



Государственное бюджетное образовательное учреждение высшего образования
Московской области

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ

Колледж космического машиностроения и технологии

КУРСОВОЙ ПРОЕКТ

По дисциплине «Прикладное программирование»

Тема: «Разработка симулятора обменного пункта валюты»

Выполнил студент

Трифонов Кирилл Алексеевич

Группа П2-17

_____ (подпись)

_____ (Дата сдачи работы)

Королев, 2020

Оглавление

Введение.....	4
Глава 1. Теоретическая часть	5
1.1. Описание предметной области.....	5
1.2. Описание существующих разработок	6
Глава 2. Проектная часть.....	8
2.1. Диаграмма прецедентов.....	8
2.2. Выбор инструментов	9
2.3. Проектирование сценария	11
2.4. Диаграмма классов	13
2.5. Описание главного модуля	14
2.6. Описания спецификаций к модулям	17
2.7. Описание модулей	19
2.8. Описание тестовых наборов модулей	20
2.9. Описание применения средств отладки	25
2.10. Анализ оптимальности использования памяти и быстродействия ..	27
Глава 3. Эксплуатационная часть.....	28
3.1. Руководство оператора	28
Заключение	37
Список литературы и интернет-источников	38
Приложение 1. Код главного модуля bank.cpp	39
Приложение 2. Код модуля Person.h	41
Приложение 3. Код модуля Money.h.....	42
Приложение 4. Код модуля constants.cpp	43

Приложение 5. Код заголовочного файла functions.h	44
Приложение 6. Код модуля functions.cpp	45

Введение

Целью данного курсового проекта является написание программы "Обменник валют", которая сможет заменить кассира в компании, предоставляющей услуги обмена валют. Это нужно для модернизации и автоматизации процесса покупки иностранной валюты. Данный проект будет отличаться быстроедействием и низкими требованиями к системным требованиям.

В первой части будут рассмотрены предметная область данной темы и несколько готовых решений.

Во второй части будут рассмотрены разработанные модули и инструменты, а также листинги некоторых частей программы.

В третьей части будет рассмотрено руководство оператора.

Глава 1. Теоретическая часть

1.1. Описание предметной области

Обмен валют - распространённая и прибыльная сфера бизнеса. Обмен валют необходим для:

- Международной торговли, так как каждая страна имеет собственную валюту. Для этого обмен валют происходит на уровне государственных банков.
- Туризма, так как при путешествиях необходима валюта страны, в которую отправляется турист. Для этого обмен валют происходит на уровне частных банков и обменников валют.
- Хранения сбережений в виде иностранных валют, особенно актуально в сегодняшних реалиях. Для этого используются частные банки и обменники.

В современном мире используются онлайн обменники, которые позволяют не только обменивать валюту, но и переводить деньги между счетами в разных банках, используя интернет.

1.2. Описание существующих разработок

На рынке программного обеспечения существуют множество банковских систем и отдельных программных продуктов, автоматизирующих валютно-обменные операции.

Первое рассмотренное решение - это "Union Business System" компании «ЮниСАБ[1]». В данной системе разработан бизнес «обменный пункт». Программа «Обменный пункт» обеспечивает полную автоматизацию работы кассира "выносного" обменного пункта и опосредованную связь с банком. Его широкие возможности позволяют настраивать программный модуль в соответствии с регламентом работы обменного пункта. Все выполняемые операции сопровождаются формированием необходимых печатных документов.

Параметры каждой операции имеют гибкие настройки в соответствии с внутрибанковской инструкцией работы обменных пунктов. Программа готовит полный набор документов авансовая заявка и заявка на подкрепление, реестры по всем выполняемым операциям, справка об остатках, препроводительные ведомости (валютная и рублевая), акт передачи другому кассиру.

Плюсами данного программного средства являются:

1. Возможность ведения электронного реестра валютно-обменных операций;
2. Формирование необходимых печатных документов и бланков строгой отчетности;
3. Простота в обучении персонала.

К минусам можно отнести:

1. Высокие требования к аппаратным средствам.

Второе рассмотренное решение - "Центавр-Дельта"[2]. Это современное решение, основанное на технологии клиент-сервер, которое обеспечивает высокую производительность при автоматизации банковской деятельности и исключает затраты, связанные с традиционными SQL системами управления базами данных.

В данном продукте присутствует модуль валютно-обменных операций. Среди основных возможностей можно отметить полную автоматизацию валютно-обменных операций, ведение операций по покупке-продаже иностранной валюты за рубли, ведение конверсионных операций с получением соответствующих отчетов, контроль правильности ввода информации, наличие девяти уровней доступа к информации и выполнению банковских операций, и другое.

Плюсы данного программного средства:

1. Возможность ведения электронного реестра валютно-обменных операций;
2. Формирование необходимых печатных документов и бланков строгой отчетности;
3. Простота в обучении персонала.

К минусам можно отнести:

1. Высокие требования к аппаратным средствам.

В разрабатываемом программном продукте достоинствами являются:

1. Простота в обучении персонала;
2. Низкие требования к аппаратным средствам.

Глава 2. Проектная часть

2.1. Диаграмма прецедентов

Для определения вариантов использования к проекту была построена диаграмма прецедентов

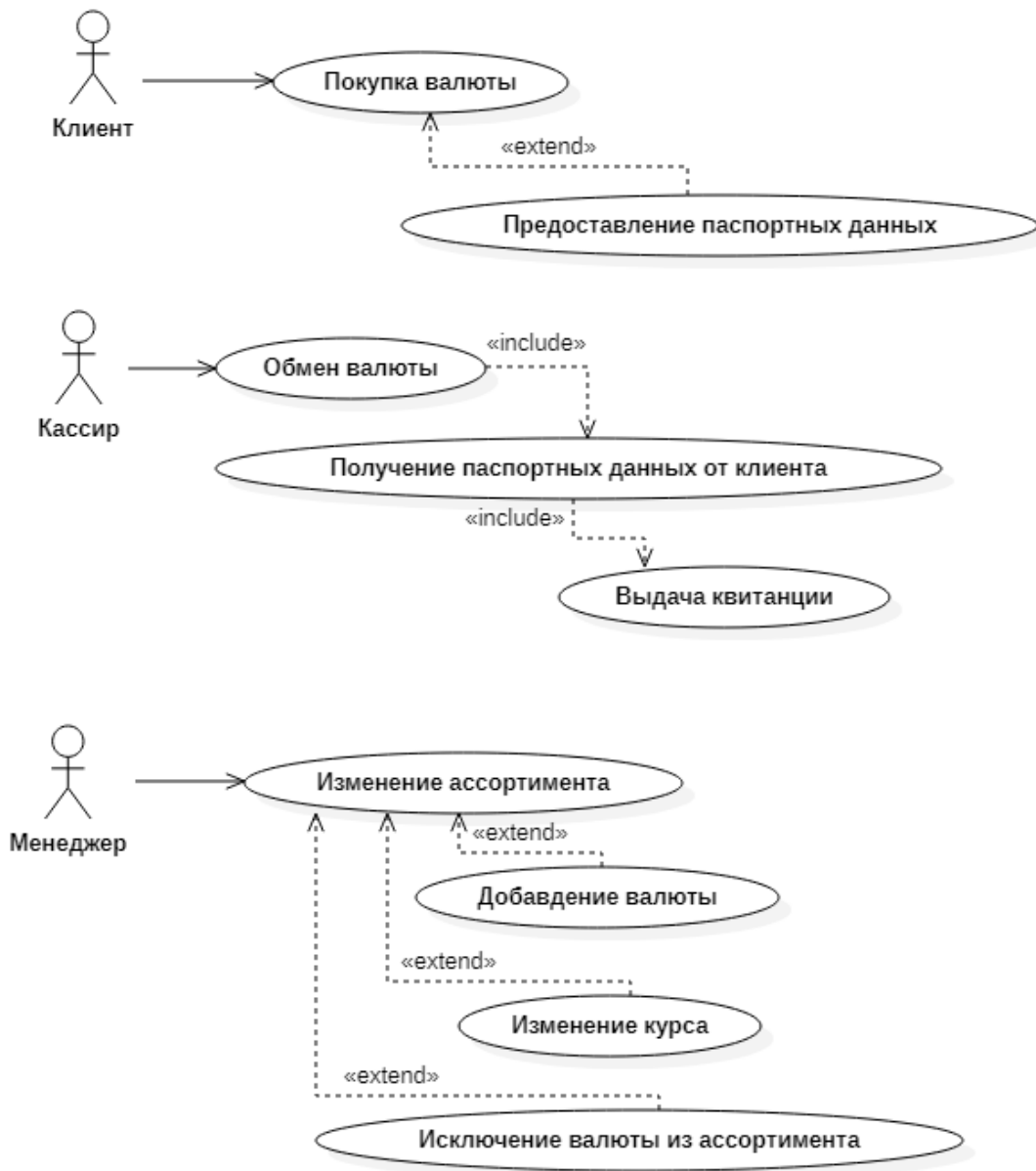


Рисунок 1. Диаграмма прецедентов для проекта

2.2. Выбор инструментов

При выборе инструмента и языка программирования было проведено сравнение по критериям, которые представлены в таблицах 1 и 3.

Степень важности критерия выбиралась из: низкая, ниже средней, средняя, ниже высокой, высокая.

Таблица 1. Критерии выбора инструмента.

Критерий	Важность критерия
Функционал	Ниже высокой
Удобство использования	Высокая
Скорость разработки	Ниже высокой

Исходя из данных критериев, я сравнил 3 языка программирования от 0 до 10 баллов за критерий

Таблица 2. Оценка языков программирования

Критерий/Язык программирования	C++	Python	C
Функционал	9	10	7
Удобство использования	9	7	6
Скорость разработки	8	7	6
Итого	26	24	19

Таблица 3. Критерии выбора среды разработки.

Критерий	Важность критерия
Простота	Средняя
Функционал	Высокая
Удобство использования	Высокая
Документация на русском языке	Ниже средней

Исходя из данных критериев, я сравнил 2 среды разработки от 0 до 10 баллов за критерий.

Таблица 4. Оценка сред разработки

Критерий/Среда разработки	CodeBlocks	Microsoft Visual Studio
Простота	6	5
Функционал	7	10
Удобство использования	6	8
Документация на русском языке	5	9
Итого	24	32

По результатам сравнения была выбрана среда разработки Microsoft Visual Studio

Microsoft Visual Studio — инструментальная среда разработки, включающая в себя интегрированную среду разработки, редактор исходного кода, встроенный отладчик. Многие другие инструменты возможно получить благодаря подключению плагинов — сторонних расширений. Эта среда разработки была выбрана из-за совокупности критериев оценки.

