

КУРСОВОЙ ПРОЕКТ

По дисциплине: МДК 01.01 Разработка программных модулей

Тема: Разработка системы классов для приложения «Обменный пункт валюты»

Специальность 09.02.07 «Информационные системы и программирование»

|  |  |  |
| --- | --- | --- |
| **Выполнил студент(ка) группы 302ИС-22** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Д.И.Романов** |
| **Руководитель** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Л.Б. Гусятинер** |

Москва 2024



УТВЕРЖДАЮ

Зам. директора КМПО

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_С.Ф. Гасанов

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ

По дисциплине: МДК 01.01 Разработка программных модулей

Специальность 09.02.07 «Информационные системы и программирование»

Студент группы 302ИС-22 Романов Дмитрий

ТЕМА: Разработка системы классов для приложения «Обменный пункт валюты»

Дата выдачи задания «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_ 2024 г.

Срок сдачи проекта «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 г.

Москва 2024

**Содержание**

Введение4

1.Теоретическая часть 4

1.1Описание предметной области 5

1.2. Описание существующих разработок 6

DORS 12507

Hitachi iH-110 8

2.Проектная часть9

2.1.Диаграмма прецедентов9

2.2. Выбор инструментов9

2.3. Проектирование сценария11

2.4. Диаграмма классов12

2.5Описание главного модуля14

2.6 Описание тестовых наборов модулей18

3. Эксплуатационная часть 20

3.1. Руководство оператора20

Заключение23

Список литературы и интернет-источников24

Приложение 1.Код программы 25

Приложение 2. Результаты тестирования32

Приложение 3. Диаграмма прецедентов.34

1.Теоретическая часть

Введение

Целью данной курсовой работы является разработка системы классов для приложения "Обменный пункт валюты", которая позволит автоматизировать процессы работы обменного пункта и улучшить обслуживание клиентов. В современном мире обмен валюты играет важную роль, и подобные приложения становятся все более востребованными. Разработка данной системы классов предполагает создание удобного интерфейса для работы операторов обменного пункта, хранение информации о текущих курсах валют, автоматизацию расчетов и другие функции, направленные на улучшение процесса обмена валюты. В первой части работы будет рассмотрена предметная область обменных пунктов, а также анализ существующих приложений данной тематики. Во второй части будет подробно описана структура разрабатываемой системы классов, ее основные функции и взаимодействие между классами. В третьей части предоставлено будет руководство пользователя для работы с разработанной системой классов. В заключительной части приведены будут общие выводы о созданной системе и планы по ее дальнейшему развитию.

1.1. Описание предметной области

Обменный пункт валюты - это учреждение, в котором осуществляется обмен различных видов национальной и иностранной валюты. В таком заведении можно обменять деньги одной страны на деньги другой страны по текущему обменному курсу. Обменные пункты обычно располагаются в туристических районах, аэропортах, железнодорожных вокзалах, торговых центрах и других местах с высокой проходимостью.

В обменном пункте доступны различные виды услуг, такие как покупка и продажа валюты, обмен туристических чеков, выдача наличных денег и т.д. Сотрудники обменного пункта должны быть профессиональными специалистами в области финансов, иметь хорошее знание обменных курсов и процедур обмена.

Для обменного пункта валюты важно соблюдать законы и правила регулирования деятельности обмена валюты, так как это может повлиять на легальность и надежность операций. Также важно обеспечить безопасность в обменном пункте, чтобы избежать мошенничества и краж.

Успешное функционирование обменного пункта зависит от хорошей репутации у клиентов, выгодных обменных курсов, удобного расположения и высокого уровня обслуживания.

1.2. Описание существующих разработок Для обменных пунктов валюты также существует несколько различных разработок, которые помогают им эффективно управлять своей деятельностью и обеспечивать качественное обслуживание клиентов.

Одной из таких разработок является специализированное программное обеспечение для учета и анализа операций обмена валюты. Это позволяет автоматизировать процессы учета и отчетности, а также повысить безопасность и контроль за финансовыми операциями.

Другим важным инструментом для обменных пунктов является онлайн-платформа для обмена валюты, которая позволяет клиентам быстро и удобно проводить операции без необходимости посещения офиса. Такие платформы также обеспечивают защищенную передачу данных и удобный интерфейс для пользователей.

Кроме того, некоторые обменные пункты используют специализированные терминалы для проведения операций, что облегчает процесс обмена и сокращает время ожидания для клиентов.

Все эти разработки помогают обменному пункту улучшить качество обслуживания, повысить эффективность работы и обеспечить конкурентные преимущества на рынке.

DORS 1250

DORS 1250 - это универсальное устройство для проверки подлинности банкнот разных валют и другой защищенной полиграфической продукции.

Преимущества DORS1250

Мощность: DORS 1250 обладает высокой мощностью, что позволяет эффективно выполнять большую часть рабочих задач.

Производительность: благодаря своей мощности и функциональности, DORS 1250 обеспечивает высокую производительность и эффективность работы.

Надежность: данный инструмент изготовлен из качественных материалов, что обеспечивает его надежность, долговечность и стабильность работы.

Удобство использования: DORS 1250 имеет удобный и эргономичный дизайн, что делает его удобным в использовании и уменьшает усталость оператора.

Многофункциональность: этот инструмент имеет множество функций и возможностей, что позволяет использовать его для различных задач и операций.

