**Разработка информационной системы банкомата**

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc527907832)

[ГЛАВА 1. ОПИСАНИЕ ОСНОВНЫХ ПОНЯТИЙ ТЕОРИИ ИНФОРМАЦИОННЫХ СИСТЕМ 5](#_Toc527907833)

[1.1. Понятие информационных систем и технологий 5](#_Toc527907834)

[1.2. Типы языков программирования 11](#_Toc527907835)

[1.3. Обзор самых популярных языков программирования для проектирования ИС 15](#_Toc527907836)

[ГЛАВА 2. СОЗДАНИЕ ГРАФИЧЕСКОГО ПРИЛОЖЕНИЯ С ПОМОЩЬЮ С++ 21](#_Toc527907837)

[2.1.Обзор C++ Builder 21](#_Toc527907838)

[2.2. Разработка проекта ИС банкомата 24](#_Toc527907839)

[ЗАКЛЮЧЕНИЕ 30](#_Toc527907840)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 32](#_Toc527907841)

# ВВЕДЕНИЕ

С развитием компьютерной техники начального этапа ее становления появился специальный машинный язык программирования (ЯП), при использовании которого разработчики могли бы применить команды, оперируя с самыми разнообразными ячейками памяти, при этом полностью используя разные возможности машины.

Но постоянное использование совокупности современных персональных компьютеров (ПК) на уровне машинного языка было очень затруднительным по разным причинам.

Поэтому от применения его пришлось со временем отказаться. На современном этапе программисты начали пользоваться целым множеством мощных языков программирования, которые позволяют выполнять самые сложные функции для их реализации.

В дальнейшем, при стремительном развитии высокоуровневых языков программирования, появилась со временем возможность как-то адаптировать процесс обработки информации при использовании программных продуктов, являющихся очень удобными для обычного пользователя, поскольку опытные программисты имеют возможность создавать программы прямо для определенного предметного сектора. Вместе с этим чрезвычайно широкой популярностью ознаменовались программные продукты с графическим интерфейсом пользователя.

Актуальными в нынешнее время является высокая эффективность в применении информационных систем с графическим интерфейсом, которые написаны на современных ЯП высокого уровня.

Целью представляемой работы является разработка информационной системы банкомата.

В соответствии с поставленной целью будут рассмотрены задачи:

– рассмотреть современные источники литературы по теории разработки информационных систем;

* подробно дать характеристику классификации ЯП;
* охарактеризовать среду по разработке ПО C++ Builder;
* создать с помощью C++ Builder оконное приложение для банкомата.

Объектом исследования является теория проектирования ИС.

Предметом исследования является информационная система банкомата.

Этим направлением в программировании занимались следующие разработчики: Б.Стауструп, А.Прата, У.Голдинг и многие другие.

# ГЛАВА 1. ОПИСАНИЕ ОСНОВНЫХ ПОНЯТИЙ ТЕОРИИ ИНФОРМАЦИОННЫХ СИСТЕМ

## 1.1. Понятие информационных систем и технологий

Современные фирмы и компании представляют собой очень сложные организационные системы, где отдельные составляющие (основные или оборотные фонды, материальные и трудовые ресурсы и иные компоненты) постоянно изменяются, а также находятся и в сложном взаимодействии непосредственно друг с другом.

Процесс функционирования предприятий и организаций самого различного типа при условиях рыночной экономики часто ставит новые задачи по усовершенствованию управленческой деятельности на базе комплексной автоматизации управления разными технологическими и производственными процессами, трудовыми ресурсами, персоналом [5].

Рыночная экономика часто приводит к возрастанию общего объема и усложнению заданий, решаемых в направлении организации производства, процессов для планирования и анализа, связей с потребителями и поставщиками продукции, финансовой работы, оперативное управление которыми практически невозможно без выполнения организации современной информационной системы (ИС).

Информационная система может предназначаться для обработки информации, а также поддержки процесса принятия самых разных управленческих решений.

Стоит отметить, что она включает в себя:

– совокупность информации;

– разные принципы применения экономико-математических методов;

– технические средства;

– программное и другое обеспечение.

ИС, используемая в управлении, постоянно должна решать все текущие задачи тактического и стратегического планирования, бухгалтерского учета, оперативного управления компанией.