2.3. Проектирование сценария

В данном разделе приведён пример сценария использования программы оператором и клиентом.

После запуска программы клиент может выбрать валюту и её необходимое количество. Затем он должен ввести свои паспортные данные (ФИО, номер паспорта). После этих действий программа выдаст квитанцию об оплате.

Также, для оператора предусмотрен режим редактирования данных, в который можно получить доступ, используя пароль. В нём оператору доступны различные функции, подробнее о которых можно узнать в Главе 3 "Руководство оператора".

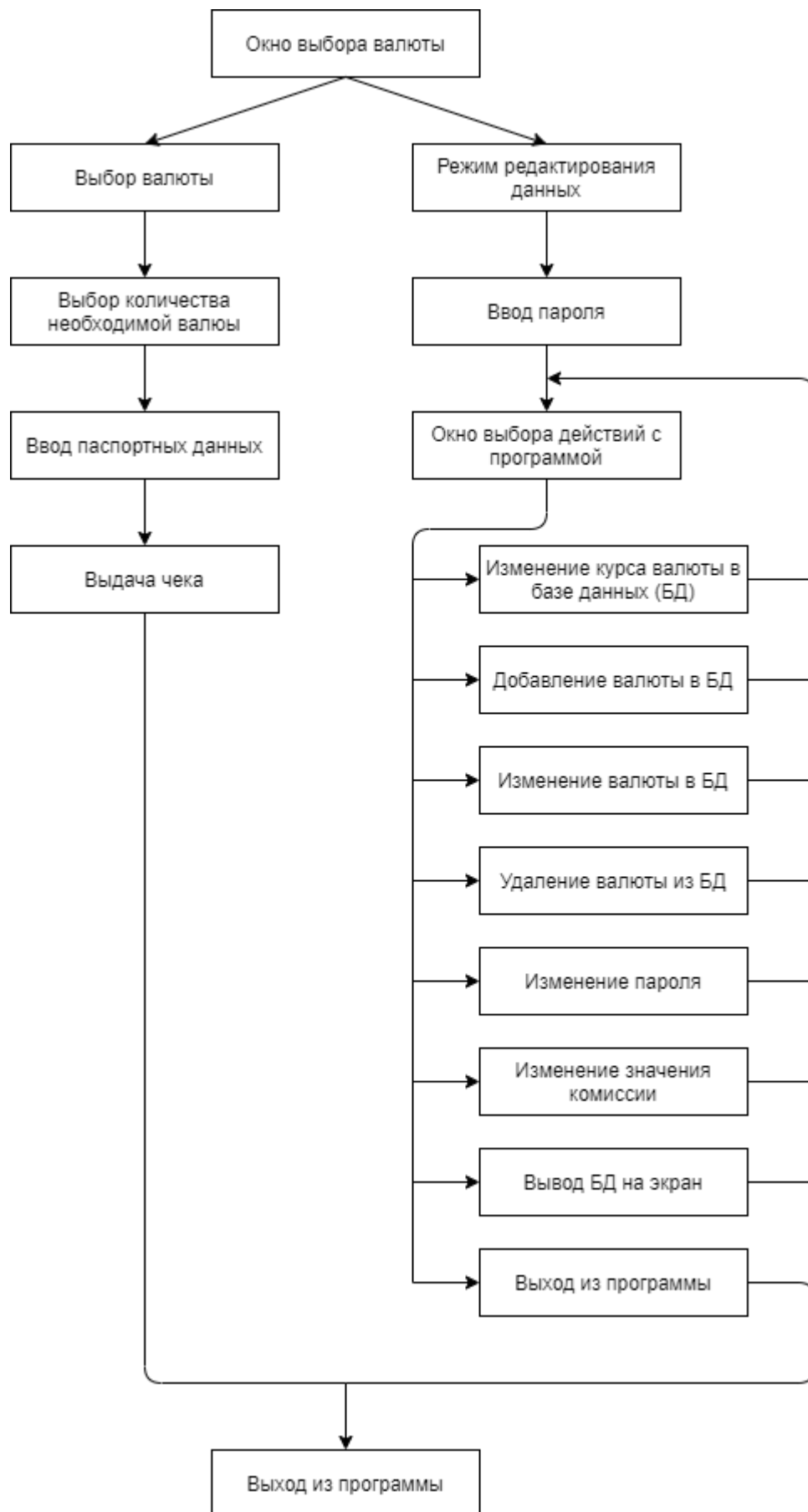


Рисунок 2. Сценарий использования

2.4. Диаграмма классов

В диаграмме классов представлены два класса Money (для валюты) и Person (для клиента).

В классах Money и Person используется спецификатор доступа public, что говорит нам о том, что доступ открыт всем, кто видит определение данных классов.

Блок стоящий после названия класса показывает нам объявленные переменные, а знак “+” указывает на спецификацию доступа, в данном случае только public.

После блока с атрибутами, расположен блок с операциями (функциями класса).

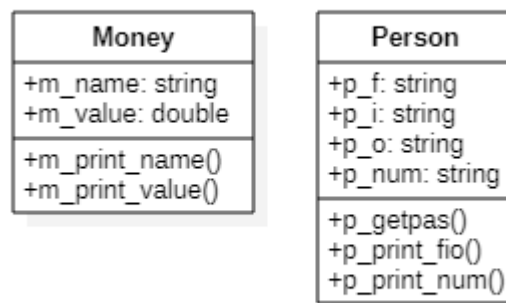


Рисунок 3. Диаграмма классов

2.5. Описание главного модуля

Главный модуль представляет собой файл bank.cpp.

К главному модулю подключаются остальные модули, содержащие в себе классы и функции. Так же в нём проверяется есть ли данные в БД.

Листинг 1. Код главного модуля

```
#include "functions.h"
#include "Person.h"

extern vector <Money> dollar;

int main()
{
    system("chcp 1251");
    system("cls");
    double commission = get_commission();
    int val;
    Person jd;
    getff();
    print();

    if (dollar.size() < 1) //проверка на отсутствие данных в бд
    {
        cout << "Ассортимент пуст, обратитесь к менеджеру" << endl;
        cin >> val;
        if (val == 0)
            worker();
    }

    else
    {
        val = choose_your_destiny();
        if (val == 0)
        {
            worker();
        }
        else
```

```
{
    double amount;
    cout << "Введите количество необходимой валюты: ";
    cin >> amount;
    jd.p_getpas();

    print_bill(jd, calculate_sum(amount, dollar[val-
1].m_value, commission),
               commission, amount, val);
}
}
system("pause");
return 0;
}
```

Нижe представлена схема главного модуля программы

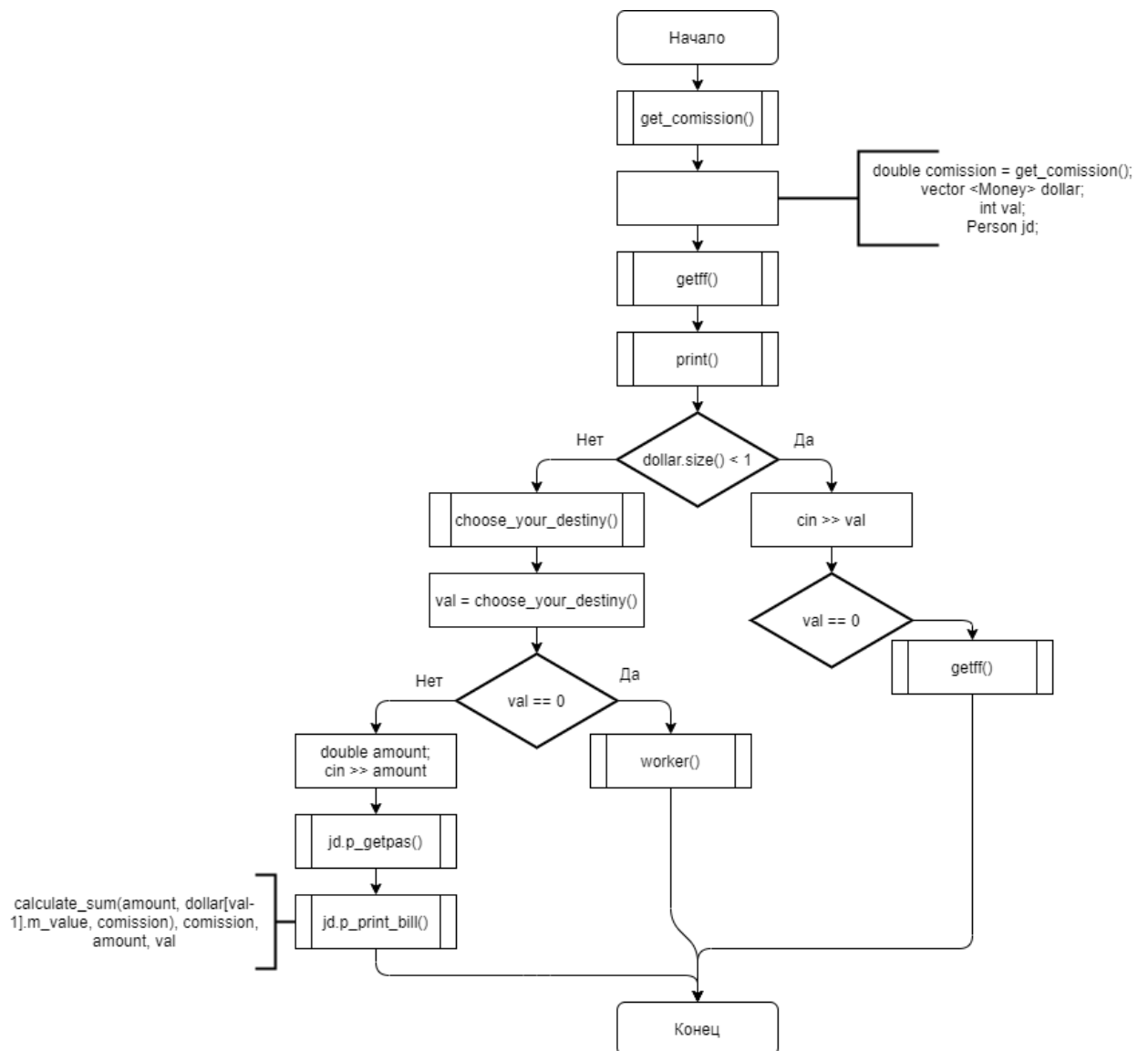


Рисунок 4. Блок-схема главного модуля

2.6. Описания спецификаций к модулям

Спецификация к модулю Money (Листинг 2).

