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Е.А. Невинная |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2019 г. |  |
| Нормоконтролер | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Н.Н. Лебедева |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2019 г. |  |

**Челябинск 2019**

**Оглавление**

Введение................................................................................................................. 3

Глава 1 Методы и средства разработки.................................................................... 5

1.1 Определение актуальности...................................................................... 13   
 1.2 Анализ методов разработки..................................................................... 13   
 1.2.1 Язык программирования Ruby................................,,,................ 13   
 1.2.2 Язык программирования Python................................................ 15   
 1.2.3 Язык программирования Java..................................................... 16

1.2.4 Язык программирования C#....................................................... 19  
 1.3 Анализ средств разработки...................................................................... 23   
Глава 2 Проектирование и разработка программного продукта.......................... 26   
 2.1 Описание логической структуры программы........................................ 26   
 2.2 разработка интерактивного интерфейса Программы............................ 28

2.3 Описание используемых процедур и функций...................................... 29

2.4руководство пользователя......................................................................... 33

2.4.1Область применения.................................................................... 33

2.4.2Уровень подготовки пользователя.............................................. 33

2.4.3Подготовка к работе..................................................................... 33  
 2.4.4 ход работы программы................................................................ 34

Заключение................................................................................................................ 38

Источники литературы............................................................................................. 39  
Приложения............................................................................................................... 40

Приложение A........................................................................................................... 40  
Приложение B........................................................................................................... 41

Приложение C........................................................................................................... 43

Приложение D........................................................................................................... 46

**Введение**

Цель данной работы создать Программное обеспечение решающая проблему расчета кредитов с выплатой по месяцам

В связи с поставленными задачами необходимо решить определенные задачи:

1. Удобный и простой для понимания пользователем интерфейс
2. Отказоустойчивость программы от различных факторов
3. Добавление информативных окон
4. возможность вычислять оплату по кредиту в месяц Аннуитетным методом
5. вычислять месячную оплату по кредиту Дифференцированным методом
6. добавить парсинг данных с сайта о курсе доллара в информативный виджет

Актуальность работы объясняется тем, что автоматизации для предоставления кредитных займов имеет большое значение при оформление кредита очень важно понимать сколько будет пере-плата и оплата в месяц по займу.

В настоящее время всё больше организаций стремятся автоматизировать данный процесс расчета данный процесс важен не только организации но и пользователям услуг кредитной организации но и людям связанными с деятельностью в сфере экономики.

С развитием прогресса появляется востребованность в более быстром доступе к информации позволяющая снизить затраты и усилия человека, также автоматизация позволяет зная сумму , период и процентную ставку займа для данного приложения важен простой и понятный пользовательский интерфейс.

О актуальности данной проблемы говорит тот факт что определённое количество времени затрачивается сотрудником для раз счета размера платежей поэтому все больше актуальной становится автоматизация целью данной работы является создание простой и удобной для использования персоналом занимающимися расчетом потребительских кредитов и потребителям кредитной организации.

Разработка велась на языке C# работает по принципу «всякая сущность.есть объект». Его причисляют к объектно-ориентированным, а точнее объектным, языкам программирования. «Язык основан на строгой компонентной архитектуре и реализует передовые механизмы обеспечения безопасности кода» –так принято характеризовать его.   
C# это объективно ориентированный язык программирования, данный язык был разработан 1998-2001 группой инженеров из компании Microsoft данной группой руководили такие люди как Андерс Хейлсберг и Скотта Вильтаумота.Он относится к семье языков с С-подобным синтаксисом.Изначально данный язык разрабатывался для веб-разработки.С#  это простой, современный, типобезопасный объектно-ориентированный язык программирования. C# базируется на семействе языков программирования Cи будет хорошо знаком программистам, работавшим с языками C, C++ и Java. Это касается, прежде всего, системы типов C#, которая отражает [BCL](https://ru.wikipedia.org/wiki/Base_Class_Library). Присутствие или отсутствие тех или иных выразительных особенностей языка диктуется тем, может ли конкретная языковая особенность быть транслирована в соответствующие конструкции CLR.

# Глава 1 Методы и средства разработки

Существуют разные методы разработки программного обеспечения: я расскажу вам о трёх из них

Первый метод это- Спиральная модель, это модель является довольно таки затратной так, что явно не подходит для небольших проектов так же большое внимание в этой модели уделяется управлению рисками спиральная модель состоит из четырех главных повторяющихся стадий. В ходе процесса разработки проект несколько раз проходит через все эти фазы:

**1. Определение требований.** С этой стадии начинается работа над проектом. формулируются цели проекта, основные требования На последующих итерациях требования формируются согласно отзывам, полученным от заказчика. Именно поэтому постоянная коммуникация между заказчиком и командой крайне важна.  
 **2. Анализ и Проектирование** является одной из самых значимых стадий разработки. это возможные события и состояния проекта, препятствующие достижению поставленных целей. Существует довольно обширный диапазон возможных рисков, от тривиальных и легко преодолимых, до крайне серьезных. Главной задачей является выявление всех возможных рисков и присвоение им определенного уровня приоритета на основе их значимости. Следующим шагом является разработка возможных стратегий преодоления этих рисков. В итоге этих действий возможны изменения в последующих стадиях разработки. В качестве результата работы на этом этапе создается прототип.  
 **3. Реализация и интеграция.** На этом этапе происходит разработка и последующее тестирование продукта. Во время первой итерации, когда общие требования еще не так четко сформулированы, разрабатывается так называемый концепция будущего продукта, которая необходима для получения отзыва заказчика. На последующих витках спирали рабочие версии продукта, или **билды**, отправляются заказчику. Это позволяет получить более детальный отзыв и четче сформулировать требования.  
 **4. Планирование следующей фазы.** На этом этапе вся полученная информация используется для планирования дальнейших этапов разработки и повтаряется первая фаза.

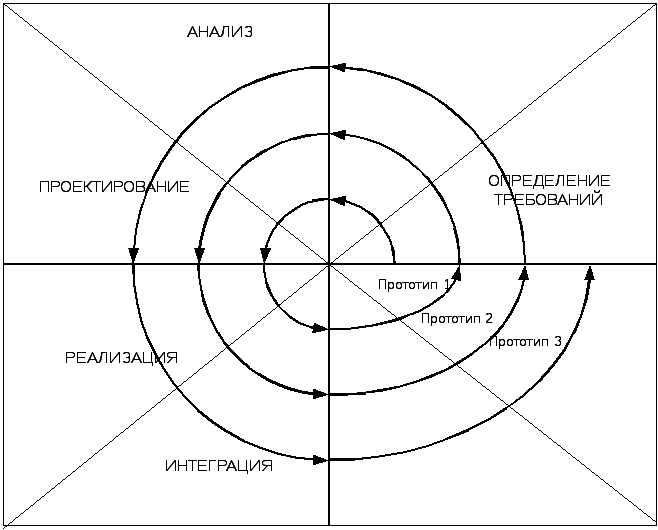


Рис 1. Спиральная модель

Плюсы данной модели:

На ранних стадиях разработки продукта есть рабочая версия для демонстрации,

деление проекта на части,

строгий контроль над документацией;

на поздних этапах разработки.

Минусы метода: неизвестна окончательная датасдачи проекта, на раннихэтапах разработки, большой объём документации, модель является затратной

Второй метод это Каскадная модель так же известная как модель (Водопад) в данной модели процесс разработки подразумевает

Рис 2. Каскадная модель

последовательное прохождение от одного этапа разработки к следующему. Каскадные технологические подходы к разработке информационных систем задают некоторую последовательность выполнения процессов, обычно изображаемую в виде каскада. Эти подходы также иногда называют подходами на основе модели водопада.

1. Анализ требований

На этом этапе важно задокументировать все требования к будущему программному обеспечению. Необходимо посвятить достаточно времени обсуждению деталей проекта со всеми заинтересованными сторонами. Все поступающие данные нужно проанализировать и систематизировать. Важно также учесть все технические ограничения, которые могут возникнуть на стороне заказчика. Итогом данного этапа должно стать создание подробной спецификации, отвечающей всем требованиям заказчика.

2. Проектирование программного обеспечения

Следующим этапом жизненного цикла ПО является создание документа, описывающего масштабы и границы проекта. Данный документ включает в себя мокапы или скетчи интерфейса будущего приложения, а также подробную спецификацию требований программного обеспечения. Необходимо отметить, что в некоторых случаях документ видения (образа) проекта и документ о масштабах и границах проекта могут быть представлены как единый документ “Об образе и границах проекта”.