Многие учетные задачи (материального и бухгалтерского учета, налогового планирования и т. д.) могут решаться без дополнительных затрат с помощью вторичной обработки данных для оперативного управления. Весь учет является необходимым и дополнительным средством для контроля. Применяя оперативную информацию, что получена в ходе функционирования любой автоматизированной информационной системы, каждый руководитель может сбалансировать и спланировать ресурсы фирмы:

– материальные;

– кадровые;

– финансовые.

При этом можно просчитать и оценить разные результаты управленческих решений, полностью наладить оперативное управление общей себестоимостью продукции (услуг, товаров), ходом выполнения плана работ, использованием ресурсов и иные показатели.

Применение ИС позволяет [10]:

– значительно повышать степень обоснованности всех принимаемых решений непосредственно за счет выполнения оперативного сбора, обработки и передачи информации;

– выполнять обеспечение своевременности принятия решений по менеджменту организацией;

– добиваться стремительного роста эффективности управления за счет представления необходимой информации в своевременные строки руководителям практически всех уровней управления с единого информационного фонда компании;

– согласовывать решения, что принимаются на различных уровнях менеджмента и в самых разных структурных подразделениях;

– непосредственно за счет информированности разного уровня персонала о текущей ситуации рассматриваемого экономического объекта обеспечивать стремительный рост производительности труда, а также сокращение непроизводственных потерь.

Самой основной составляющей частью каждой автоматизированной информационной системы считается информационная технология (ИТ), непосредственное развитие которой тесно связывается с развитием и работой ИС.

Главной целью ИТ является получение посредством переработки самых первичных данных информации нового качества, на базе которой вырабатываются самые оптимальные управленческие решения.

Такой эффект достигается за счет:

– непосредственной интеграции информации;

– обеспечения ее актуальности или непротиворечивости;

– применения современных технических средств по внедрению и функционированию качественно новых форм поддержки деятельности для аппарата управления.

Основные функции информационных технологий показаны на рисунке 1:



Рисунок 1 – Функции ИТ

Все информационные технологии справляются с существенным увеличением массивов перерабатываемой информации, а также ведет к сокращению суммарных сроков ее ресурсов в менеджменте.

Многие автоматизированные информационные системы считаются для информационной технологии самой основной средой, элементами которой являются способы и средства для преобразования данных.

Практически все разновидности информационных систем вне зависимости от сферы их применения включают в себя одну и ту же совокупность компонентов (рисунок 2):



Рисунок 2 – Компоненты ИС

То есть, это:[5]

– функциональные компоненты;

– [компоненты для системы обработки информации](http://uchebnikirus.com/informatika/mitni_informatsiyni_tehnologiyi_-_pashko_pv/komponenti_sistemi_opratsyuvannya_danih.htm);

– организационные компоненты.

Функциональными компонентами является система функций управления, или же полный набор (то есть, комплекс) взаимосвязанных во времени или пространстве работ по менеджменту, необходимых для удовлетворения поставленных перед организацией целей.

Любая сложная функция управления делится на ряд намного мелких задач и доводится до непосредственного их исполнителя.

Именно от того факта, как именно будет выполнено задание отдельным работником и зависит успех выполнения всех конечных задач компании в целом.

Стоит отметить, что таким образом, вся совокупность управленческих воздействий может иметь своим конечным результатом выполнение общих задач, которые стоят перед предприятием, а также к каждому конкретному исполнителю независимо от его положения на службе.

Приведенные положения сильно подчеркивают не только так называемый индивидуальный, а и групповой характер функций менеджмента, а деловой (или практический) результат может образоваться не эпизодически, а в постоянном направлении.

Весь процесс по управлению предприятием сводится или к линейному (административному) управлению предприятием или же его структурным подразделением, либо к функциональному (к примеру, материально-техническое обеспечение, бухгалтерский учет и прочее).

Поэтому итоговую декомпозиция ИС по функциональному признаку заключается в выделении с ее отдельных составных частей, которые называют функциональными подсистемами (ФС) (функциональные модули или бизнес-приложения), которые выполняют систему функций управления.

Каждому функциональному признаку обусловлено назначение некоторой подсистемы, то есть то, для какой именно сферы деятельности она предназначается и каковы ее основные цели, задачи или функции она выполняет.