Листинг 2. Код класса Money

```
#pragma once
#include <iostream>

using namespace std;

class Money // класс для валюты
{
public:
    string m_name; // название валюты
    double m_value; // курс валюты к рублю

    Money(string name, double value)
    {
        this->m_name = name;
        this->m_value = value;
    }

    void m_print_name() // вывод названия с учётом '_'
    {
        for (int i = 0; i < m_name.length(); i++)
        {
            if (m_name[i] == '_')
                cout << " ";
            else
                cout << m_name[i];
        }
    }

    void m_print_value() // вывод курса
    {
        cout << m_value;
    }
};
```

Спецификация к модулю Person (Листинг 3).

Листинг 3. Код класса Person

```
#pragma once
#include <iomanip>
#include <vector>
#include "Money.h"

using namespace std;

class Person // класс для паспортных данных пользователя
{
public:
    string p_f; // фамилия
    string p_i; // имя
    string p_o; // отчество
    string p_num; // номер паспорта

    void p_getpas() // метод получения паспортных данных клиента
    {
        cout << "Введите свои паспортные данные (ФИО, серия и №) через пробел: ";
        cin >> p_f >> p_i >> p_o >> p_num;
    }

    void p_print_fio() // вывод ФИО
    {
        cout << p_f << " " << p_i << " " << p_o;
    }

    void p_print_num() // вывод номера паспорта
    {
        cout << p_num;
    }
};
```

2.7. Описание модулей

В данной главе описаны используемые модули.

1. Главный модуль

Главный модуль представляет собой файл `bank.cpp`. К главному модулю подключаются остальные модули, содержащие в себе классы и функции. Так же в нём проверяется есть ли данные в БД.

2. Модуль функций

В модуле функций находятся функции, используемые в проекте:

- `getff()` - заполнение БД данными из входного файла
- `print()` - вывод БД на экран
- `vector_to_file()` - заполнение входного файла данными из БД
- `get_commission()` - получение комиссии из входного файла
- `choice_your_destiny()` - функция выбора валюты пользователем
- `add_field()` - функция добавления поля в БД
- `edit_field()` - функция изменения курса валюты в поле БД
- `delete_field()` - функция удаление поля БД
- `change_field()` - функция изменения поля БД
- `change_password()` - функция изменения пароля входа в режим редактирования
- `change_commission()` - функция изменения комиссии
- `worker_menu()` - функция выбора действия с БД
- `worker()` - функция проверки пароля для доступа к режиму редактирования БД
- `calculate_sum()` - функция вычисления итогового счёта
- `print_bill()` - функция выводение окончательного счёта на экран

3. Модуль, содержащий класс Money (Money.h)

Класс Money предназначен для корректной работы базы данных. В него входят следующие поля и методы:

- поле названия валюты
- поле курса валюты (относительно рубля)
- метод, предназначенный для вывода названия валюты
- метод, предназначенный для вывода курса валюты

4. Модуль, содержащий класс Person (Person.h)

Класс Person предназначен для работы с данными клиента (фамилия, имя, отчество, серия и номер паспорта)

2.8. Описание тестовых наборов модулей

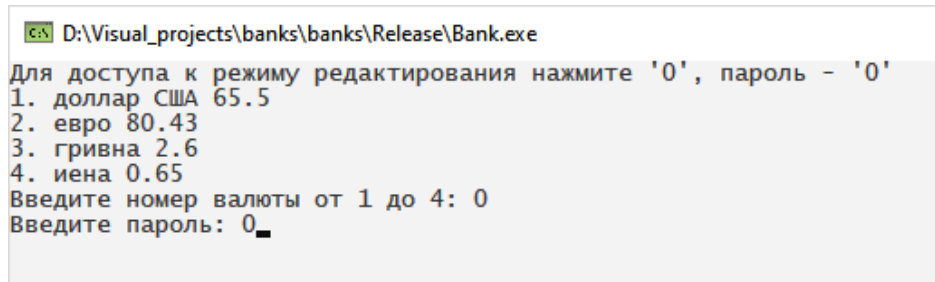
В этом разделе будут продемонстрированы результаты тестирования "черного ящика"

Тест 1. Изменение курса валют

Действия: Зайти в режим редактирования, выбрать действие "изменение курса валют", изменить курс выбранной валюты, в данном случае валюта - "Доллар США".

Ожидаемый результат: Изменение предыдущего курса валюты "Доллар США" (65.5р) на новое значение - 70.5р.

Результат теста:



```
D:\Visual_projects\banks\banks\Release\Bank.exe
Для доступа к режиму редактирования нажмите '0', пароль - '0'
1. доллар США 65.5
2. евро 80.43
3. гривна 2.6
4. иена 0.65
Введите номер валюты от 1 до 4: 0
Введите пароль: 0
```

Рисунок 5. Вход в режим редактирования данных

```
CS D:\Visual_projects\banks\banks\Release\Bank.exe
1. Изменение курса валюты
2. Добавление поля
3. Изменение поля
4. Удаление поля
5. Изменение пароля
6. Изменение комиссии
7. Вывод таблицы
8. Выход
Выберите действие: 1

1. доллар США 65.5
2. евро 80.43
3. гривна 2.6
4. иена 0.65
Выберите номер поля: 1
Введите новое значение: 70.5
```

Рисунок 6. Изменение курса валюты "доллар США"

```
CS D:\Visual_projects\banks\banks\Release\Bank.exe
1. Изменение курса валюты
2. Добавление поля
3. Изменение поля
4. Удаление поля
5. Изменение пароля
6. Изменение комиссии
7. Вывод таблицы
8. Выход
Выберите действие: 7

1. доллар США 70.5
2. евро 80.43
3. гривна 2.6
4. иена 0.65
Для продолжения нажмите любую клавишу . . .
```

Рисунок 7. Новое значение валюты "доллар США"

Тест 2. Добавление поля в таблицу

Действия: Зайти в режим редактирования, выбрать действие "добавление поля", добавить поле "тенге" с курсом 0.17р.

Ожидаемый результат: Добавление в таблицу нового поля с валютой "тенге" и курсом 0.17.

Результат теста:

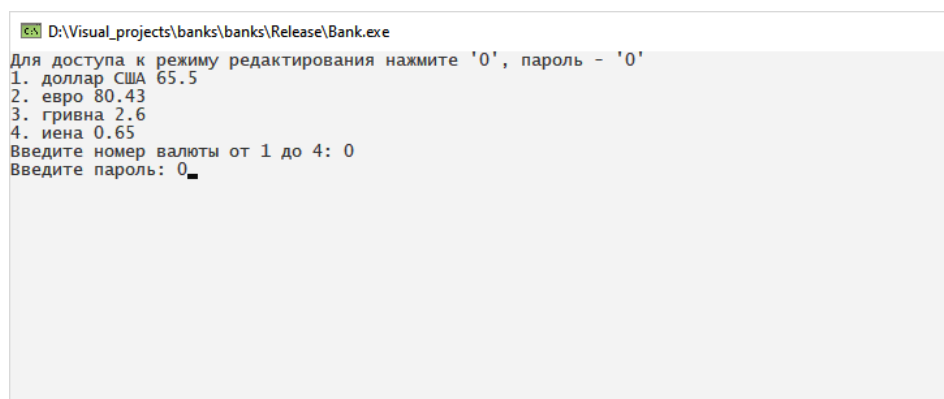


Рисунок 8. Вход в режим редактирования данных

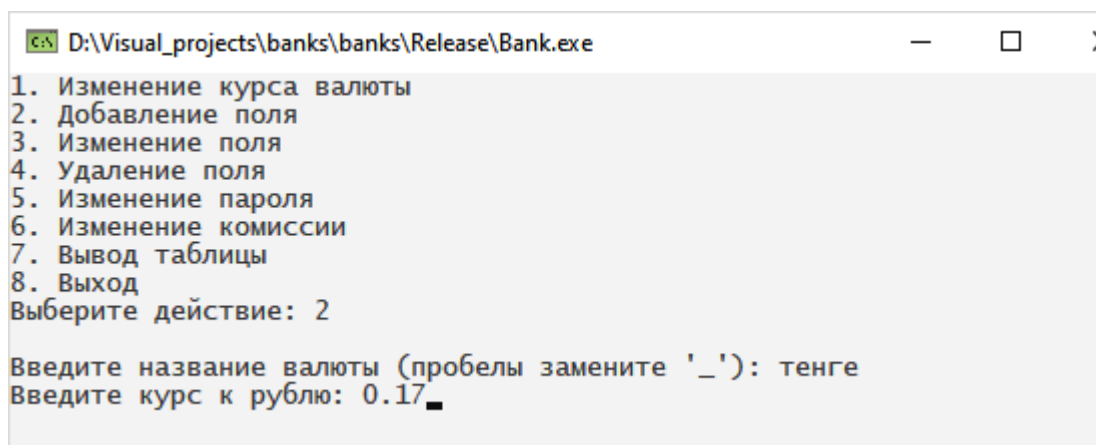


Рисунок 9. Добавление валюты

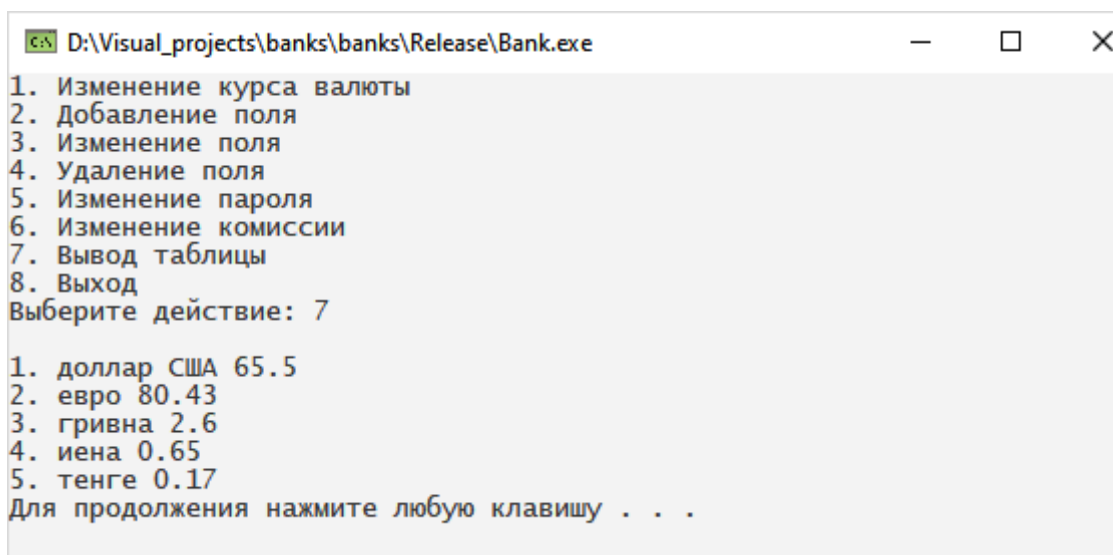


Рисунок 10. Валюта добавлена

Тест 3. Смена пароля доступа в режим редактирования данных

Действия: Зайти в режим редактирования, выбрать действие "изменение пароля", поменять пароль с "0" на "Здравствуйте, Леонид Борисович".