3. Разработка программного обеспечения

Необходимо отметить, что [разработка программного обеспечения](https://xbsoftware.ru/uslugi/#web-razrabotka) может также включать в себя создание интерактивного прототипа, который, в сущности, является основой будущего приложения. Такой прототип помогает определить архитектуру системы в целом. На данном этапе пишется мало кода: например, код кнопок и простых форм, чтобы дать заказчику общее представление о том, как будет работать конечный продукт. Поэтому мы включили создание прототипа в этап разработки программного обеспечения.

Как только интерактивный прототип и дизайн приложения готов и утвержден заказчиком, начинается разработка стандартов приложения (конвенции наименований, способа документирования кода, инструкций для конечного пользователя и т.д.). После этого можно смело переходить к следующему этапу жизненного цикла, а именно, к разработке программного обеспечения. Разработка ПО может быть разделена на небольшие части, или юниты, и каждый юнит разрабатывается и тестируется разработчиками для проверки его функциональности (модульное тестирование).

4. Тестирование программного обеспечения

После завершения этапа разработки продукт должен пройти тщательное [тестирование](https://xbsoftware.ru/uslugi/#kontrol-kachestva), чтобы убедиться в том, что он соответствует поставленным требованиям. На этапе приемочного тестирования необходимо, чтобы заказчик попытался применить продукт локально точно таким же образом, как он собирается использовать его после релиза. Когда будут исправлены основные ошибки, программное обеспечение можно внедрять. Для исправления незначительных ошибок может использоваться простая система отслеживания, что позволит исправлять любые недоработки уже на этапе сопровождения ПО.

5. Техническая поддержка программного обеспечения

После того, как продукт был протестирован и развернут на сервере заказчика, начинается следующая фаза жизненного цикла разработки программного обеспечения, которая называется сопровождением или технической поддержкой ПО. В целом, сопровождение подразумевает под собой исправление мелких багов, которые обнаруживаются на этом этапе.  
Тем не менее, вполне возможно, что вам придется вносить некоторые изменения в созданное программное обеспечение, несмотря на все усилия, приложенные вами на предыдущих этапах. Заказчик может решить внести изменения в функциональность разработанного продукта. Следовательно, вам придется собирать, описывать и обсуждать новые требования с заказчиком, чтобы внести в продукт необходимые изменения. В данном случае, вам предстоит работа с новым каскадным проектом, и все вышеописанные шаги придется повторять с начала.

Плюсы этого метода заключаются в том, что: устранение проблем последовательно, более ясен заказчикам, удобен в сложных проектах, в течении жизненного цикла работает стабильно.

Минусы этого метода в его: негибкости, сложно составляются требования, непрост в финансовом плане, требований в начале жизненного цикла сложно составляются. Она также может применяться при разработке небольших и несложных проектов. Однако, если на одном из начальных этапов будет допущена ошибка, существует вероятность того, что она будет обнаружена лишь на этапе разработке или тестирования. Поэтому рекомендуется применять данную модель только в том случае, если все требования предельно понятны и не будут меняться с течением времени.

## Третей являетсяинкрементная модель полные требования к системе делятся на различные сборки. Часто используется для описания поэтапной сборки ПО. Имеют место несколько циклов разработки, и вместе они составляют жизненный цикл «мульти-водопад». Цикл разделен на более мелкие легко создаваемые модули. Каждый модуль проходит через фазы определения требований, проектирования, кодирования, внедрения и тестирования. Процедура разработки по инкрементной модели предполагает выпуск на первом большом этапе продукта в базовой функциональности, а затем уже последовательное добавление новых функций, так называемых «инкрементов». Процесс продолжается до тех пор, пока не будет создана полная система. она похожа н спиральную модель

#### https://habrastorage.org/files/8b7/4e2/a86/8b74e2a8640d4ad59e7ac82b991be970.jpg

Рис. 3 Инкрементная модель

Преимущества инкрементной модели:

1. на момент создания определенного инкремента требования стабилизируются (посредством включения в процесс пользователей)

2. не требуется заранее тратить средства, необходимые для разработки всего проекта (поскольку сначала выполняется разработка и реализация основной функции или функции из группы высокого риска);

3. в результате выполнения каждого инкремента получается функциональный продукт;

4. заказчик располагает возможностью высказаться по поводу каждой разработанной версии системы;

5. существует возможность поддерживать постоянный прогресс в ходе выполнения проекта;

6. снижаются затраты на первоначальную поставку программного продукта;

7. риск распределяется на несколько меньших по размеру инкрементов (не сосредоточен в одном большом проекте разработки);

8. в конце каждой инкрементной поставки существует возможность пересмотреть риски, связанные с затратами и соблюдением установленного графика;

9. потребности клиента лучше поддаются управлению, поскольку время разработки каждого инкремента очень незначительно;

10. поскольку переход из настоящего в будущее не происходит моментально, заказчик может привыкать к новой технологии постепенно.

Недостатки инкрементной модели:

1. в модели не предусмотрены итерации в рамках каждого инкремента;

2. определение полной функциональности системы должно осуществляться в начале жизненного цикла, чтобы обеспечить определение инкрементов;

3. формальный критический анализ и проверку намного труднее выполнить для инкрементов, чем для системы в целом;

4. заказчик должен осознавать, что общие затраты на выполнение проекта не будут снижены;

5. поскольку создание некоторых модулей будет завершено значительно раньше других, возникает необходимость в четко определенных интерфейсах;

6. для модели необходимо хорошее планирование и проектирование;

7. может возникнуть тенденция к оттягиванию решений трудных проблем на будущее с целью продемонстрировать руководству успех, достигнутый на ранних этапах разработки.

1.1 Определение актуальности

Актуальность данного приложения разъясняется теми фактами что пользователи банков и кредитных организаций при выборе услуг преследующие цель найти самую выгодную услугу в этом помогают интеграции на их web-сайт данного сервиса или полноценного приложения по расчету банковских займов. Теперь банки не представляют свое существование без подобных программных продуктов. также существуют проблемы с тем что для того чтоб пользователь услуг банка решился пользоватся услугами кридетавания видел примущество которое показывает программа а именно переплату по кредиту и сколько будет требоватся оплата в месяц по кредиту. А данная программа решает данную проблему

1.2. Анализ методов разработки

Прежде чем приступать к работе нам нужно определиться с языком программирования для нашего продукта.

Я рассматривал свои любимые языки программирования а именно: Ruby, Python, Java и C# в итоге я решил остановить свой выбор на C# так как данный язык более удобен для данной цели тем что можно использовать графический интерфейс WindowsForms который в свою очередь интегрирован в VisualStudio.

остановимся на языках поподробней

1.2.1 Язык программирования Ruby

Ruby – это тщательно сбалансированный язык. Его создатель Юкихиро Мацумото (так же известный как “Matz”)

Рис 5. Логотип Ruby

Объединил части его любимых языков (Perl, Smalltalk, Eiffel, Ada и Lisp) чтобы сформировать новый язык, в котором парадигма функционального программирования сбалансирована принципами императивного программирования.

Ruby — динамический, рефлективный, интерпретируемый высокоуровневый язык программирования. Язык обладает независимой от операционной системы реализацией многопоточности, сильной динамической типизацией, сборщиком мусора и многими другими возможностями.

Особенности языка: Имеет лаконичный и простой синтаксис, частично разработанный под влиянием [Ада](https://ru.wikipedia.org/wiki/Ада_(язык_программирования)), [Eiffel](https://ru.wikipedia.org/wiki/Eiffel) и [Python](https://ru.wikipedia.org/wiki/Python).

Позволяет обрабатывать [исключения](https://ru.wikipedia.org/wiki/Обработка_исключений) в стиле [Java](https://ru.wikipedia.org/wiki/Java) и [Python](https://ru.wikipedia.org/wiki/Python).

Позволяет [переопределять операторы](https://ru.wikipedia.org/wiki/Перегрузка_операторов), которые на самом деле являются [методами](https://ru.wikipedia.org/wiki/Метод_(языки_программирования)).

### Рост популярности Ruby

Со времени выпуска публичной версии в 1995 году, Ruby привлек внимание программистов со всего мира. В 2006 году Ruby завоевал массовое признание. В крупнейших городах по всему миру активно действуют группы пользователей Ruby, а конференции, посвященные Ruby, заполнены до предела.

Ruby-Talk, основная [рассылка](https://www.ruby-lang.org/ru/community/mailing-lists/) для обсуждения языка Ruby, содержала в среднем 200 сообщений ежедневно в 2006 году. В последние годы это количество уменьшилось, так как сообщество стало использовать несколько тематических рассылок вместо одной общей.