Все функциональные подсистемы существенно могут зависеть от предметной области (или сферы применения) таких информационных систем.[30]

Выбор итогового состава функциональных задач разных функциональных подсистем управления обычно осуществляется с учетом основных уровней (фаз) управления, а именно:

– планирование;

– контроля и анализа;

– учет;

– регулирования.

Планированием называют управленческую функцию, обеспечивающую формирование планов, по которым будет организовываться функционирование объекта управления.

По традиции выделяют:[14]

– летнее (1 год);

– перспективное (5 - 10 лет);

– оперативное (декада, сутки, неделя, месяц) планирование.

Учет, контроль, анализ – это функции, обеспечивающие получение информации о состоянии системы управления за определенный временной промежуток, определения причины и факта отклонения состояния объекта менеджмента от его планируемого поведения или состояния, а также расчет размеров данного отклонения.

Весь учет ведется непосредственно по показателям плана по выбранному диапазону планирования (то есть, оперативный, среднесрочный и другие).

Регулирование управления – это специальная функция, которая обеспечивает сравнение фактических и планируемых показателей функционирования каждого объекта управления и непосредственную реализацию необходимых управляющих средств при наличии разного рода отклонений от запланированных в указанном диапазоне. Согласно выделенных подсистем функционирования и с учетом требований по управлении определяется весь состав задач для функциональных подсистем ИС.

## 1.2. Типы языков программирования

Процессы функционирования ПК заключаются в применении самых разных инструкций какого-то перечня четко определенных команд в указанном порядке.

Машинный вид обрабатываемой информации, который может состоять только с символов 1 и 0, указывает, какое именно действие должен выполнять процессор.

Для задания в ПК перечня действий на выполнение, надо задать какую-то определенную последовательность с двоичного кода. [4]

Программы, изображенные в машинных кодах, состоят с нескольких сотен тысяч команд.

Выполнение качественного их описания является занятием утомительным и очень сложным. Программисты должны помнить всегда все сочетания символом 1 и 0 для всех команд двоичные коды адресов памяти ПК, что используются при выполнении.

Намного проще написать программу на естественном языке, а всю работу по переводу ее в необходимый машинный код поручить только ПК. [11]

Все создатели ЯП рассматривают понятие языка немного по–своему. К одним с самых современных утверждений, признаваемым практически большинством разработчиков, относятся:[15]

– язык программирования предназначается для разработки программ, которые будут применятся непосредственно в процессе передачи ПК некоторых определенных инструкций по разработке вычислительного процесса.

– ЯП при этом отличаются от человеческих языков предназначением, а именно для трансляции данных от пользователя программой к ПК. Можно даже выполнить обобщение этих всех определений:

ЯП – это методология передачи приказов, определенных инструкций, четкого руководства, тогда как обычные ЯП будут предназначены только при выполнении обмена некоторой информацией.

ЯП представляется чаще всего в виде набора инструкций, определяющих основоположные понятия семантики и его синтаксиса. Для многих таких ЯП также применимы международные стандарты. В указанном случае специальные организации проводят систематические обновление спецификаций, определения соответствующего ЯП, продолжают модернизацию.

После создания вычислительных машин все опытные разработчики создали более 2650 ЯП и количество их с каждым годом значительно увеличивается. Некоторыми такими ЯП программируют также небольшое количество разработчиков, а другие становятся очень популярными.

На современном этапе развития ЯП перед опытными программистами поставлена задача для проектирования специальных систем хранения информации, что 30-40 лет назад были практически невозможными. Появляются все новые технологии, требующие разных подходов к выполнению процессов для программирования, а стремительное развитие глобальной сети Интернет также имеет возможность предоставить новейшие методы программирования, которые не освоены до конца опытными разработчиками.

Всё это служит прекрасной почвой для реализации создания новейших ЯП, отвечающих практически всем задачам для нашего времени и также используют новейшие способы, парадигмы программирования, что позволяют решать все современные проблемы разработки ПО с графическим интерфейсом.[6]

Но, не смотря на огромнейшее количество ЯП, лишь самые выбранные с них удостоены признания программистов.

Сравнительный анализ всех современных ЯП по возможностям, способам реализации или сложности освоения является задачей очень актуальной.

Оценивать удобство различных семантических схем лишь можно на каком-то конкретном примере и для всех ЯП можно указывать те задачи, для которых они будут пользоваться лучше, чем для остальными.