Ожидаемый результат: Изменение пароля на "Здравствуйте, Леонид Борисович".

Результат теста:

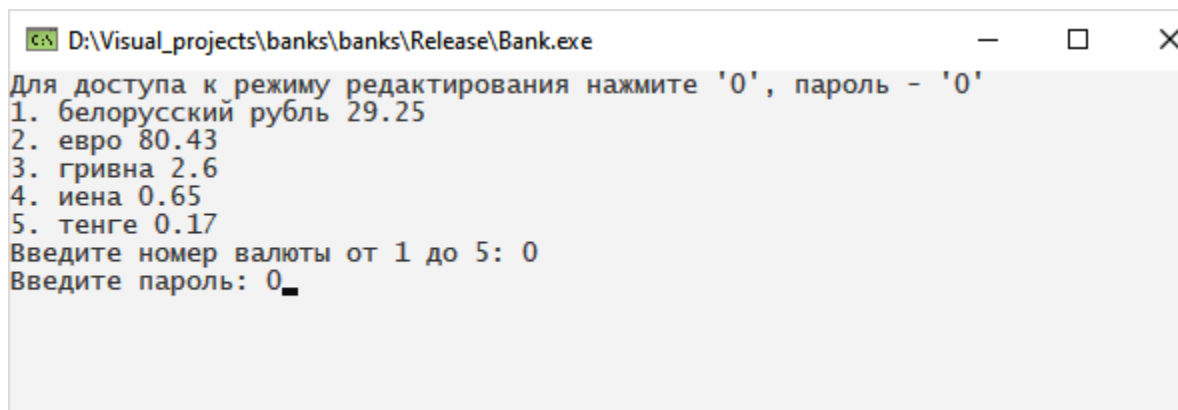


Рисунок 11. Вход в режим редактирования данных

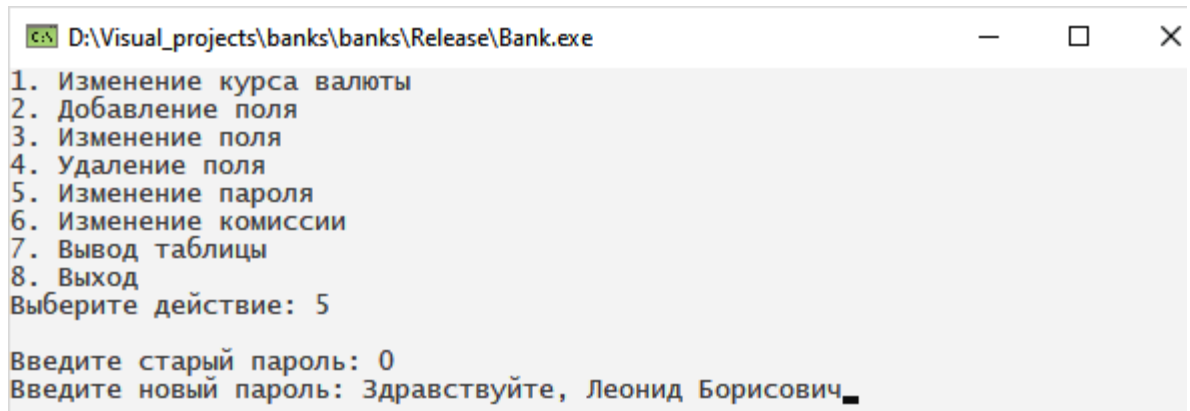


Рисунок 12. Смена пароля

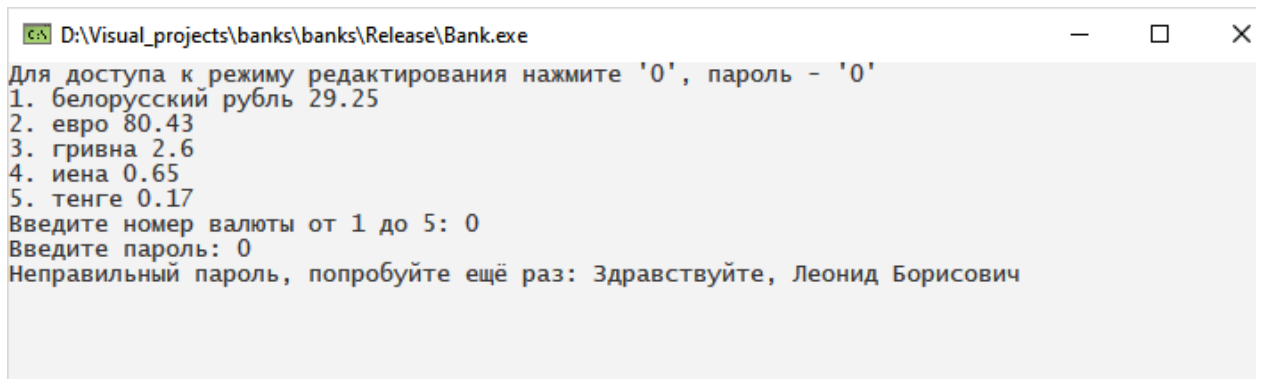


Рисунок 13. Предыдущий пароль не верен

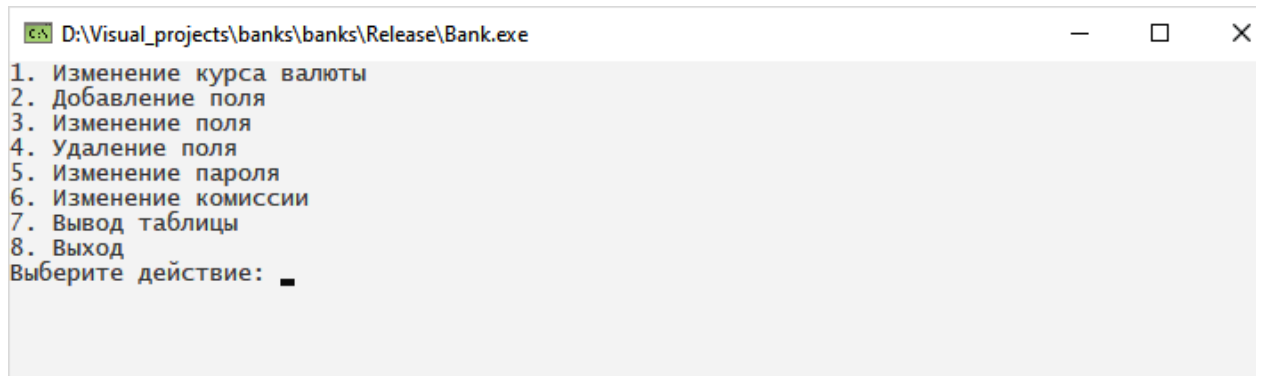


Рисунок 14. Режим редактирования, вход в который был произведён с помощью нового пароля

2.9. Описание применения средств отладки

В ходе написания курсового проекта было получено следующее сообщение:

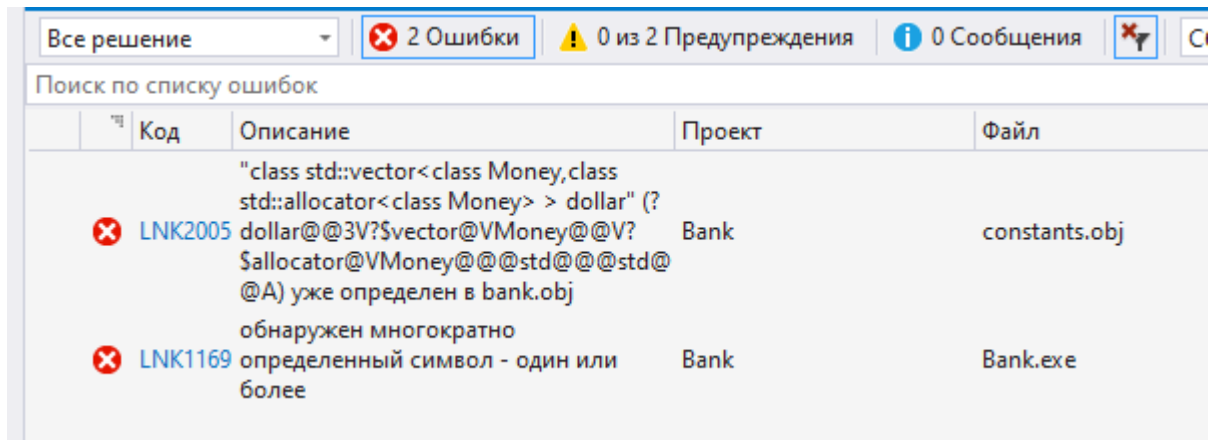


Рисунок 15. Обнаружена ошибка

Суть ошибки заключается в векторе, в котором хранится информация о валютах. У вектора отсутствует спецификатор `extern`, который сообщает компилятору, что следующие за ним типы и имена переменных объявляются где-то в другом месте. Другими словами, `extern` позволяет компилятору знать о типах и именах глобальных переменных без действительного создания этих переменных.

Для решения данной ошибки нужно перейти в файл `bank.cpp` и обратить внимание на 4 строку (рис 15).

```
#include "functions.h"
#include "Person.h"

vector <Money> dollar;
```

Рисунок 16. Ошибка в файле bank.cpp

Необходимо изменить строку `vector <Money> dollar;` на строку `extern vector <Money> dollar;`, тем самым указав компилятору, что вектор определён в другом месте.