Индекс [TIOBE](http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html), который измеряет рост популярности языков программирования, помещает Ruby на 9 место среди общепризнанных языков программирования. Во многом, рост популярности языка приписывается популярности программного обеспечения, написанного на Ruby, в особенности – [Ruby on Rails](http://rubyonrails.org/), среде разработки веб-приложений.

Ruby также [абсолютно открыт](https://www.ruby-lang.org/en/about/license.txt). Открыт для бесплатного использования, изменений, копирования и распространения.

1.2.2 Язык программирования Python

Python — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций

Рис 6. Логотип Python

Разработка языка Python была начата в конце [1980-х годов](https://ru.wikipedia.org/wiki/1980-е) сотрудником голландского института CWI [Гвидо ван Россумом](https://ru.wikipedia.org/wiki/Гвидо_ван_Россум). Для распределённой ОС [Amoeba](https://ru.wikipedia.org/wiki/Amoeba_(операционная_система)) требовался расширяемый [скриптовый язык](https://ru.wikipedia.org/wiki/Сценарный_язык), и Гвидо начал писать Python на досуге, позаимствовав некоторые наработки для языка [ABC](https://ru.wikipedia.org/wiki/ABC_(язык_программирования)) (Гвидо участвовал в разработке этого языка, ориентированного на обучение программированию). В феврале [1991 года](https://ru.wikipedia.org/wiki/1991_год) Гвидо опубликовал исходный текст в [группе новостей](https://ru.wikipedia.org/wiki/Группа_новостей) alt.sources. С самого начала Python проектировался как [объектно-ориентированный язык](https://ru.wikipedia.org/wiki/Объектно-ориентированный_язык_программирования).Появившись сравнительно поздно, Python создавался под влиянием множества языков программирования:

ABC, Modyla-3, C, [Smalltalk](https://ru.wikipedia.org/wiki/Smalltalk), [Lisp](https://ru.wikipedia.org/wiki/Lisp), [Fortran](https://ru.wikipedia.org/wiki/Fortran),[Miranda](https://ru.wikipedia.org/wiki/Миранда_(язык_программирования)), [Java](https://ru.wikipedia.org/wiki/Java). Несмотря на то, что Python обладает достаточно самобытным синтаксисом, одним из принципов дизайна этого языка является принцип наименьшего удивления.

#### Стандартная библиотека

Богатая стандартная библиотека является одной из привлекательных сторон Python. Здесь имеются средства для работы со многими сетевыми протоколами и форматами Интернета. Существуют модули для работы с регулярными выражениями, текстовыми кодировками, мультимедийными форматами, криптографическими протоколами, архивами. Помимо стандартной библиотеки существует множество библиотек, предоставляющих интерфейс ко всем системным вызовам на разных платформах.  
Для Python принята спецификация программного интерфейса к базам данных DB-API 2 и разработаны соответствующие этой спецификации пакеты для доступа к различным СУБД: Oracle, MySQL, PostgreSQL, Sybase, Firebird (Interbase), Informix, Microsoft SQL Server и SQLite.  
Библиотека NumPy для работы с многомерными массивами позволяет достичь производительности научных расчётов, сравнимой со специализированными пакетами. SciPy использует NumPy и предоставляет доступ к обширному спектру математических алгоритмов. Numarray специально разработан для операций с большими объёмами научных данных.  
Python предоставляет простой и удобный программный интерфейс Си API для написания собственных модулей на языках Си и C++. Такой инструмент как SWIG позволяет почти автоматически получать привязки для использования C/C++ библиотек в коде на Python. Инструмент стандартной библиотеки ctypes позволяет программам Python напрямую обращаться к динамическим библиотекам, написанным на Си. Существуют модули, позволяющие встраивать код на С/C++ прямо в исходные файлы Python, создавая расширения «на лету».  
Python и подавляющее большинство библиотек к нему бесплатны и поставляются в исходных кодах. Более того, в отличие от многих открытых систем, лицензия никак не ограничивает использование Python в коммерческих разработках и не налагает никаких обязательств кроме указания авторских прав.

**1.2.3****Язык программирования Java**

Изначально язык назывался Oak («Дуб»), разрабатывался [Джеймсом Гослингом](https://ru.wikipedia.org/wiki/%D0%93%D0%BE%D1%81%D0%BB%D0%B8%D0%BD%D0%B3,_%D0%94%D0%B6%D0%B5%D0%B9%D0%BC%D1%81) для программирования бытовых электронных устройств. Из-за того, что язык с таким названием уже существовал, вскоре Oak был переименован в Java. Назван в честь марки [кофе](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D1%84%D0%B5) Java, которая, в свою очередь, получила наименование одноимённого острова ([Ява](https://ru.wikipedia.org/wiki/%D0%AF%D0%B2%D0%B0)), поэтому на официальной эмблеме языка изображена чашка с горячим кофе. Существует и другая версия происхождения названия языка, связанная с [аллюзией](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%BB%D1%8E%D0%B7%D0%B8%D1%8F) на кофе-машину как пример бытового устройства, для программирования которого изначально язык создавался. В соответствии с этимологией в русскоязычной литературе с конца двадцатого и до первых лет двадцать первого века название языка нередко переводилось как Ява, а не транскрибировалось, как это стало общепринятым позднее

Программы на Java [транслируются](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D1%81%D0%BB%D1%8F%D1%82%D0%BE%D1%80) в [байт-код Java](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B9%D1%82-%D0%BA%D0%BE%D0%B4_Java), выполняемый [виртуальной машиной Java](https://ru.wikipedia.org/wiki/Java_Virtual_Machine) (JVM) — программой, обрабатывающей байтовый код и передающей инструкции оборудованию как [интерпретатор](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D1%80%D0%B5%D1%82%D0%B0%D1%82%D0%BE%D1%80).

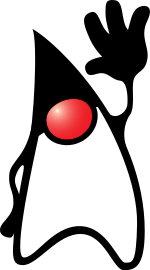
[](https://commons.wikimedia.org/wiki/File:Duke_(Java_mascot)_waving.svg?uselang=ru)

Рис 7. Дюк, талисман Java

Достоинством подобного способа выполнения программ является полная независимость байт-кода от [операционной системы](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0) и [оборудования](https://ru.wikipedia.org/wiki/%D0%90%D0%BF%D0%BF%D0%B0%D1%80%D0%B0%D1%82%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B0), что позволяет выполнять Java-приложения на любом устройстве, для которого существует соответствующая виртуальная машина. Другой важной особенностью технологии Java является гибкая система безопасности, в рамках которой исполнение программы полностью контролируется виртуальной машиной. Любые операции, которые превышают установленные полномочия программы (например, попытка несанкционированного доступа к данным или соединения с другим компьютером), вызывают немедленное прерывание.

Часто к недостаткам концепции виртуальной машины относят снижение производительности. Ряд усовершенствований несколько увеличил скорость выполнения программ на Java:

* применение технологии трансляции байт-кода в машинный код непосредственно во время работы программы ([JIT](https://ru.wikipedia.org/wiki/JIT)-технология) с возможностью сохранения версий класса в машинном коде,
* широкое использование [платформенно-ориентированного кода](https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) (native-код) в стандартных библиотеках,
* аппаратные средства, обеспечивающие ускоренную обработку байт-кода (например, технология [Jazelle](https://ru.wikipedia.org/wiki/Jazelle), поддерживаемая некоторыми процессорами [архитектуры ARM](https://ru.wikipedia.org/wiki/ARM_(%D0%B0%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0))).

По данным сайта shootout.alioth.debian.org, для семи разных задач время выполнения на Java составляет в среднем в полтора-два раза больше, чем для C/C++, в некоторых случаях Java быстрее, а в отдельных случаях в 7 раз медленнее. С другой стороны, для большинства из них потребление памяти Java-машиной было в 10—30 раз больше, чем программой на C/C++. Также примечательно исследование, проведённое компанией [Google](https://ru.wikipedia.org/wiki/Google_(%D0%BA%D0%BE%D0%BC%D0%BF%D0%B0%D0%BD%D0%B8%D1%8F)), согласно которому отмечается существенно более низкая производительность и бо́льшее потребление памяти в тестовых примерах на Java в сравнении с аналогичными программами на [C++](https://ru.wikipedia.org/wiki/C%2B%2B).