Подобные сравнения переходят в самую настоящую «битву» между опытными программистами, в которой каждая с сторон яростно может выполнять оборону «своего» языка программирования, не принимая доводы иной стороны.

Самым первым этапом для разработки языков, стали языки, которые выполняли перевод некоторые символические обозначения непосредственно в машинный код.[13]

ЯП низкого уровня – язык программирования, который предназначается для определенного типа ПК и отражающий его непосредственный код; языки низкого уровня программисты называют часто машинно-ориентированными ЯП.

Стоит отметить, что их сложно как-то сконвертировать для реализации на ПК с разными центральными микропроцессорами, довольно тяжело выполнить их изучение, поскольку требуется знать очень хорошо все принципы схемы ПК.[2]

К ЯП, для так называемого, низкого уровня программирования принадлежат также специальные языки – ассемблеры – это языки машинно-зависимого класса (рисунок 3).



Рисунок 3 – Пример кода Ассемблера

Для каждого с типов процессора существует только свой язык, а о портативности созданных таким образом программ в другую какую-то платформу говорить не приходилось.

Немного позже были спроектированы компьютерные программы для выполнения трансляции самых разных математических выражений, а в 1954 г. был запущен транслятор для Фортрана.

Все команды ЯП высокого уровня являются эквивалентными сразу нескольким командам, написанным в машинных кодах, такие программы, являются намного компактнее.[3]

Машинно-ориентированными языки называются языки, где наборы разных типов средств для написания программ сильно зависят от ЭВМ (или внутреннего машинного языка).

Указанные языки также позволяют применять такие возможности программирования:

* высокое качество написания программ;
* предсказуемость программного кода;
* возможность разработки ПО с помощью ресурсов;
* очень низкая скорость программирования;
* трудоемкость процесса составления программного кода, плохо защищенного от появления ошибок;
* невозможность использования программных продуктов, составленных на таких ЯП, в других ПК.

Машинно-независимые языки – средства описания алгоритмов для реализации задач, подлежащей обработке.

Стоит отметить, что они также очень удобны в использовании для большого круга программистов.

## 1.3. Обзор самых популярных языков программирования для проектирования ИС

В нынешнее время С++ считают одним из популярнейших и основных языков, что могут применяться для разработки ПО.

В последние годы господство С++ слегка пошатнулось, ведь широкое развитие получили Java, Python и C#, но маятник мнения опытных разработчиков уже качнулся в обратную сторону, большинство программистов, что оставили программировать на С++, поспешили сразу возвратится к привычному ЯП. [6]

ЯП С++ – ЯП для общего назначения, что часто применим для написания кода системного программирования, понимаемом при этом в очень широком понимании.

Кроме того, ЯП С++ успешно применяется для написания приложений, выходящих далеко за рамки классического кода. Реализации ЯП С++ также часто присутствует полностью на всех ПК, от самых малофункциональных – и для применения в супер-ПК. [20]

Б. Страуструп является самым первым создателем и разработчиком ЯП С++, а также создателем транслятора. Он является авторитетным и очень опытным сотрудником научного центра корпорации AT&T, что находится в Нью-Джерси.

Страуструп получил почетное звание почетного магистра по вычислительной техники в институте города Арус, а имеющееся у него докторское звание – в Кембридже. [16]

Он специализируется и в сфере операционных систем, разных распределенных ИС, принципов программирования, моделирования, а также является автором руководства С++.

ЯП С++ обязан безусловно языку С [17], который сохраняется в качестве некоторое его подмножество. В нем сохранены все свойственные для С средства и методы программирования низкого уровня, предназначенные для разрешения задач по системного программирования. [6]

Название С++ придумано Р. Маскитти в 1982 г. Оно имело возможность показать свой эволюционный характер для нового ЯП С++. Обозначение «++» определяет операцию инкремента.

Изначально ЯП С++ был спроектирован также для того, чтоб Б. Страуструпу и его команде не надо было выполнять программирование программы на ЯВ ассемблера или других ЯП низких уровней. [3]

Главным предназначением было сделать еще более приятным сам процесс программирования, упрощать его для некоторых программистов, которые имеют свое видение на реализацию программного продукта.