Результатом отладки стало следующее сообщение:

```
1>Сборка проекта "banks.vcxproj" завершена.  
===== Сборка: успешно: 1, с ошибками: 0, без изменений: 0, пропущено: 0 =====
```

Рисунок 17. Результат отладки программы

2.10. Анализ оптимальности использования памяти и быстродействия

В данном разделе представлен анализ оптимальности использования памяти и быстродействия программы.

Принятые оптимальные решения:

- В целях оптимизации кода, было принято решение часто используемые блоки кода сделать функциями. Примеры таких функций представлены в Листинге 4.

Листинг 4. Примеры функций

```
void print() //вывод таблицы с данными на экран
{
    for (int i = 0; i < dollar.size(); i++)
    {
        cout << i + 1 << ". ";
        dollar[i].m_print_name();
        cout << " ";
        dollar[i].m_print_value();
        cout << endl;
    }
}

void vector_to_file() //запись таблицы с данными в файл "filename"
{
    ofstream fin(filename);
    for (int i = 0; i < dollar.size(); i++)
        fin << dollar[i].m_name << " " << dollar[i].m_value << endl;
    fin.close();
}
```

Глава 3. Эксплуатационная часть

3.1. Руководство оператора

1. Назначение программы

Функциональное назначение программы

Основной функцией проекта "Bank helper" является автоматизация процесса покупки валюты.

Эксплуатационное назначение программы

Программное обеспечение "Bank helper" может быть использовано в любом предприятии, которое занимается продажей валюты.

Состав функций

Функции оператора

1. Изменение курса валюты - отвечает за изменение курса валюты, без изменения остальных параметров поля в таблице
2. Добавление поля - добавление поля с названием валюты и курсом в конец таблицы
3. Изменение поля - полное изменение поля, включая название и курс
4. Удаление поля - удаление поля таблицы
5. Изменение пароля - изменение пароля для входа в режим редактирования данных
6. Изменение комиссии
7. Вывод таблицы на экран
8. Выход

Функции клиента

1. Выбор валюты
2. Выбор количества валюты
3. Ввод паспортных данных
4. Получение квитанции об оплате

2. Условия выполнения программы

Минимальный состав аппаратных средств

Минимальный состав используемых технических (аппаратных) средств:

- Windows 7, 8.1, 10;
- Процессор с тактовой частотой не ниже 1,8 ГГц.
- ОЗУ - 10 МБ;
- Место на жестком диске: 45+ КБ (42 КБ программа и 3 текстовых файла);
- Видеоадаптер с минимальным разрешением 144p.

Минимальный состав программных средств

Дополнительные программы не требуются

Требования к персоналу (оператору)

Элементарные умения работы с компьютером.

3. Выполнение программы

Загрузка и запуск программы

Программа состоит из трёх файлов: "Bank helper.exe", "in.txt", "passfile.txt", "commission.txt". В текстовых файлах находятся: данные о валютах и курсах, пароль и процент комиссии. Все эти файлы должны находиться в одном каталоге для корректной работы.

При запуске программы должно открыться следующее окно:

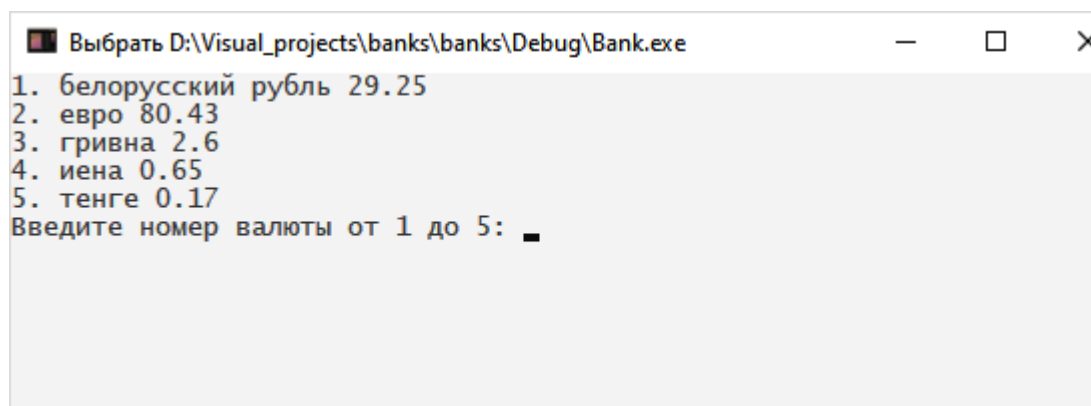


Рисунок 18. Главное окно программы

Выполнение программы

Для доступа к режиму редактирования данных нужно нажать клавишу '0'. Появится строка, сообщающая, что нужно ввести пароль для входа в режим редактирования (по умолчанию '0'). Оператору даётся три попытки ввести правильный пароль.

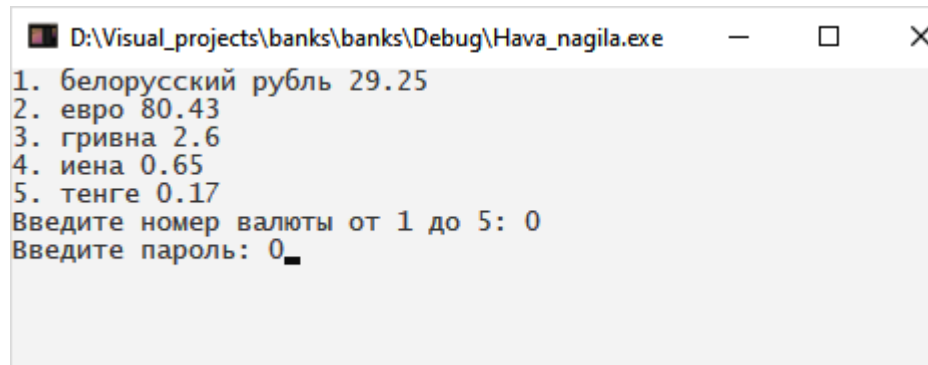


Рисунок 19. Окно ввода пароля

Если пароль введён верно, то откроется окно режима редактирования данных

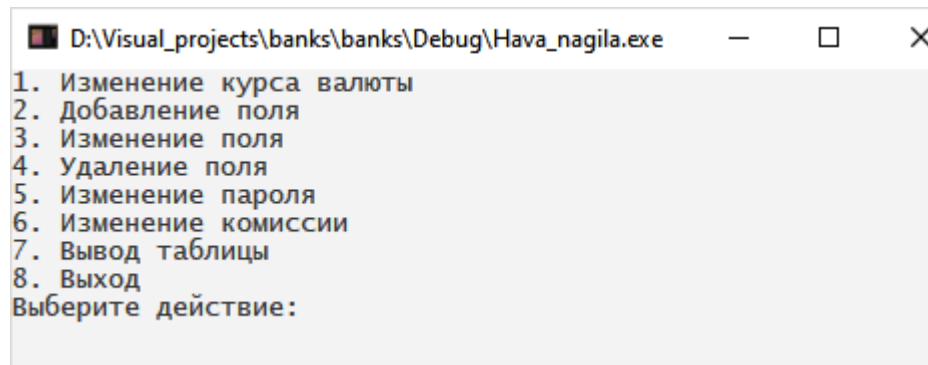


Рисунок 20. Окно режима редактирования

Далее будут рассмотрены функции, которые может использовать оператор. Для использования функции, нужно ввести соответствующую ей цифру.