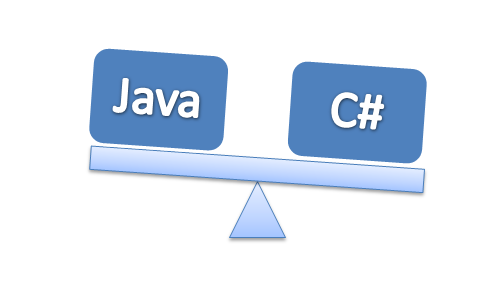
Идеи, заложенные в концепцию и различные реализации среды виртуальной машины Java, вдохновили множество энтузиастов на расширение перечня языков, которые могли бы быть использованы для создания программ, исполняемых на виртуальной машине. Эти идеи нашли также выражение в спецификации общеязыковой инфраструктуры [CLI](https://ru.wikipedia.org/wiki/Common_Language_Infrastructure), заложенной в основу платформы [.NET](https://ru.wikipedia.org/wiki/.NET_Framework) компанией [Microsoft](https://ru.wikipedia.org/wiki/Microsoft).

1.2.4 Язык программирования С#

Так, с развитием CLR от версии 1.1 к 2.0 значительно обогатился и сам C#; подобного взаимодействия следует ожидать и в дальнейшем (однако, эта закономерность была нарушена с выходом C# 3.0, представляющего собой расширения языка, не опирающиеся на расширения платформы .NET). CLR предоставляет C#, как и всем другим [.NET](https://ru.wikipedia.org/wiki/.NET_Framework)-ориентированным языкам, многие возможности, которых лишены «классические» языки программирования.

Рис 8. Логотип C#

Например, [сборка мусора](https://ru.wikipedia.org/wiki/Сборка_мусора) не реализована в самом C#, а производится CLR для программ, написанных на C# точно так же, как это делается для программ на [VB.NET](https://ru.wikipedia.org/wiki/Visual_Basic_.NET), [J#](https://ru.wikipedia.org/wiki/Visual_J♯) и др.Общей чертой каждой версии, начиная с С# 1.0, является то, что разработчики C# предусматривали и учитывали практические потребности профессиональных программистов, дополняя С# синтаксисом, с которым в языке более естественным становится выражение смысла, предполагаемого разработчиками. В отличие от характерной для динамического языка совместимости со снижением безопасности типов данных в C#, новая возможность фактически делает существующие небезопасные части программы более поддерживаемыми и простыми в обращении. Следовательно, такой код, вероятнее всего, будет корректным, даже при отсутствии проверки компилятором безопасности типов. его синтаксис взял в себя такие языки как:Java, C++, VisualBasic. И около. C# разрабатывался как язык программирования прикладного уровня для [CLR](https://ru.wikipedia.org/wiki/Common_Language_Runtime) и, как таковой, зависит, прежде всего, от возможностей самой CLR.В настоящее время широкое распространение получили компьютеры с многоядерными процессорами, и поэтому возможность распараллеливать выполнение кода среди всех доступных процессоров приобретает все большее значение практически для всех, кто программирует на C#. Microsoft выпустила инструментарий для разработки приложений – платформу .NET. Она также объединяла «под одной крышей» несколько языков программирования, что было в новинку для того времени.  
Еще одним новшеством платформы .NET была технология активных серверных страниц ASP.NET (Active Server Page). С её помощью можно было относительно быстро разработать веб-приложения, взаимодействующие с базами данных.  
Язык программирования C# был создан специально для ASP.NET. На C# полностью была написана и сама ASP.NET.

Как и Java, C# изначально предназначался для веб-разработки, и примерно 75% его синтаксических возможностей аналогичны языку программирования Java. Его также называют «очищенной версией Java». 10% позаимствовано из C++, а 5% – из Visual Basic. И около 10% C# – это реализация собственных идей разработчиков.  
  
  
Рис. 9 сравнение   
Несмотря на весьма существенные различия между компонентной объектной моделью COM (основного стандарта Microsoft для компонентного проектирования и реализации программного обеспечения) и моделью Java, языки программирования имеет довольно много общего.   
  
Единая среда выполнения программ основана на использовании промежуточного языка IL (Intermediate Language — промежуточный язык), исполняющего почти ту же роль, что и байт-код виртуальной машины Java. Используемые в рамках технологии .NET компиляторы с различных языков транслируют программы в IL-код. Так же, как и байт-код Java, IL-код представляет собой команды гипотетической стековой вычислительной машины. Но есть и разница в устройстве и использовании IL.  
  
Во-первых, в отличие от JVM, IL не привязан к одному языку программирования. В составе, предварительных версий Microsoft.NET имеются компиляторы языков Си++, Си#, Visual Basic. Независимые разработчики могут добавлять другие языки, создавая свои компиляторы в IL-код.  
  
Во-вторых, IL предназначен не для программной интерпретации, а для последующей компиляции в машинный код. Это позволяет достичь существенно большего быстродействия программ. Содержащие IL-код файлы несут достаточно информации для работы оптимизирующего компилятора.

1.3.1VisualStudio

Microsoft Visual Studio .NET - это интегрированная среда разработки (Integrated Development Environment (IDE)) для создания, документирования, запуска и отладки программ, написанных на языках .NET. Это мощный инструмент профессиональной разработки сложных приложений, один из лучших во всем мире. Набор его функций чрезвычайно велик, сейчас мы рассмотрим его основные элементы. Другие пункты меню и управляющие окна Обычно среда разработки включает в себя:

* текстовый редактор
* компилятор и/или интерпретатор
* средства автоматизации сборки
* отладчик.

Иногда содержит также средства для интеграции с системами управления версиями и разнообразные инструменты для упрощения конструирования графического интерфейса пользователя. Многие современные среды разработки также включают браузер классов, инспектор объектов и диаграмму иерархии классов — для использования при объектно-ориентированной разработке ПО. Хотя, и существуют среды разработки, предназначенные для нескольких языков программирования — такие, как Eclipse, NetBeans, Embarcadero RAD Studio, Qt Creator или Microsoft Visual Studio, обычно среда разработки предназначается для одного определённого языка программирования - как, например, Visual Basic, Delphi, Dev-C++. Ниже перечислены основные преимущества IDE-среды Visual Studio.

Встроенный Web-сервер. Для обслуживания Web-приложения ASP.NET необходим Web-сервер, который будет ожидать Web-запросы и обрабатывать соответствующие страницы. Наличие в Visual Studio интегрированного Web-сервера позволяет запускать Web-сайт прямо из среды проектирования, а также повышает безопасность, исключая вероятность получения доступа к тестовомуWeb-сайту с какого-нибудь внешнего компьютера, поскольку тестовый сервер может принимать соединения только с локального компьютера.

Поддержка множества языков при разработке. Visual Studio позволяет писать код на своем языке или любых других предпочитаемых языках, используя все время один и тот же интерфейс (IDE). Более того, Visual Studio также еще позволяет создавать Web-страницы на разных языках, но помещать их все в одно и то же Web-приложение. Единственным ограничением является то, что в каждой Web-странице можно использовать только какой-то один язык (очевидно, что в противном случае проблем при компиляции было бы просто не избежать).

1.3 Сравнительный анализ

Сравнив разные средства разработки, я выбрал VisualStudio так как

Среда интерактивной разработки (IDE) Visual Studio получила наиболее значимый успех  более 10 лет назад, когда де-факто стала продвигаться корпорацией Microsoft для своей основной операционной системы Windows. В то время, благодаря размеру финансирования и количеству штата она быстро начала обгонять несколько других IDE, популярных в то время. Что бы вы понимали, разработка такого крупного проекта в 2000-ых требовала реально больших усилий. Ну а в 2012 году, точно стала самой популярной для разработчиков на C++. При этом, платформа .Net Framework (которая примерно к тому времени достигла определенных промежуточных этапов развития) очень сильно повлияла на рост популярности Visual Studio, поскольку других IDE для языков .Net (VB, C#, F#) не существовало.

Популярность Visual Studio во многом связана с ростом популярности технологий из ряда DirectX и .COM (у программистов C++) и росту продаж Windows. Дело в том, что компилятор C++ в этой среде разработки (MSVC) был максимально адаптирован под постройку готовых коммерческих или свободных приложений, а для сопровождения требовал лишь установки готового redistributable-пакета Visual C++. При этом компилятор MSVC был нестабилен, с ним было очень много проблем. Но все равно, создавать крупные связанные между собой проекты в одном решении и настраивать их было намного быстрее и проще. К тому же, вместе со средой разработки, Microsoft устанавливала на ваш компьютер огромную справочную библиотеку, локализованную под разные языки, среди прочего, по Windows API и т.д. Именно поэтому я выбрал VisualStudia

Интуитивный стиль кодирования. По умолчанию Visual Studio форматирует код по мере его ввода, автоматически вставляя необходимые отступы и применяя цветовое кодирование для выделения элементов типа комментариев. Такие незначительные отличия делают код более удобным для чтения и менее подверженным ошибкам. Применяемые Visual Studio автоматически параметры форматирования можно даже настраивать, что очень удобно в случаях, когда разработчик предпочитает другой стиль размещения скобок (например, стиль K&R, при котором открывающая скобка размещается на той же строке, что и объявление, которому она предшествует).