До какого-то времени определенного какого-то графика по разработке для языка С++ не было. Реализация и документирование для всех средств шли параллельно.

Поэтому указанный язык продолжает развиваться для быстрого преодоления некоторых проблемы, возникающие для разных пользователей.

Примерно в 1984 г. стало очевидным, что работы по стандартизации С++ будут неизбежными и надо приступить незамедлительно к проектированию базиса.

Организация AT&T внесла также свой вклад в рассматриваемый этап работы. Больше 90 представителей выполняли приемы для изучение и комментировали аспекты языка, что стали современной версией для руководства по С++.

В процессе разработки ЯП С++ самым важным этапом была его простота. Поскольку при возникновении вопросов, именно что надо упростить: либо руководство, или документацию, то всегда выбирали первое. Огромное значение также все разработчики придавали совместимости ЯП С++ с С, что мешало изменить весь синтаксис.[9]

В ЯП С++ не используются различные типы данных, а также и операции высокого уровня. К примеру, в нем не существует одного типа под названием «матрица» с операциями обращения ее, или «строка» с операциями встроенной конкатенации.

Хотя иногда пользователю понадобится самому создать некоторый тип, то он может его определить с легкостью посредством самого языка. Написание программ в С++ также сводится и к определению зависящих параметров для типов или области программирования. [9]

Язык С++ создавался и по причине использования его для традиционной сферы, а именно в системах для программирования С на ОС Linux. Но есть уже обоснованные способы по применению С++ в этой богатой программной среде. К примеру, все системы трансляции, динамическую загрузка данных и БД, можно применять также и в Linux.[16]

ЯП Java является кросс-платформенным языком программирования, что был создан корпорацией Sun Microsystems.

Все имеющиеся приложения для Java будут также обычно компилироваться в специально созданный стандартный байт-код, а потому работать они могут практически на всех виртуальных Java-машинах быть выполнены не зависимо от любой архитектуры ПК.

Сегодня популярная технология разработки ПО Java достигает своего совершенства, она может предоставлять инструменты по превращении любых статических веб-страниц в интерактивные динамические документы, что могут использоваться для создания распределенных и не зависящих от платформ приложений.

Чтоб успешно конкурировать с другими производителями в сфере создания бытовой электроники, любая из компаний также должна рассматривать для работы процессоры в качестве специального товара, который заменятся на более дешевые, а также обеспечивать совместимость работы, соблюдать все возможные стандарты.

Java – не лишь ЯП, но и специальная платформа проектирования приложений.

Изначально указанный язык назывался Oak, и был разработан Джеймсном Гослингтом с целью программирования разных бытовых электронных устройств.

Потом, его переименовали, и в последствии, на Java и стал использоваться для создания самых разных клиентских приложений серверного программного обеспечения.

ЯП был назван в честь вида кофе Java, который любили некоторые из разработчиков, поэтому на официальном логотипе изображена чашка с парящим кофе.

Также существует и другая из версий происхождения этого названия, Java – это сленговое название кофе со специальной кофе-машины, как пример для бытового устройства или выполнения программ для которых он изначально был создан.

Чтобы не связывать разработку программного обеспечения с конкретно применяемой платформой, Д. Гослингт начал использовать разные расширение компилятора С++.

Заметим, со временем он понял, что только один С++ не будет удовлетворять всем необходимым потребностям, как бы не расширять его. Поэтому в середине 1990 г. спроектировал язык Oak.

Программы на Java сразу транслируются в байт-код, который выполняется на любой виртуальной машине Java посредством специальной программы, что обрабатывает транслированный код и сразу впередает се инструкции оборудованию методами работы интерпретатора.

Достоинство методов выполнения программ находится в полной независимости от рассмотренного байт-кода и другого оборудования, которое выполняет все имеющиеся Java-приложения в основном на любых устройствах, для которых существует уже соответствующая виртуальная машина.

Еще одной специфической особенностью технологии считается гибкая система безопасности, благодаря чему выполнение программы полностью может контролироваться виртуальной машиной.

Практически все основные операции, которые превышают установленные полномочия программ (а именно, попытка несанкционированного доступа, соединения с ПК) вызывают немедленное их прерывание.

Часто к основным недостаткам концепции применения виртуальной машины можно относить и то, что применение специального байт-кода для конкретной виртуальной машине значительно будет снижать общую производительность программ и алгоритмов, что реализованы с помощью Java.