1. Изменение курса валюты

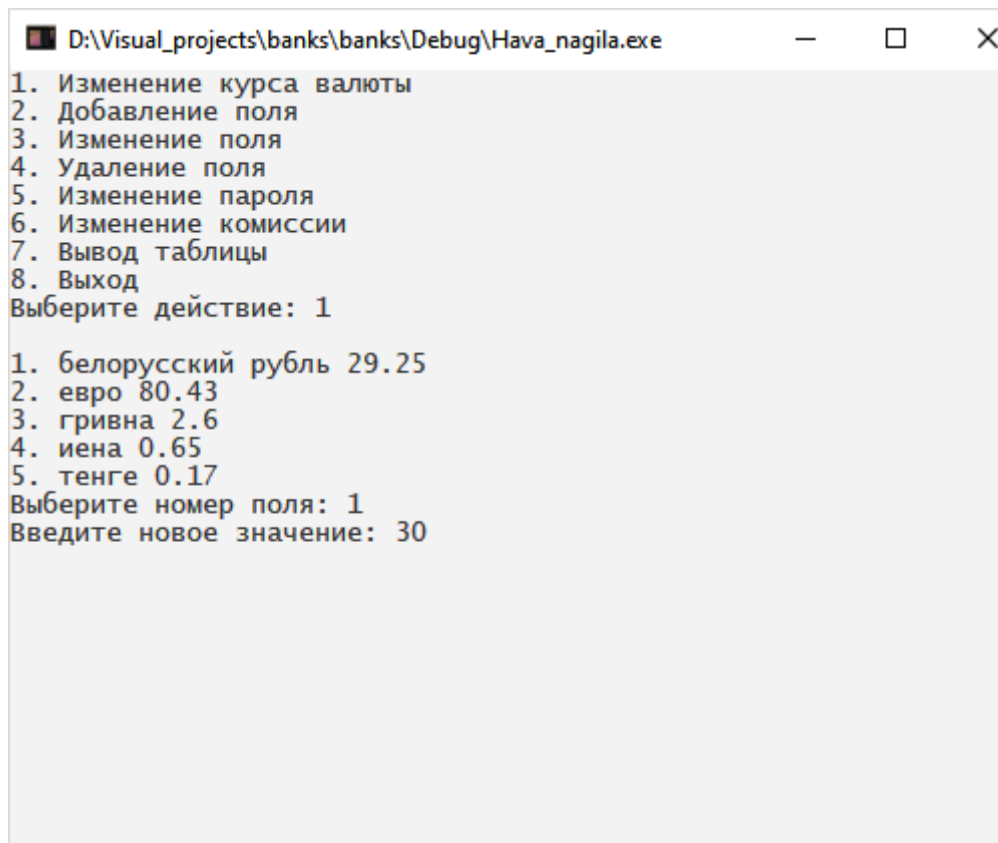


Рисунок 21. Функция изменение курса валюты с примером входных данных

2. Добавления поля

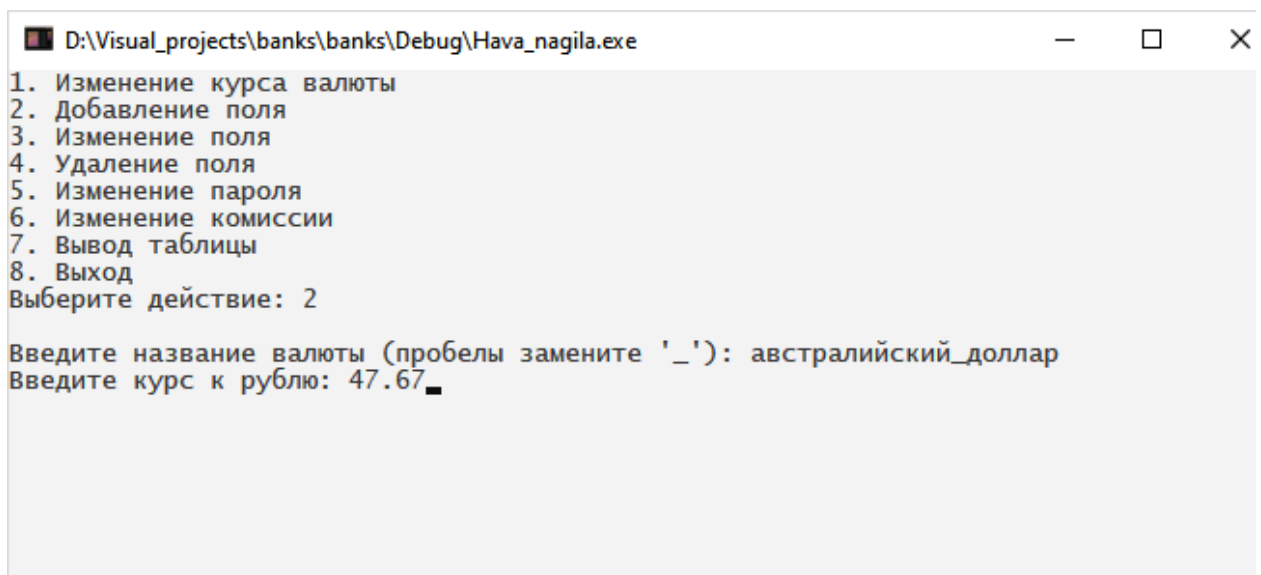


Рисунок 22. Функция добавления поля с примером входных данных

3. Изменения поля

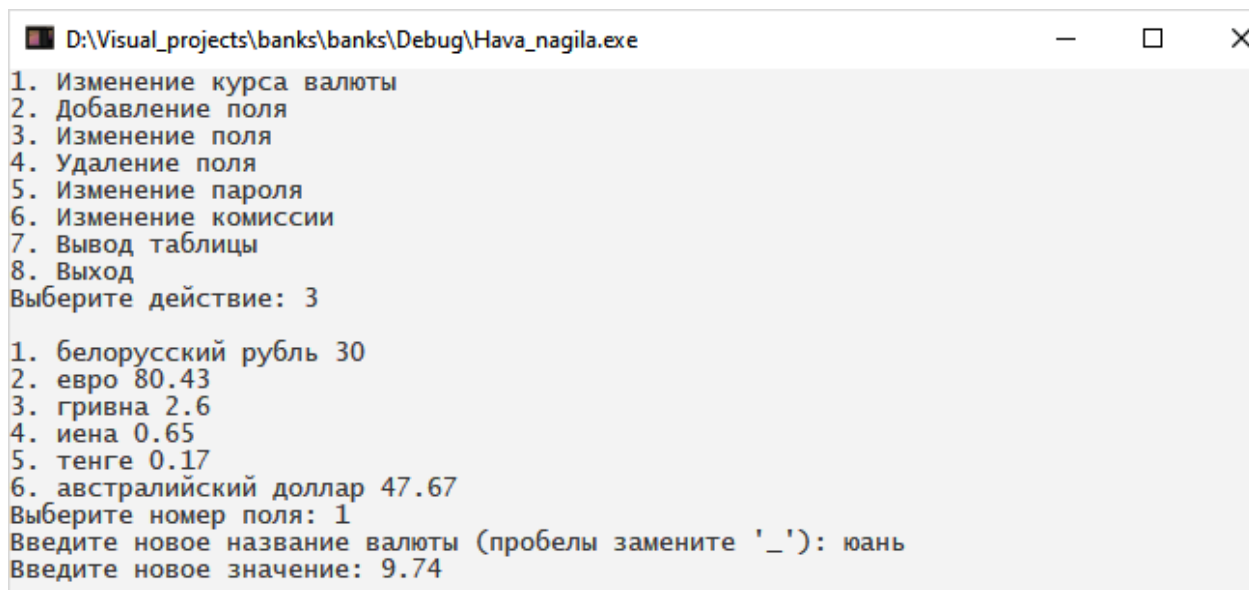


Рисунок 23. Функция изменения поля с примером входных данных

4. Удаление поля



Рисунок 24. Функция удаления поля с примером входных данных

5. Изменение пароля

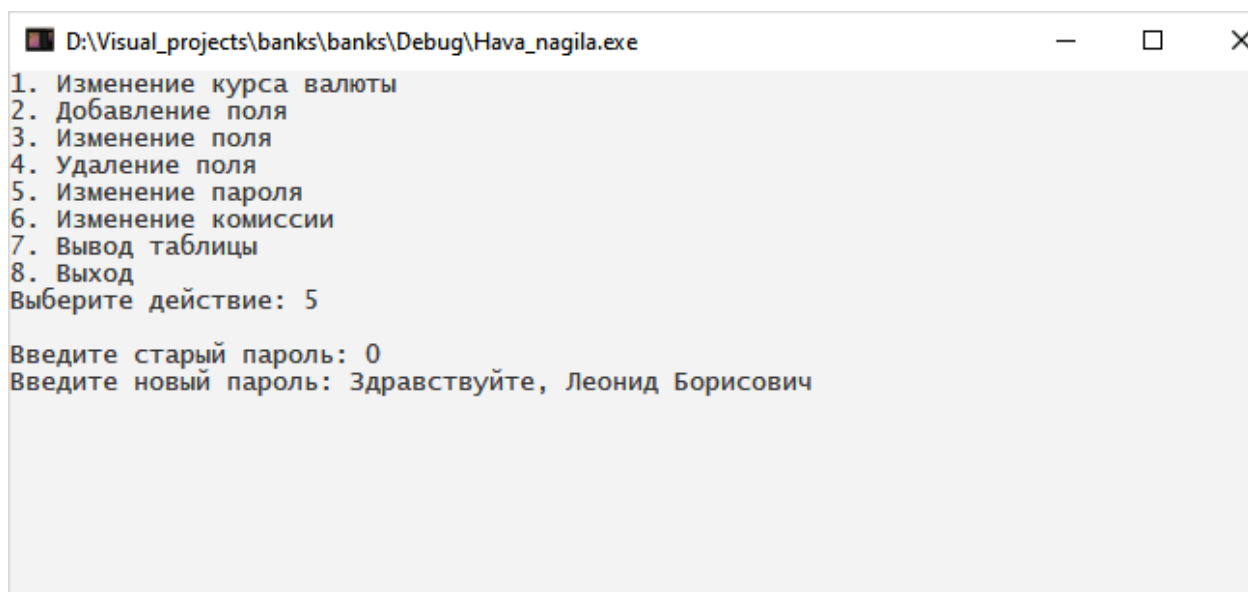


Рисунок 25. Функция изменения пароля с примером входных данных

6. Изменение комиссии

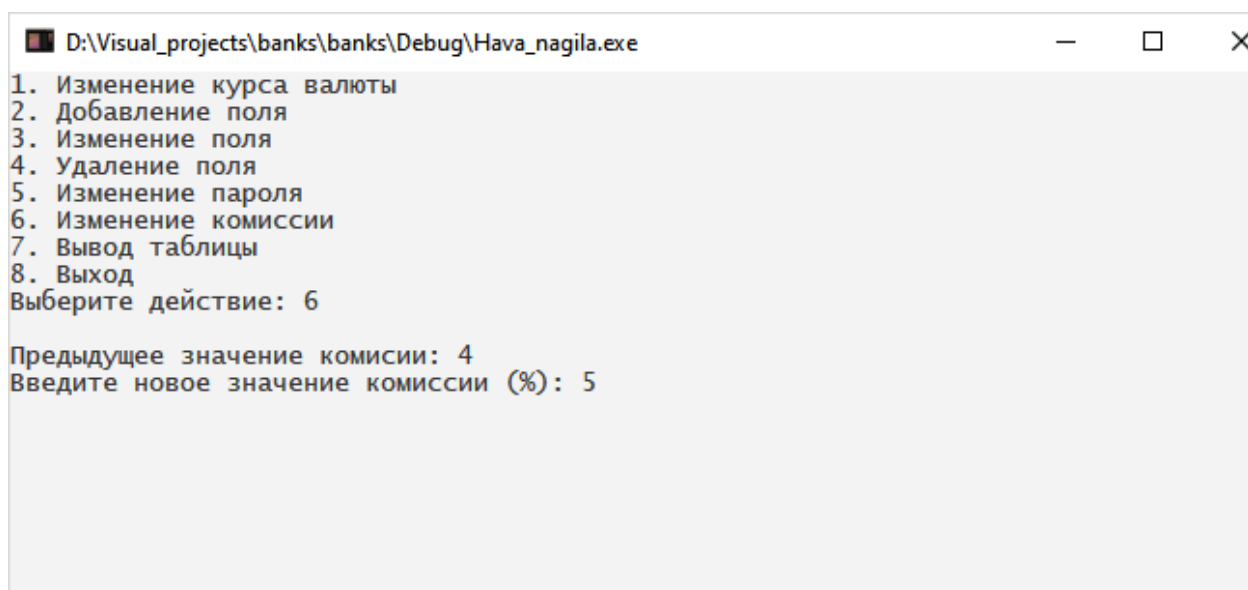


Рисунок 26. Функция изменения комиссии с примером входных данных

7. Вывод таблицы на экран

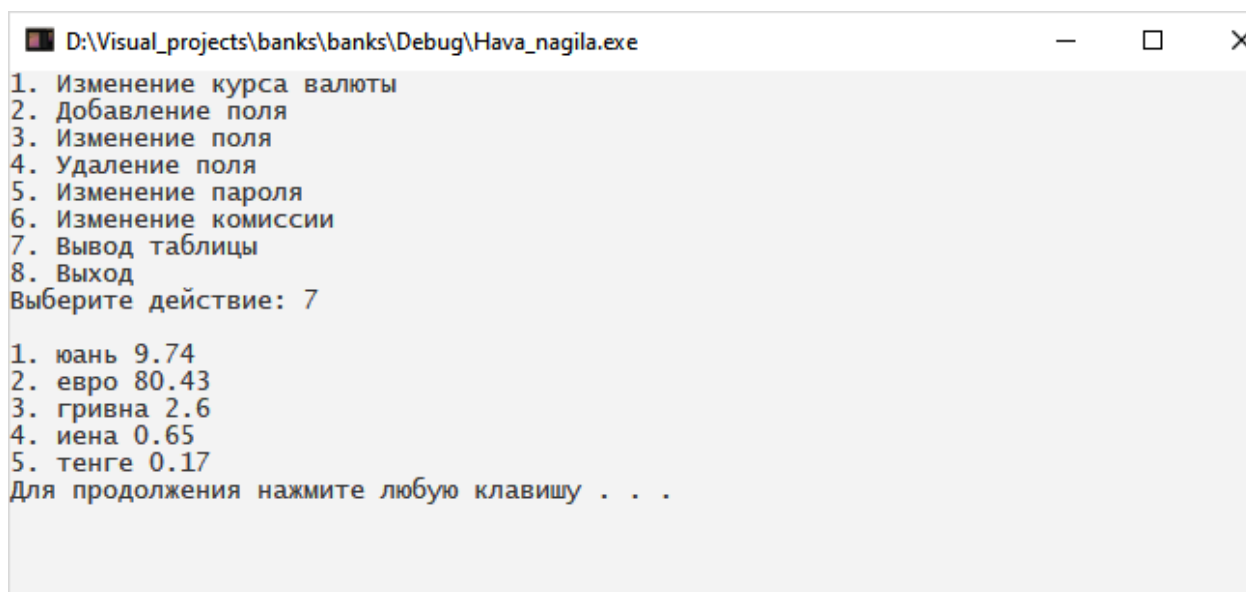


Рисунок 27. Функция вывода таблицы на экран с примером входных данных

8. Завершение работы программы

Далее будут рассмотрен порядок действий клиента.

1. При запуске откроется следующее окно. В нём клиенту будет предложено выбрать валюту для покупки из списка.

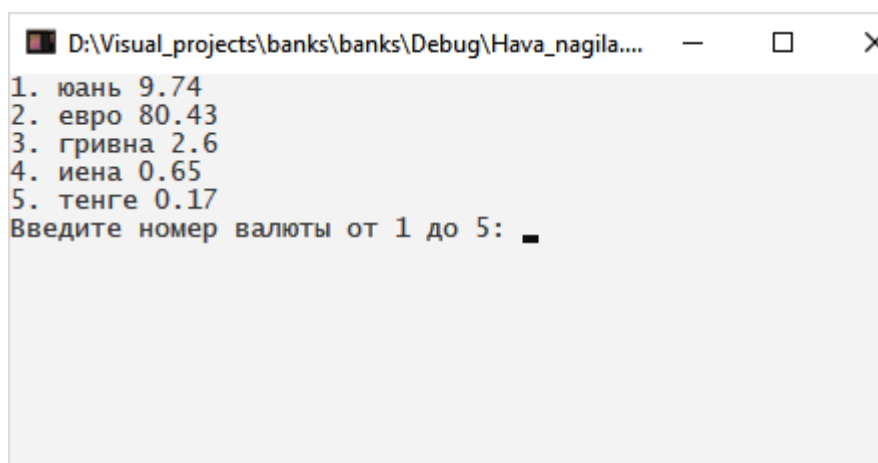


Рисунок 28. Окно выбора валюты