Более высокая скорость разработки. Многие из функциональных возможностей Visual Studio направлены на то, чтобы помогать разработчику делатьсвою работу как можно быстрее. Удобные функции, вроде функции IntelliSense (которая умеет перехватывать ошибки и предлагать правильные варианты), функции поиска и замены (которая позволяет отыскивать ключевые слова как в одном файле, так и во всем проекте) и функции автоматического добавления и удаления комментариев (которая может временно скрывать блоки кода), позволяют разработчику работать быстро и эффективно.

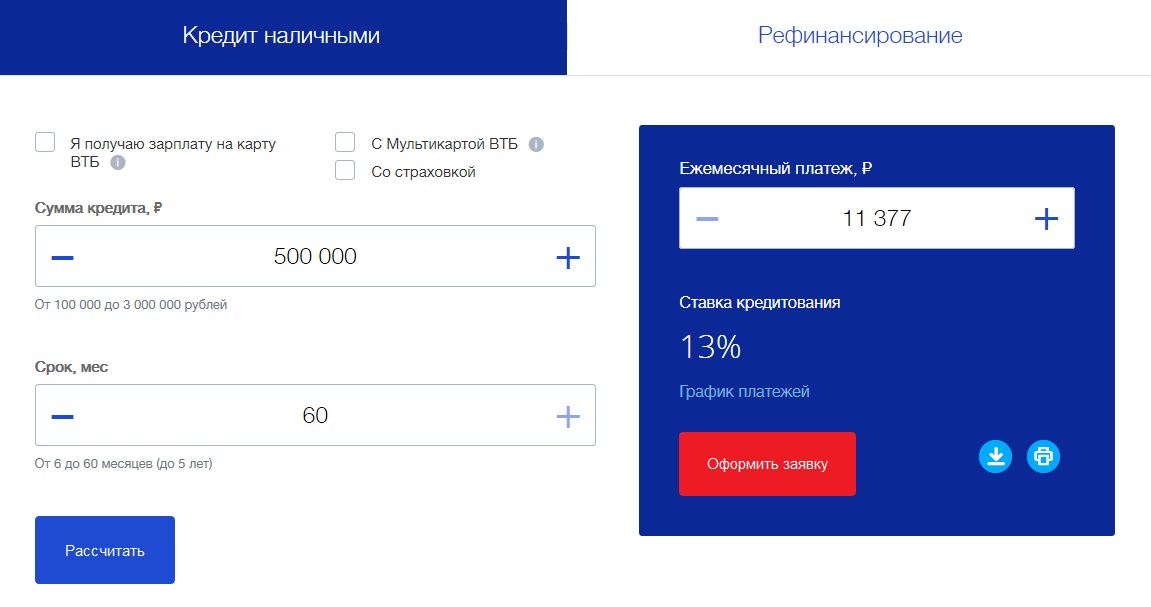
Далее проведем сравнение с аналогичными продуктами многие подобные приложения к примеру приложение банка втб аналогичный расчет под условие конкретного банка 

Рис. 11 интерфейс программы Втб

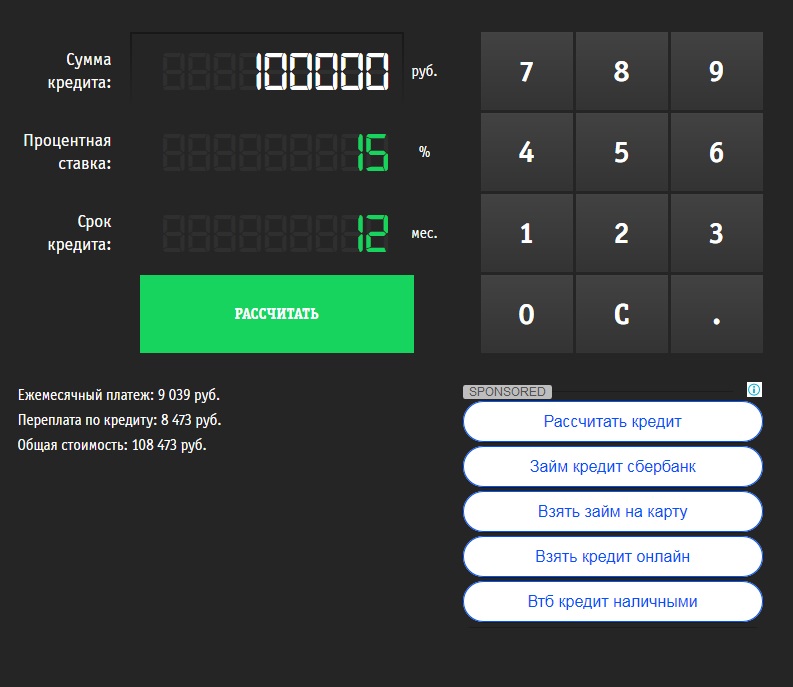
или подобное приложения от сбербанка даст Вам полный обзор того, как будет обстоять ситуация ежемесячных платежей по предполагаемому кредиту. Просчитайте с помощью данного калькулятора наиболее привлекательное и выгодное для Вас кредитное предложение. 

Рис. 12 Интерфейс программы Сбербанка

Аналоги хороши быстрым доступом к расчетам но главный их минус в том что они ориентированы на определённый банк но в какомто случае это компенсируется тем что иза размещения на сайте банка часть пользователей это их пользователи банка . А проэкт курсовой работы является более уневерсальным из за того что алгоритм расчета не диктуеца определённым банком этим самым можно белиемение уневерсально расчитать оплату по займу также можно отслеживать курс доллара этим самым программа более информативна также в отличии от браузерных аналогов есть возможность распечатать результаты расчетов а с распечаткой уже можно пойти в банк и просто больше доступ к результатом так как ими можно воспользоватся вне устройства также можно отдельно выбрать аннуитетный метод и диференцированыйе методы расчета а изза этого повышается вариативность выбора пользователя Аннуитетные. Являются самым удобным и распространённым способом погашения кредитов среди заемщиков. Ежемесячно необходимо выплачивать фиксированную сумму до конца кредитного договора, Дифференцированные. Способ погашения кредита, предполагающий постепенное уменьшение суммы ежемесячных платежей.  При этом нельзя забывать, что все цифры будут примерными. Точную сумму переплаты, ежемесячных платежей можно узнать только в банке. Все расчеты будут произведены специалистом во время заключения договора.

Правильно пользоваться данной программой учиться не нужно. Это довольно простой и удобный способ провести требуемые расчеты и получить точные данные по интересующим вас аспектам

**Глава 2 Проектирование и разработка программного продукта**

* 1. **Описание логической структуры программы**

aga-lone является специализированной программой, которая работает так же, как и обычный, привычный каждому человеку, калькулятор. Однако, основное отличие заключается в том, что в aga-lone вписывается специальный алгоритм, который, в свою очередь, дает возможность производить определенные расчеты на основе введенной клиентом банка информации а также позволяет сравнить с другими валютами Алгоритм данного приложения работает следующим способом: есть главная форма меню из которой пользователь определяет какой тип расчета ему требуется далее вводит данные в форму и получает нужный ему результат (стандартно выводится в рублях) также есть возможность посмотреть курс доллара на сегодня реализованный методом сбора данных с сайта в онлайн режиме. Важно решить, какой именно тип платежа, возможен ли пересчет и по какой схеме будет возврат средств. На этом основывается выбор Аннуитетного или Дифференцированного метода.