В последнее время внесен ряд новых усовершенствований, которые увеличивали скорость выполнения некоторых программ.

* широкое использование кода в платформенно-ориентированном виде;
* применение технологий трансляции для специального байт-кода в машинный код прямо во время работы программы и сохранения промежуточных версий,
* аппаратные средства, что обеспечивают ускоренную обработку байт-кодов.

Идеи, что заложены в концепцию, а различные системы для реализации виртуальных машин в Java, значительно вдохновляли энтузиастов для расширения перечня языков, что могли бы быть применены для создания программных продуктов.

Такие идеи нашли выражение в структуре CLI.[20]

# ГЛАВА 2. СОЗДАНИЕ ГРАФИЧЕСКОГО ПРИЛОЖЕНИЯ С ПОМОЩЬЮ С++

## 2.1.Обзор C++ Builder

Почти каждый ЯП высокого уровня имеет также разработанную для него IDE, а именно интегрированную среду по разработке и реализации программ. К примеру: Visual Studio, C++ Builder, NetBeans и Eclipse. Все эти системы предназначаются также для реализации в них в более сложных графических программах.[1]

IDE C++ Builder считается средством, выпущенным корпорацией Borland, для проектирования программ, которые дают возможность для разработки оконного приложения с использованием C++, при этом используя библиотеку VCL. [19]

C++ Builder является специальным SDI-приложением, главное окно для которого может использовать инструментальную панель, а также палитры компонентов (рисунок 4). [7]

Рисунок 4 – Окно C++ Builder

Помимо этого, при запуске C++ Builder будут отображены окно инспектора объектов, формы и другие панели.

Формой является основа для приложений в C++ Builder. Стоит отметить, что в стандартном создании интерфейса для приложения нужно добавлять в окно формы некоторые элементы объектов с системы C++ Builder, что также часто называют компонентами.

Все компоненты располагаются в специальных палитрах, которая выполнена в качестве инструментальной панели с вкладками. [18]

Важной особенностью C++ Builder является возможность проектирования собственных пользовательских компонентов, при этом можно настраивать всю палитру. В среде возможность для создания самых различные версий палитр.[18]

Компоненты разделены на визуальные и невизуальные. К примеру, визуальные могут появляться при запуске программ (к примеру, списки, редактируемые поля). Невизуальные – отображены лишь при проектировании, а во время выполнения программы их не видно, хотя они также и обладают определенной степенью функциональности (запуск таймера, открытие окон) [6](рисунок 5).

Рисунок 5 – Использование разных типов компонентов

Каждый компонент, применяемый в C++ Builder использует 3 свои характеристики:

– методы;

– свойства;

– события.[8]

Если выбирать компонент на палитре непосредственно, потом добавлять на форму, инспекторы объектов могут отобразить автоматически события и свойства (рисунок 6). [3]

Рисунок 6 – Инспектор объектов

Некоторые основные свойства компонентов называются атрибутами, определяющими поведение и внешний их вид. Все свойства компонентов в таких столбцах свойств принимают самые разные значение, которые устанавливаются по умолчанию. [8]

Свойство определенного компонента может отображаться и на вкладке свойств. А в это время инспекторы для объектов могут отображать некоторые опубликованные свойства компонентов. [14]

Помимо рассмотренных свойств, все компоненты используют также общие свойства, являющиеся доступными во время программного запуска. Инспектор объектов применим при установке некоторых основных свойств и проектировании программ.

## 2.2. Разработка проекта ИС банкомата

Рассмотрим методику создания графической программы на примере создания программного продукта для обслуживания банкомата. [20]

Графическая программа предназначена для формирования заказа пользователя на наличку.

Программа написана в среде C++ Builder, что дает возможность использования большое количество различных визуальных компонентов, встроенных функций.

При проектировании программы создано 3 пользовательских класса, 22 функции обработки данных.

Созданы скрытые возможности для обслуживания банкомата: пополнения денежного хранилища банкомата и извлечение заблокированных карт из банкомата.

Обращение к программе производится в несколько этапов:

– при вводе банковской карты в терминал;

– при вводе PIN-кода;

– при снятии денежных средств и просмотра остатка на счету;

– при обслуживании банкомата с помощью «секретных возможностей».