2. После выбора валюты, пользователю будет предложено выбрать необходимое количество валюты.

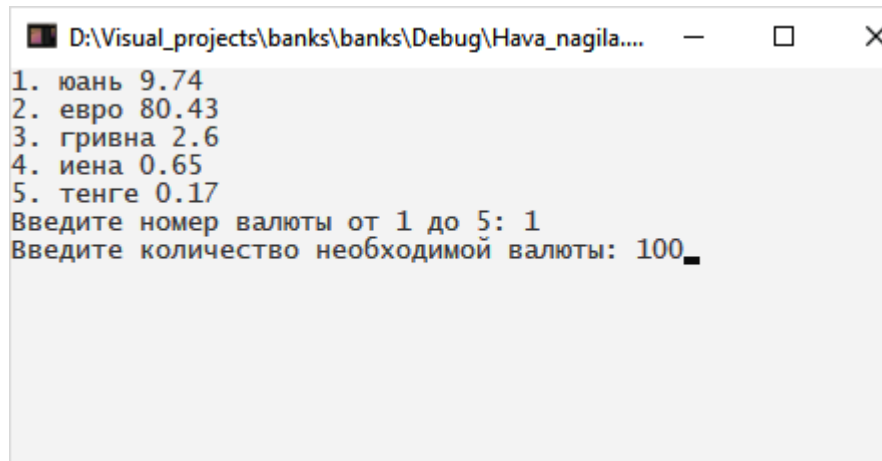


Рисунок 29. Окно выбора количества валюты

3. Затем пользователю необходимо ввести свои паспортные данные

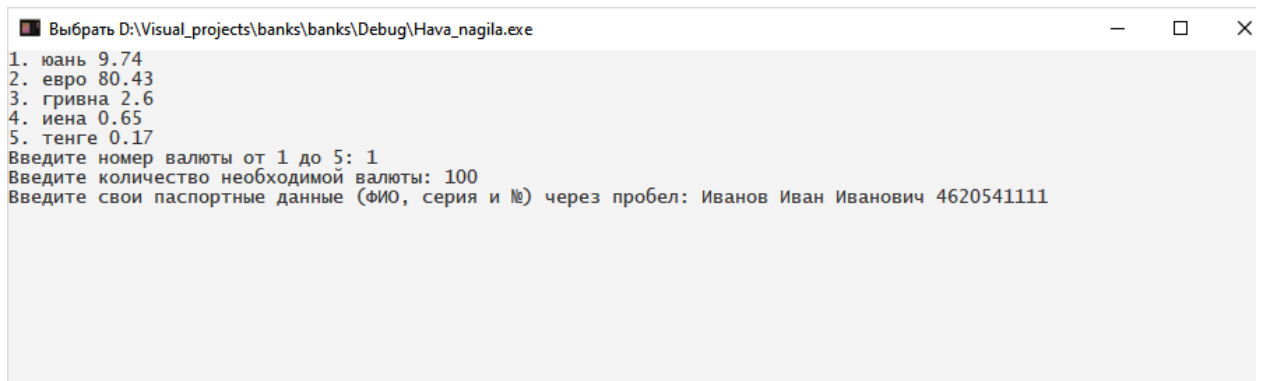


Рисунок 30. Ввод паспортных данных клиента

4. После совершённых действий, клиент получит чек с информацией об операции. При нажатии любой клавиши происходит выход из программы.

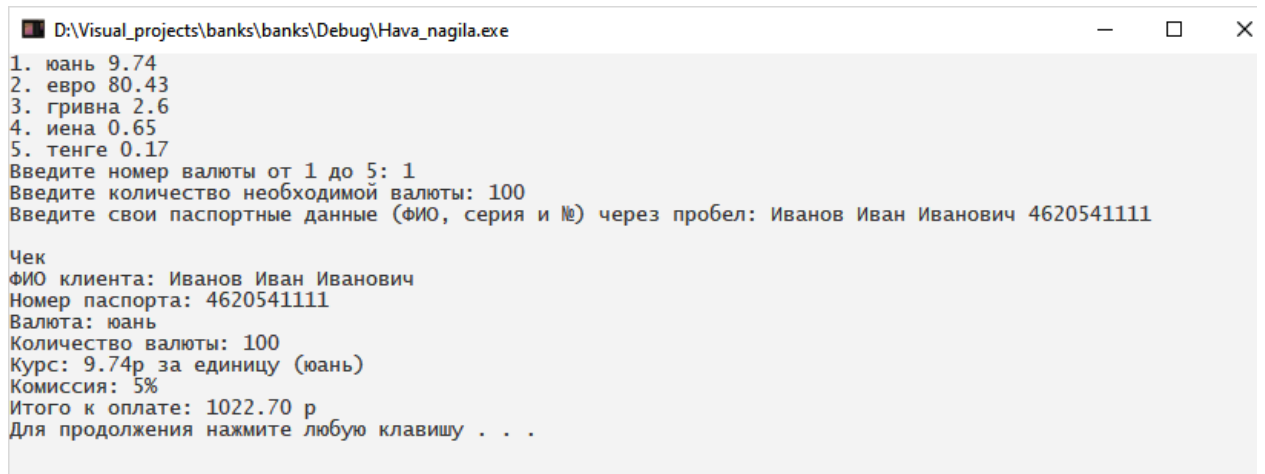


Рисунок 31. Чек

Заключение

В результате выполнения курсового проекта была написана программа "Bank helper" для автоматизации процесса покупки валюты.

В ходе работы были проанализированы предметная область, существующие разработки, посвященные данному направлению, получены практические навыки с работой в Microsoft Visual Studio.

Так же планируется продолжить работу над данным проектом с целью расширения возможностей и удобства для пользователей. Планы по доработке представлены ниже.

1. Реализация методов и функций, учитывающих количество оставшихся в наличии валют.
2. Создание и последующая доработка интерфейса.
3. Исправление багов
4. Повышение уровня безопасности, а именно добавление шифрования пароля.