Первый рассматриваемый метод Дифференцированный:

заключается в том, что на первые месяцы выплат приходятся максимальные суммы, в которые входит часть основного долга и проценты по кредиту. При дифференцированных платежах сумма основного долга, так называемое тело долга, делится равными частями на весь срок платежа, а вот проценты ежемесячно начисляются на остаток долга. Соответственно, в первый месяц суммы платежей наиболее велики, потому что проценты по кредиту существенны. А к концу срока выплаты будут минимальны. Дифференцированные платежи удобны для тех, у кого доход не носит характер неизменной величины, и через некоторое время может появиться возможность досрочно погасить долг. В этом случае переплата по кредиту будет меньше, чем при аннуитетном расчёте. Формула данного расчета такова:

od = sk / kp

np = ok \* (ps / 12)

ok = sk - (od \* kp)

m = od + (sk - (od \* 0)) \* 0,1 / 12

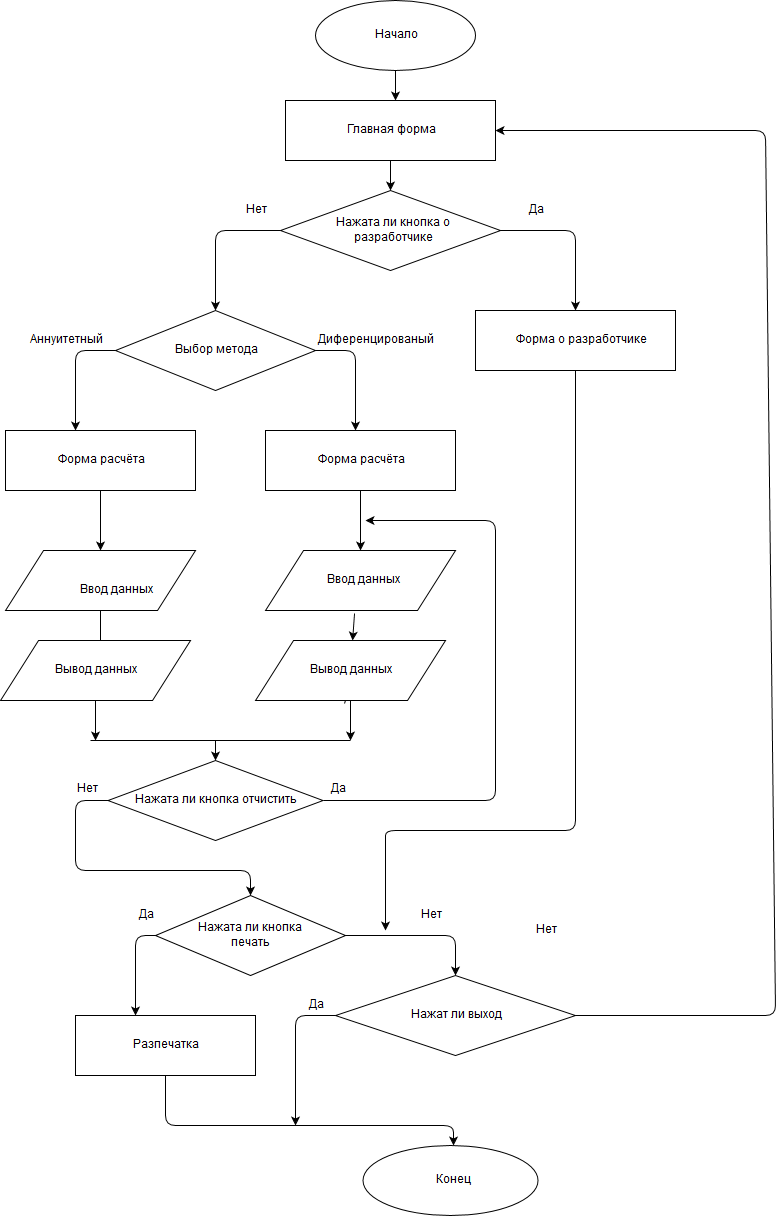
1. od — возврат основного долга; sk — первоначальная сумма кредита; kp — количество периодов; np — начисленные проценты; ok — остаток кредита в данном месяце; ps — годовая процентная ставка. m — плата за месяц

Переплатаопределяется путем сложения всех платежных месяцев Далеко не каждый кредитополучатель может позволить себе выплачивать в первые месяцы суммы, складывающиеся при дифференцированном расчёте платежей. Ему придётся отказаться от подобного кредита, либо взять сумму меньшую, чем необходимо, чтобы беспроблемно расплачиваться по долговым обязательствам. Такому заёмщику больше подойдёт аннуитетный вид платежа.

Второй Метод Аннуитетный сумма ежемесячного взноса всегда неизменна, но вот структура этой суммы меняется из месяца в месяц. Основную часть в первые месяцы составляют проценты по кредиту, а сумма тела долга — минимальна. Таким образом банк страхует риски недополучения прибыли в случае досрочного погашения кредита заёмщиком. Подобный график погашения платежей с ежемесячной суммой — константа очень выгоден людям, имеющим фиксированный доход: нет необходимости каждый месяц сверяться с графиком платежей, чтобы заранее зарезервировать нужную сумму для оплаты кредита; равные доли платежа позволяют исключить возможность остаться без средств к существованию после уплаты ежемесячного взноса. Формула данного расчета такова:

Mпс = Гпс / 100 / 12

Мп = СЗ \* ( Мпс / ( 1- ( 1 + Мпс )^-СК ) )

Гпс = годовая процентная ставка; Мп – месячный платеж по займу; Сз – общее количество средств, взятых взаймы; Мпс – размер месячной процентной ставки; Ск – срок займа (количество месяцев) когда будут начисляться проценты по нему. переплата определяется умножением полученой суммы на количество периодов и отнять сумму займа окончательная сумма переплаты по дифференцированному виду платежа несколько ниже, чем при аннуитетном расчёте. Если сумма кредита существенна, то разница будет более чем очевидна. Рис 10. Блок схема

Таким образом, прежде чем заключить договор по кредитованию, необходимо взвесить все плюсы и минусы обоих видов платежей: дифференцированного и аннуитетного. Блок схема Программы :

**2.2 разработка интерактивного интерфейса Программы**

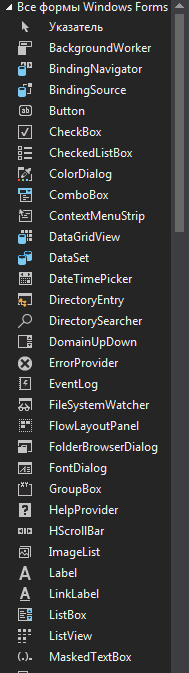
при его создание была использована среда разработки VisualStudia а именно ее функция WindowForm элементы интерфейса взяты из панели элементов скриншоты полученного результата находятся в приложение.

Рис. 11 Панель элементов

**2.3 Описание используемых процедур и функций**

Начнем с того что переход по формам реализован в событие клика по кнопке в ней описано то что мы создаем новую форму и отображаем ее:

private void button1\_Click(object sender, EventArgs e)

{

full frm = new full();

frm.Show();

}

закрытие формы прописанная в событие клика по кнопке выглядит так :

private void button4\_Click(object sender, EventArgs e)

{

Close();

}

для того чтоб отчистить введенные и полученные данные использовалась данная команда в событие клика по кнопке:

private void button1\_Click(object sender, EventArgs e)

{

summ.Text = "";

srok.Text = "";

stav.Text = "";

txtResult.Text = "";

}

далее переходим к одному из самых сложнейших функций программы это парсинг данных с сайта скурсам доллара реализовано это при помощи библиотек xNet, Fizzer и HtmlAgilityPack.

создана функция parses которая в последующем вызываема при старте формы

в самой функцие создаётся Http запрос на определённый вэб-сайт с последующим сохранением исходного кода сайта сохраняемый в фаил st.html в кодировке UTF-8 далее он считывается и в нем находится нужная нам строка по так называемой маске и результат сохраняется в переменную которая выводится на label.

void parses()

{

var dan = new HttpRequest();

string response = dan.Get("https://www.banki.ru/products/currency/cash/ekaterinburg/").ToString();

HtmlAgilityPack.HtmlDocument hap = new HtmlAgilityPack.HtmlDocument();

hap.LoadHtml(response);

File.WriteAllText("st.html", response, Encoding.UTF8);

string usd = hap.DocumentNode.QuerySelector("div.currency-table\_\_large-text").InnerHtml;

label16.Text = usd;

}

далее рассмотрим как реализован расчет по формуле платежа методом Деференцированого расчета в начале обвялены переменные с проверкай на введенные данные далее создан массив в который перебирается в циклекаждый платежный период это 1 итерация цикла который в котором рассчитывается по формуле и выводится результат каждой итерации цикла

int n =0;

double a = 0.1 ;

int S = Convert.ToInt32(summ.Text);

int N = Convert.ToInt32(srok.Text);

int pp = Convert.ToInt32(stav.Text);

if (N <= 0)

{

label4.Text = "Введено некоректный период займа";

}

else

{

double b = S / N;

double sn = S - (b \* n);

double p = sn \* pp / 12;

double[] res = new double[N];

for (int i = 0; i < N; i++)

{

res[i] = b + (S - (b \* i)) \* a / 12;

int mes = 1 + i;

txtResult.Text += mes + " месяцоплата = " + res[i] +"\n";

}

}

}

и последняя из функций это разпечатка результата создано 3 события разпечатка страницы документа, диалоговое окно и клик по кнопке в 1 событие описано то что мы отрисовываем полученую переменную после расчета в строку с определённым шрифтом и позицией а в 3 событие идет сравнение если результат диалогового окна работает то мы выводим документ на печать.