Входными данными являются PIN-код для активации работы с банкоматом, остаток на персональном банковском счету, сумма в хранилище банкомата.

Выходные данные будут представлены в виде сообщений-квитанций об получении запрашиваемой суммы денег, пополнения хранилища банкомата.

В программе используются следующие сообщения:

Таблица 1 – Сообщения, использующиеся в программе

|  |  |  |
| --- | --- | --- |
| № п/п | Сообщение | Действия по сообщении |
| 1 | Вставьте карту | Нужно вставить карту в терминал |
| 2 | Введите PIN-код | Нужно ввести PIN-код. В случае неправильного ввода будут еще 2 попытки, в случае дальнейшего неправильного ввода карта будет заблокирована и передана в хранилище. |
| 3 | Введите сумму | Нужно ввести сумму для снятия с банковской карты |
| 4 | Вы сняли сумму Х. Остаток Y | Сообщение-квитанция об снятии денежных средств с карты |
| 5 | Заберите деньги | Надо нажать кнопку «Забрать» |
| 6 | Заберите карту | Нужно изъять карту (нажать на кнопку «Изъять») |
| 7 | Введите сумму пополнения хранилища | Нужно ввести сумму для пополнения денежного хранилища банкомата |
| 8 | Всего в хранилище Х | Вывод информации об балансе хранилища. |
| 9 | Карта заблокирована | Произведена блокировка карты |

Рассмотрим методику работы с созданной программой.

После запуска программы интерфейс приобретет следующий вид (рисунок 7):



Рисунок 7 – Форма программы после запуска

Стоит отметить, что пока карта не вставлена (не нажата кнопка «Вставить») все остальные элементы формы будут неактивными. После вставки карты будет предложено ввести PIN-код (рис.8):



Рисунок 8 –Ввод PIN-кода

В папке с проектом имеется файл baza.txt в котором внесен PIN-код (1111) и сумма денежных средств пользователя, больше которой он не может снять.

Если PIN-код введен неправильно 3 раза, то карта конфискуется, о чем сигнализируется в разделе «Конфискация карты» (рис.9):



Рисунок 9 –Произведена блокировка карты

Извлечь карту можно с помощью «секретной кнопки».

После ввода PIN-кода и нажатия на кнопку «Enter» появится сообщение о выборе одного из двух вариантов работы банкомата: «Получения налички» и «Просмотр остатка на счету» (рис.10):



Рисунок 10 –Выбор вариантов работы банкомата

В случае выбора переключателя «Остаток» и нажатия на кнопку «Ввод» будет отображена сумма остатка на которая пользователя, которая считывается с текстового файла baza.txt (рис.11):



Рисунок 11 – Отображение баланса пользователя

В случае выбора пункта «Снять наличку» нужно ввести необходимую суму и нажать кнопку «Ввод». В последствии отобразится сообщение (рис.12):



Рисунок 12 – Сообщение-квитанция о снятии денежных средств

В случае, когда введенная сумма превышает баланс на банковской карте, будет показано сообщение (рис.13):



Рисунок 13 – Сообщение о слишком большой сумме для вывода

После снятия денег система предложит забрать деньги и извлечь карту (рис.14):



Рисунок 14 – Окончание обслуживания клиента

Для осуществления возможности пополнения денежного хранилища банкомата или извлечения конфискованных карт нужно нажать на место, обведенное на рис. 15 справа красным овалом:



Рисунок 15 – Секретные кнопки

В результате окно формы примет вид (рис.16):



Рисунок 16 – Операции для обслуживания банкомата

При нажатии на кнопку «Забрать карты с хранилища» количество конфискованных карт обнулится. При нажатии на кнопку «Пополнить хранилище» будет предложено с помощью окна для ввода данных пополнить счет на определенную сумму. После этого появится окно с подтверждением (рис.17):



Рисунок 17 – Подтверждения пополнения денежного хранилища банкомата

Для скрытия кнопок для обслуживания банкомата нужно нажать на место, обведенное красным овалом (слева).

Проанализировав главу 2 можно сделать вывод, что графические приложения в наше время являются самыми популярными как среди пользователей, так и среди программистов. Использование оконного интерфейса дает новые возможности в обеспечении удобного интерфейса, графического отображения результатов вычислений, реализации модели реально действующих устройств.