Список литературы и интернет-источников

1. Сайт компании "ЮниСаб":

<http://www.unisab.ru>

2. Сайт представителя "Центавр-Дельта":

http://www.absonline.ru/software/native/centavr_d/

3. Теоретические материалы:

https://studopedia.su/11_101121_valyutniy-obmenniy-punkt.html

https://studbooks.net/1237028/bankovskoe_delo/poryadok_organizatsii_raboty_obmennogo_punkta

<https://infopedia.su/6xa2fa.html>

Приложение 1. Код главного модуля bank.cpp

```
#include "functions.h"
#include "Person.h"

extern vector <Money> dollar;

int main()
{
    system("chcp 1251");
    system("cls");
    double commission = get_commission();
    int val;
    Person jd;
    getff();
    print();

    if (dollar.size() < 1) //проверка на отсутствие данных в бд
    {
        cout << "Ассортимент пуст, обратитесь к менеджеру" << endl;
        cin >> val;
        if (val == 0)
            worker();
    }

    else
    {
        val = choose_your_destiny();
        if (val == 0)
        {
            worker();
        }
        else
        {
            double amount;
            cout << "Введите количество необходимой валюты: ";
            cin >> amount;
            jd.p_getpas();
        }
    }
}
```

```
        print_bill(jd, calculate_sum(amount, dollar[val-1].m_value,
commission),
                    commission, amount, val);
    }
}
system("pause");
return 0;
}
```


Приложение 2. Код модуля Person.h

```
#pragma once
#include <iomanip>
#include <vector>
#include "Money.h"

using namespace std;

class Person // класс для паспортных данных пользователя
{
public:
    string p_f; // фамилия
    string p_i; // имя
    string p_o; // отчество
    string p_num; // номер паспорта

    void p_getpas() // метод получения паспортных данных клиента
    {
        cout << "Введите свои паспортные данные (ФИО, серия и №) через пробел: ";
        cin >> p_f >> p_i >> p_o >> p_num;
    }

    void p_print_fio() // вывод ФИО
    {
        cout << p_f << " " << p_i << " " << p_o;
    }

    void p_print_num() // вывод номера паспорта
    {
        cout << p_num;
    }
};
```

Приложение 3. Код модуля Money.h

```
#pragma once
#include <iostream>

using namespace std;

class Money // класс для валюты
{
public:
    string m_name; // название валюты
    double m_value; // курс валюты к рублю

    Money(string name, double value)
    {
        this->m_name = name;
        this->m_value = value;
    }

    void m_print_name() // вывод названия с учётом '_'
    {
        for (int i = 0; i < m_name.length(); i++)
        {
            if (m_name[i] == '_')
                cout << " ";
            else
                cout << m_name[i];
        }
    }

    void m_print_value() // вывод курса
    {
        cout << m_value;
    }
};
```

Приложение 4. Код модуля constants.cpp

```
#include <string>
#include <vector>
#include "Money.h"

using namespace std;

const string filename = "in.txt"; // файл с данными БД
const string pasFileName = "passfile.txt"; // файл с паролем
const string commissionFileName = "commission.txt"; // файл с
комиссией
vector <Money> dollar; //вектор с данными БД
```

Приложение 5. Код заголовочного файла functions.h

```
#pragma once
extern class Person;
void getff(); //данные из файла в вектор
void print(); //вывод таблицы с данными на экран
void vector_to_file(); //запись таблицы с данными в файл "filename"
double get_commission(); //получение комиссии из файла
"commissionFileName"
int choose_your_destiny(); //функция выбора номера валюты
void add_field(); // добавление поля
void edit_field(); // изменение курса валюты у поля
void delete_field(); //удаление поля
void change_field(); // изменение поля
void change_password(); //смена пароля
void change_commission(); //изменение комиссии
void worker_menu(); // меню для работника
void worker(); // проверка пароля и доступ к режиму редактирования
данных
long double calculate_sum(double coins, double value, double
commission); //вычисление итогового счёта
void clear_buf(); //очистка буфера ввода
void print_bill(Person &jd, long double total, double commission,
double amount, int val); // печать чека
```

Приложение 6. Код модуля functions.cpp

```
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include "Money.h"
#include "Person.h"

using namespace std;

extern const string filename = "in.txt";
extern const string pasFileName = "passfile.txt";
extern const string commissionFileName = "commission.txt";
extern vector <Money> dollar;

void clear_buf() // очистка буфера ввода
{
    cin.clear(); //очистка
    cin.ignore(32767, '\n'); //игнорирование до enter
}

void getff() //данные из файла в вектор
{
    string name;
    double value;
    ifstream fin(filename);

    if (!fin.is_open())
    {
        cout << "Файл с валютой не найден" << endl;
        return;
    }

    while (fin >> name >> value)
        dollar.push_back(Money(name, value));
    fin.close();
}
```

```
void print() //вывод таблицы с данными на экран
```

```
{
    for (int i = 0; i < dollar.size(); i++)
    {
        cout << i + 1 << ". ";
        dollar[i].m_print_name();
        cout << " ";
        dollar[i].m_print_value();
        cout << endl;
    }
}
```

```
void vector_to_file() //запись таблицы с данными в файл "filename"
```

```
{
    ofstream fin(filename);
    for (int i = 0; i < dollar.size(); i++)
        fin << dollar[i].m_name << " " << dollar[i].m_value << endl;
    fin.close();
}
```

```
double get_commission() //получение комиссии из файла
```

```
"commissionFileName"
{
    double commission;
    ifstream fin(commissionFileName);
    if (!fin.is_open())
    {
        cout << "Файл с комиссией не найден" << endl;
        return 0;
    }
    fin >> commission;
    fin.close();
    return commission;
}
```

```
int choose_your_destiny() //функция выбора номера валюты
```

```
{
```

```

    int num;
    while (1)
    {
        if (dollar.size() < 2)
            cout << "Введите номер валюты: ";
        else
            cout << "Введите номер валюты от 1 до " << dollar.size()
<< ": ";
        cin >> num;
        if (num <= dollar.size() && num >= 0)
            break;
    }
    return num;
}

void add_field() // добавление поля
{
    clear_buf();
    cout << "Введите название валюты (пробелы замените '_'): ";
    string name;
    double value;
    cin.sync(); // синхронизация потоков (почитать или попробовать с
разбиением одной строки)
    getline(cin, name);
    cout << "Введите курс к рублю: ";
    cin >> value;
    dollar.push_back(Money(name, value));
    vector_to_file();
}

void edit_field() // изменение курса поля
{
    int num;
    double value;
    print();
    cout << "Выберите номер поля: ";
    cin >> num;
    cout << "Введите новое значение: ";

```

```

        cin >> value;
        dollar[num - 1].m_value = value;
        vector_to_file();
    }

void delete_field() //удаление поля
{
    int num;
    print();
    cout << "Выберите номер поля: ";
    cin >> num;
    dollar.erase(dollar.begin() + num - 1);
    vector_to_file();
}

void change_field() // изменение поля
{
    int num;
    string name;
    double value;
    print();
    cout << "Выберите номер поля: ";
    cin >> num;
    cout << "Введите новое название валюты (пробелы замените '_'): ";
    cin >> name;
    cout << "Введите новое значение: ";
    cin >> value;
    dollar[num - 1].m_name = name;
    dollar[num - 1].m_value = value;
    vector_to_file();
}

void change_password() //смена пароля
{
    clear_buf();
    int tries_not_null = 4;
    string password, filePassword;
    ifstream file(pasFileName);

```



```

cout << "Введите старый пароль: ";
getline(file, filePassword);
while (tries_not_null) //цикл для проверки правильности пароля
{
    getline(cin, password);
    if (password == filePassword)
    {
        file.close();
        ofstream file(pasFileName);
        cout << "Введите новый пароль: ";
        getline(cin, filePassword);
        file << filePassword;
        break;
    }
    else
    {
        tries_not_null--;
        if (tries_not_null != 0)
            cout << "Неправильный пароль, попробуйте ещё раз: ";
        else
        {
            cout << "Не пущу, пока не вспомнишь пароль" << endl;
            return;
        }
    }
}
file.close();
}

void change_commission() //изменение комиссии
{
    double commission;
    cout << "Предыдущее значение комиссии: " << get_commission() <<
endl;;
    cout << "Введите новое значение комиссии (%): ";
    cin >> commission;
    ofstream fout(commissionFileName);
    fout << commission;
}

```

```

    fout.close();
}

void worker_menu() // меню для работника
{
    int num = 0;

    while (num != 8)
    {
        system("cls");
        cout << "1. Изменение курса валюты" << endl;
        cout << "2. Добавление поля" << endl;
        cout << "3. Изменение поля" << endl;
        cout << "4. Удаление поля" << endl;
        cout << "5. Изменение пароля" << endl;
        cout << "6. Изменение комиссии" << endl;
        cout << "7. Вывод таблицы" << endl;
        cout << "8. Выход" << endl;
        cout << "Выберите действие: ";
        cin >> num;
        cout << endl;
        switch (num)
        {
            case 1:
                edit_field();
                break;
            case 2:
                add_field();
                break;
            case 3:
                change_field();
                break;
            case 4:
                delete_field();
                break;
            case 5:
                change_password();
                break;

```

```

        case 6:
            change_commission();
            break;
        case 7:
        {
            print();
            system("pause");
            break;
        }

    }
}

void worker() // проверка пароля и доступ к режиму редактирования
данных
{
    clear_buf();
    int tries_not_null = 4;
    string password, filePassword;
    ifstream file("passfile.txt");
    if (!file.is_open())
    {
        cout << "Файл с паролем не найден" << endl;
        return;
    }
    getline(file, filePassword);
    cout << "Введите пароль: ";
    while (tries_not_null)
    {
        getline(cin, password);
        if (filePassword == password)
        {
            worker_menu();
            break;
        }
        else
        {

```

```

        tries_not_null--;
        if (tries_not_null != 0)
            cout << "Неправильный пароль, попробуйте ещё раз: ";
        else
            cout << "Не пушу, пока не вспомнишь пароль" << endl;
    }
}
file.close();
}

```

```

long double calculate_sum(double coins, double value, double
commission) //вычисление итогового счёта
{
    double total;
    total = coins * value + (coins * value / 100 * commission);
    return total;
}

```

```

void print_bill(Person &jd, long double total, double commission,
double amount, int val) // печать квитанции
{
    cout << "\nЧек";
    cout << "\nФИО клиента: ";
    jd.p_print_fio();

    cout << "\nНомер паспорта: ";
    jd.p_print_num();

    cout << "\nВалюта: ";
    dollar[val - 1].m_print_name();

    cout << "\nКоличество валюты: " << amount;

    cout << "\nКурс: ";
    dollar[val - 1].m_print_value();
    cout << "р за единицу (";
    dollar[val - 1].m_print_name();
    cout << ")";
}

```

```
    cout << "\nКомиссия: " << commission << "%";  
    cout << fixed << setprecision(2) << "\nИтого к оплате: " << total  
<< " p";  
  
    cout << endl;  
}
```