private void printDocument1\_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)

{

e.Graphics.DrawString(txtResult.Text, new Font("Times New Roman", 14, FontStyle.Bold), Brushes.Black, new PointF(100, 100));

}

private void button2\_Click(object sender, EventArgs e)

{

if (printPreviewDialog1.ShowDialog() == DialogResult.OK)

{

printDocument1.Print();

}

}

**2.4руководство пользователя**

**2.4.1Область применения**

предназначена для автоматизации процесса расчета оплаты по кредиту с возможностью дальнейшей печати полученных результатов и демонстрации курса доллара на сегодня используется в банковской сфере но также может использоваться и частными лицами

**2.4.2Уровень подготовки пользователя**

Пользователь aga-loan должен иметь опыт работы с ОС MS Windows (95/98/NT/2000/XP), навык работы с ПО

данное приложение преднозначено для занятых в кредитных и банковских организациях автоматизируещая функцию расчета оплаты по кредитам обеспечивается применение средства автоматизации в соответствии с назначением а именно в ОС Windows также иметь доступ к сети интернет и подключение принтера

**2.4.3Подготовка к работе**

Для работы с aga-loan необходимо следующее программное обеспечение: Internet Explorer (входит в состав операционной системы Windows) или любой другой браузери установленное приложение aga-loan

Перед началом работы с aga-loan на рабочем месте пользователя необходимо выполнить следующие действия:

Необходимо запустить фаил aga-loan.ехе

После чего запуститься установка на Ваш компьютер. Выбираем кнопку Далее затем выбераем путь нажимаем далее и затем Установить.

**2.4.4 ход работы программы**

При запуске программы нас встречает главное меню где на выбор представлены 4 сценария событий на выбор:

1 разузнать о разработчике

2 Расчет ежемесячного платежа деференцированным методом

3 Расчет ежемесячного платежа Аннуететным методом

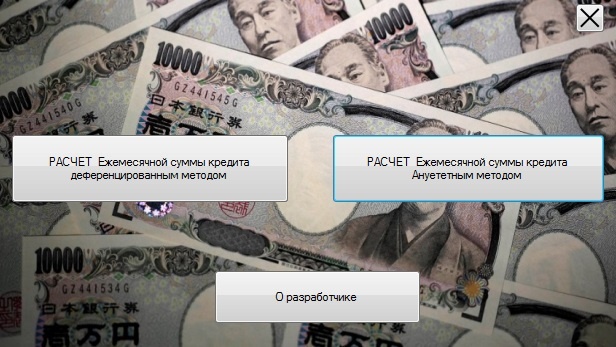
4 выход из программы

Рис. 12 Главное меню

при выборе 1 произойдет открытие новой формы в которой находится информация о разработчике где написана информация о разработчике



Рис. 13 О разработчике

при выборе 2 произойдет открытие новой формы в которой присутствует форма ввода данных пользователя, виджет с курсам доллара на сегодня, кнопки: отчистить, рассчитать, выход и печать а также форма вывода результата для Расчет ежемесячного платежа деференцированным методом при нажатие на кнопку отчистить все введенные данные пользователя и вывод вычислений отчищаются при нажатие на кнопку рассчитать произойдет расчет на основе введенных данных пользователем а результат расчета будет выведен в поле вывода , при нажатие на кнопку печать будет открыто окно предварительного просмотра печать и отправка файла на печать, при нажатие на кнопку выход происходит закрытие окна.

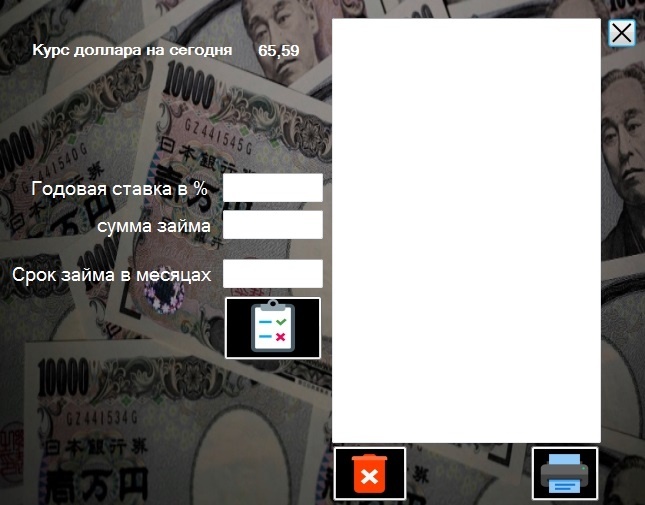


Рис. 14 Аннуитетный метод

при выборе 3 произойдет открытие новой формы в которой присутствует форма ввода данных пользователя, виджет с курсам доллара на сегодня, кнопки: отчистить, рассчитать, выход и печать а также форма вывода результата для Расчет ежемесячного платежа Аннуететным методом при нажатие на кнопку отчистить все введенные данные пользователя и вывод вычислений отчищаются при нажатие на кнопку рассчитать произойдет расчет на основе введенных данных пользователем а результат расчета будет выведен в поле вывода , при нажатие на кнопку печать будет открыто окно предварительного просмотра печать и отправка файла на печать, при нажатие на кнопку выход происходит закрытие окна.

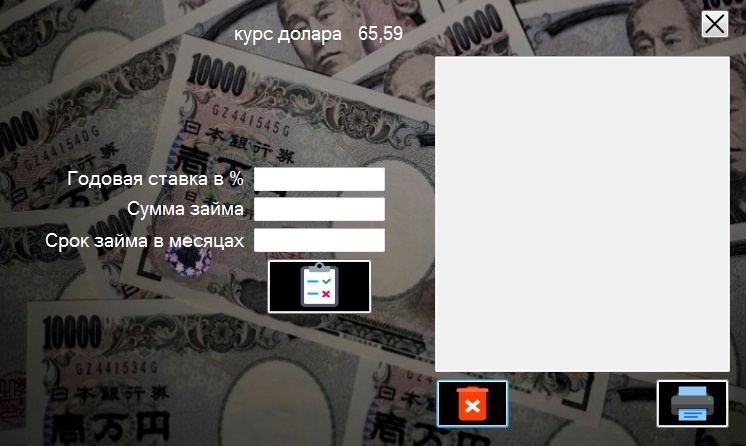


Рис. 15 Диференцированный расчет

**Заключение**

Цели данной работы достигнуты создав Программное обеспечение решающая проблему расчета кредитов с выплатой по месяцам

1. В связи с этим решены поставленные задачи:

2. Удобный и простой для понимания пользователем интерфейс

3. Отказоустойчивость программы от различных факторов

4. Добавление информативных окон

5. возможность вычислять оплату по кредиту в месяц Ануитетным методом

6. вычислять месячную оплату по кредиту Дифференцированным методом

7. добавить парсинг данных с сайта о курсе доллара в информативный виджет

Этим самым автоматизировав расчет кредитов и получен полезный виджет указывающий на курс доллара изучив литературу и информационные ресурсы определил модель разработки из трех рассмотренных остановился на спиральной рассмотрев плюсы и минусы языков программирования и определившись остановился на C# для данного продукта также определились с средствами разработки под определённую задачу подходит хорошо VisualStudio далее была продумана логическая состовляющая продукта после чего создан интерфейс при помощи стандартных возможностях WindowsForm далее написаны логические процедуры такие как: расчет, парсер(считывание информации с вэб сайта) и распечатка документа после этого написано рукаводство пользователя

**Электронные Источники**

1.http://znatokdeneg.ru/uslugi-bankov/kredity/raschet-annuitetnyh-platezhej-po-kreditu-formula-primer.html

2.https://www.banki.ru/wikibank/raschet\_annuitetnogo\_plateja/

3.https://www.banki.ru/wikibank/raschet\_differentsirovannogo\_plateja/

4.https://habr.com/ru/post/111674/

5.https://habr.com/ru/post/313694/

7.https://www.java.com/en/

8.https://www.ruby-lang.org/ru/

9.https://www.python.org/

10.https://visualstudio.microsoft.com/ru/?rr=https%3A%2F%2Fyandex.ru%2F

Книги

# 1. Изучаем C#, 3-е издание. Эндрю Стиллмен, Дженнифер Грин 2014г.

2. Боннер, Е. А. Банковское кредитование / Е.А. Боннер. - М.: Городец, **2017**. - 160 c.