# ЗАКЛЮЧЕНИЕ

В нынешнее время, при росте связей между миром бизнеса и коммерции с миром проектирования и разработки ПО, все большие корпорации тратят очень много финансовых средств и своих усилий на непосредственное планирование бизнеса, ощущается очень острая надобность в соответствии с разными абстрактными бизнес-процессами, программными или функциональными реализациями.

Стоит отметить, что большинство ЯП какого-то прямого пути для реализации связи кода с логикой, к сожалению, совсем не имеют.

К примеру, в нынешнее время разные программисты часто могут комментировать свои созданные программные продукты для подробного объяснения, какие именно классы реализуют какой-то абстрактный бизнес-объект.

К примеру, ЯП C# позволяет применять очень масштабные типизированные, а также расширяемые метаданные, что, в свою очередь, часто могут применяться к некоторому конкретному объекту.

Архитектурой всякого проекта также могут определяться все локальные программные атрибуты, которые как-то будут связанны с всяческими элементами ЯП.

В процессе выполнения представленного исследования были получены, а также закреплены практические навыки для разработки программ при помощи различных визуальных компонентов в написании программных продуктов для среды C++ Builder.

В процессе написания курсовой работы были реализованы следующие задачи:

– рассмотрены современные источники литературы по теории разработки информационных систем;

* подробно дана характеристика классификации ЯП;
* охарактеризована среда по разработке ПО C++ Builder;
* создано с помощью C++ Builder приложения для банкомата.

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Бобровский С. Самоучитель програмирования на языке C++ в среде Borland C++ Builder М.: ИНФРА-М, 2014.–251 c.
2. Бруно Бабэ. Просто и ясно о Borland C++: Пер. с англ. - Москва: БИНОМ, 2014. – 400с.
3. Заварыкин В.М.. Основы информатики и вычислительной техники. — М.: Просвещение, М.: - 2015. 180 с.
4. Задачи по программированию / С. А. Абрамов, Г. Г. Гнездилова, Е. Н. Капустина, М. И. Селюн. — М.: Наука, 2016. – 344 с.
5. Зубов В. С. Программирование на языке С++. — М.: Информационно-издательский дом «Филинъ», 2013. – 452 с.
6. 3yeв Е. А. Практическое программирование на языке С++. — М.: Радио и связь, 2014. – 406 с.
7. Информатика. Задачник-практикум: В 2 т. / Под ред. И. Г. Семакина, Е.К.Хеннера. — М.: Лаборатория Базовых Знаний, 2013. – 182 с.
8. Йенсен К. С++ — руководство для пользователей и описание языка. — М.: Мир, 2016. – 402 с.
9. Касаткин В. Н. Информация. Алгоритмы. ЭВМ. — М.: Просвещение, 2015. – 210 с.
10. Керниган Б. Язык программирования Си: Пер. с англ. — М.: Финансы и статистика, 2014. – 322 c.
11. Культин Н.Б. Визуальное программирование.— СПб.: BHV — Санкт-Петербург, 2013. – 266 c.
12. Культин Н.Б. С++ Buider в задачах и примерах – СПб.:БХВ-Петербург, 2016. – 336с.: ил.
13. Лаптев В.В. C++. Экспресс-курс . М.: 2014.–322 с.
14. Оллисон Ч. Философия С++. Практическое программирование. С.Петербург 2014. – 608 с.:ил.
15. Послед Б.С. Borland C++ Builder 6. Разработка приложений баз. М.: 2013г. -360 г.
16. Стенли Б. Липпман. C++ для начинающих: Пер. с англ. 2тт. - Москва: Унитех; Рязань: Гэлион, 2016. – 345с.
17. Строуструп Б. Справочное руководство по языку C++ с комментариями: Пер. с англ. - Москва: Мир, 2016. – 445с.
18. Технология разработки приложения на языке С++. Методическое указание к лабораторным работам для студентов первого курса специальности 080801.65 “Прикладная информатика”. МГУ. – М.: – 2013.–222 с.
19. Холингворт Д. Учебник по программированию в среде С++ Builder 5. – Наука.–М.: 2015. –865 с.
20. Юпашников A.M. Программирование в среде C++ Builder. — М.: МИФИ, 2014. – 360 c.