3. Professional Visual Studio 2017 Bruce Johnson 2017г.

### 4. C# Programming Yellow Book Rob Milles 2015г.

# 5. Fundamentals of Computer Programming with C# Svetlin Nakov 2013г.

# 6. C# 7.0. Карманный справочник. Скорая помощь для программистов на C# 7.0 [Албахари Дж., Албахари Б.](https://www.chitai-gorod.ru/books/authors/albakhari_dzh_albakhari_b/) 2017г.

7. Крюков, Р. В. Банковское дело и кредитование / Р.В. Крюков. - М.: А-Приор, **2016**. - 236 c.

**Приложение**

**Приложение А главная форма**

usingSystem;

usingSystem.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace banc\_soft

{

public partial class index : Form

{

public index()

{

InitializeComponent();

}

private void button2\_Click(object sender, EventArgs e)

{

pomes frm = new pomes();

frm.Show();

}

private void button1\_Click(object sender, EventArgs e)

{

full frm = new full();

frm.Show();

}

private void button4\_Click(object sender, EventArgs e)

{

Close();

}

private void index\_Load(object sender, EventArgs e)

{

}

private void button3\_Click(object sender, EventArgs e)

{

o\_programere frm = new o\_programere();

frm.Show();

}

}

}

**Приложение B форма Ануететный метод**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using xNet;

using Fizzler;

using Fizzler.Systems.HtmlAgilityPack;

using HtmlAgilityPack;

using System.Text.RegularExpressions;

using System.IO;

using System.Diagnostics;

namespace banc\_soft

{

public partial class full : Form

{

public full()

{

InitializeComponent();

}

private void button3\_Click(object sender, EventArgs e)

{

double Mps, Mp, Pp ;

double Gps = Convert.ToInt32(stavv.Text);

double Sz = Convert.ToInt32(zaim.Text);

double Sk = Convert.ToInt32(priod.Text);

Mps = Gps / 100 / 12;

Mp = Sz \* ( Mps / ( 1 - (Math.Pow( 1 + Mps , -Sk ) ) ) ) ;

Pp = (Mp \* Sk) - Sz;

txtValue.Text = "\b " + " Оплатавмесяц = " + (Convert.ToString(Mp)) + " " + "\n" + "\b" + "Переплатасоставляет = " + (Convert.ToString(Pp));

}

private void button1\_Click(object sender, EventArgs e)

{

stavv.Text = "";

zaim.Text = "";

priod.Text = "";

txtValue.Text = "";

}

void parses()

{

var dan = new HttpRequest();

string response = dan.Get("https://www.banki.ru/products/currency/cash/ekaterinburg/").ToString();

HtmlAgilityPack.HtmlDocument hap = new HtmlAgilityPack.HtmlDocument();

hap.LoadHtml(response);

File.WriteAllText("st.html", response, Encoding.UTF8);

string usd = hap.DocumentNode.QuerySelector("div.currency-table\_\_large-text").InnerHtml;

label4.Text = usd;

}

private void full\_Load(object sender, EventArgs e)

{

parses();

}

private void printDocument1\_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)

{

e.Graphics.DrawString(txtValue.Text, new Font("Times New Roman",14, FontStyle.Bold), Brushes.Black, new PointF(100, 100));

}

private void button5\_Click(object sender, EventArgs e)

{

if(printPreviewDialog1.ShowDialog() == DialogResult.OK)

{

printDocument1.Print();

}

}

private void button4\_Click(object sender, EventArgs e)

{

Close();

}

}

}

**Приложение C Форма расчета деференцированным методом**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using xNet;

using Fizzler;

using Fizzler.Systems.HtmlAgilityPack;

using HtmlAgilityPack;

using System.Text.RegularExpressions;

using System.IO;

using System.Diagnostics;

namespace banc\_soft

{

public partial class pomes : Form

{

public pomes()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

summ.Text = "";

srok.Text = "";

stav.Text = "";

txtResult.Text = "";

}

void parses()

{

var dan = new HttpRequest();

string response = dan.Get("https://www.banki.ru/products/currency/cash/ekaterinburg/").ToString();

HtmlAgilityPack.HtmlDocument hap = new HtmlAgilityPack.HtmlDocument();

hap.LoadHtml(response);

File.WriteAllText("st.html", response, Encoding.UTF8);

string usd = hap.DocumentNode.QuerySelector("div.currency-table\_\_large-text").InnerHtml;

label16.Text = usd;

}

private void button3\_Click(object sender, EventArgs e)

{

int n =0;

double a = 0.1 ;

int S = Convert.ToInt32(summ.Text);

int N = Convert.ToInt32(srok.Text);

int pp = Convert.ToInt32(stav.Text);

if (N <= 0)

{

label4.Text = "Введено некоректный период займа";

}

else

{

double b = S / N;

double sn = S - (b \* n);

double p = sn \* pp / 12;

double[] res = new double[N];

for (int i = 0; i < N; i++)

{

res[i] = b + (S - (b \* i)) \* a / 12;

int mes = 1 + i;

txtResult.Text += mes + " месяцоплата = " + res[i] +"\n";

}

}

}

private void button4\_Click(object sender, EventArgs e)

{

Close();

}

private void printDocument1\_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)

{

e.Graphics.DrawString(txtResult.Text, new Font("Times New Roman", 14, FontStyle.Bold), Brushes.Black, new PointF(100, 100));

}

private void button2\_Click(object sender, EventArgs e)

{

if (printPreviewDialog1.ShowDialog() == DialogResult.OK)

{

printDocument1.Print();

}

}

private void label16\_Click(object sender, EventArgs e)

{

}

private void pomes\_Load(object sender, EventArgs e)

{

parses();

}

}

}

**Приложение D скриншоты интерфейса**

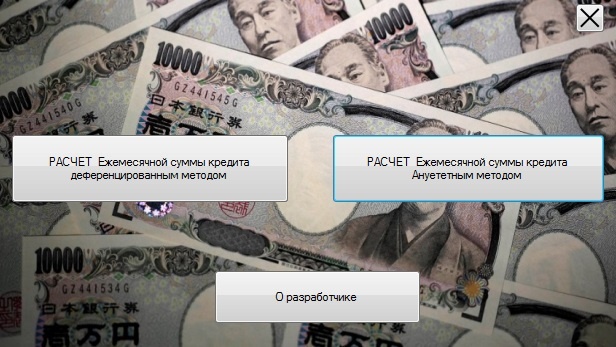


Рис 11. Главное меню

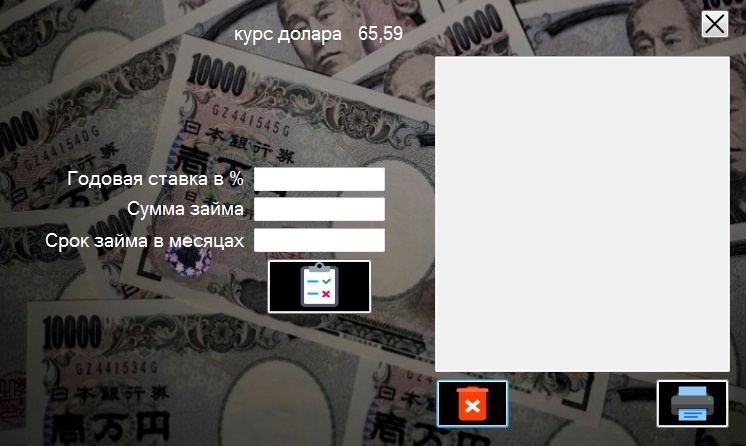


Рис 12. Ануитетный метод расчета

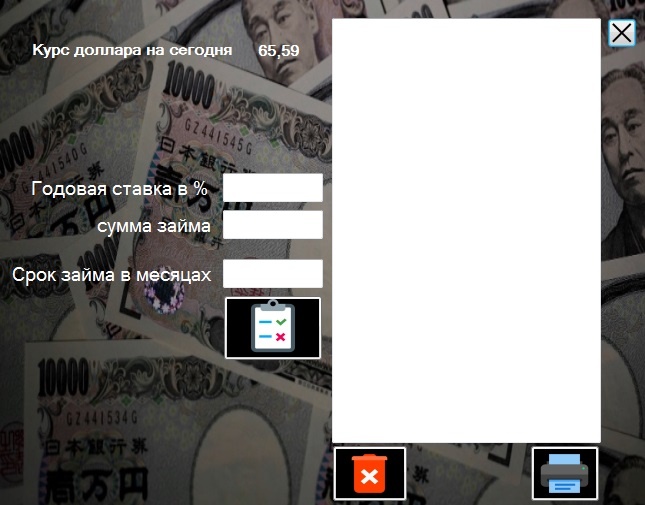


Рис 13. Диференципрованый метод расчета



Рис 14. О разработчике