Безопасность: DORS 1250 оборудован различными системами безопасности, что делает его безопасным в использовании и защищает оператора от возможных травм или аварийных ситуаций.

Hitachi iH-110

Hitachi iH-110 - это компактное устройство, предназначенное для быстрого и точного подсчета денежных купюр. Он оснащен современной технологией, которая позволяет быстро и надежно подсчитывать купюры различных валют, а также обнаруживать поддельные банкноты. Hitachi Ih

110 также обладает функцией сортировки купюр по номиналу и удобным дисплеем для отображения результатов подсчета.

Преимущества Hitachi iH-110:

Высокая скорость и точность подсчета банкнот - до 1000 банкнот в минуту.

Широкий диапазон распознаваемых валют - поддерживает более 10 валют.

Надежность и долговечность - изготовлен из прочных материалов, что обеспечивает длительный срок службы.

Простота использования - интуитивный интерфейс и понятное меню делают работу с устройством удобной и эффективной.

Функция детекции поддельных денег - автоматически обнаруживает поддельные купюры и предотвращает их принятие.

Компактный размер и легкий вес - удобно переносить и использовать в различных условиях.

2.Проектная часть

2.1.Диаграмма прецедентов

Приложение 3. Рисунок 6

2.2. Подбор оборудования

При проведении выбора инструментов произведено сопоставление по критериям, перечисленным в таблице 1. Оценка важности критерия производилась на основе следующих уровней: низкий, ниже среднего, средний, выше среднего, высокий.

Таблица 1. Критерии выбора инструмента.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| критерий | Участие в корпоративном проекте | Простота сопровождения | Наличие библиоте | Наличие документации на русском языке | Скорость разработки |
| Важность критерия | очень высокая | высокая | очень высокая | очень высокая | высокая |

На основе данных критериев, я провел сравнение трех языков программирования, оценивая каждый критерий по шкале от 0 до 10 баллов.

Таблица 2. Оценка языков программирования.

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий Язык программирования | Python | C++ | C# |
| Участие в корпоративном проекте | 7 | **3** | 5 |
| Простота сопровождения | 10 | **7** | 10 |
| Наличие библиотек | **8** | **7** | 8 |
| Наличие документации на русском языке | 6 | 8 | 6 |
| Скорость разработки | 10 | **5** | 9 |
| Итого баллов | 41 | 30 | 38 |

После анализа было принято решение использовать Python в качестве основного языка программирования.

2.3. Проектирование сценария

В данном разделе приведен сценарий использования программы

Запуск программы

Ввести номер счета

Меню

Обмена

Ваш баланс:

Добро пожаловать!

USD на SEK

EUR на USD

RUB на USD

USD на

EUR

Введите сумму в USD

Введите сумму в JPY

Введите сумму в SEK

Введите сумму в USD

Введите сумму в EUR

Введите сумму в USD

Введите сумму в RUB

USD на JPY

JPY на USD

SEK на USD

Вы получите … USD

Вы получите … JPY

Вы получите … USD

Вы получите … SEK

Вы получите … USD

Вы получите … EUR

Вы получите … USD

2.4. Диаграмма классов

Данный раздел содержит обзор всех классов, применяемых в данном проекте.



Класс Currency

Содержит поля name и rate.

Используется для создания валюты.

Класс Wallet

Имеет поля currency и amount.

Используется для создания кошелька с валютой и определенной суммой.

Класс Exchange

Имеет поле currencies.

Используется для добавления валюты и обмена валюты на кошельке.

Класс CurrencyExchange

Имеет поле convert

Используется для обмена различных валют по заданным правилам.

2.5Описание главного модуля

В главный модуль импортируeтся библиотека : CyrrencyConverter.

Главный модуль является точкой входа в программу,в нем релизованы вызовы всех функций,открытие базы данных и логика меню,с помощью которого пользователь может взаимодействовать с программой.

bank\_number = input("Введите номер вашего счета ")  
  
if bank\_number == '20102006' :  
  
 print('Добро пожаловать!')  
  
else:  
 print('Неправильно набран номер')  
  
print("Ваш баланс (USD):", wallet.amount)  
  
  
  
  
print('Меню возможных действий:')  
  
  
print("1. Обмен RUB на USD")  
  
print("2. Обмен USD на EUR")  
  
print("3. Обмен EUR на USD")  
  
print("4. Обмен USD на SEK")  
  
print("5. Обмен SEK на USD")  
  
print("6. Обмен USD на JPY")  
  
print("7. Обмен JPY на USD")  
  
  
choice = input("Выберите операцию: ")  
  
exchange = CurrencyExchange(cur)  
  
if choice == '1':  
  
 rub\_amount = float(input("Введите сумму в RUB: "))  
  
 usd\_amount = cur.convert(rub\_amount, 'RUB', 'USD')  
  
 print(f"Вы получите {usd\_amount} USD")  
  
elif choice == '2':  
  
 usd\_amount = float(input("Введите сумму в USD: "))  
  
 eur\_amount = cur.convert(usd\_amount, 'USD', 'EUR')  
  
 print(f"Вы получите {eur\_amount} EUR")  
  
  
elif choice == '3':  
  
 eur\_amount = float(input("Введите сумму в EUR: "))  
  
 usd\_amount = cur.convert(eur\_amount, 'EUR', 'USD')  
  
 print(f"Вы получите {usd\_amount} USD")  
  
elif choice == '4':  
  
 usd\_amount = float(input("Введите сумму в USD: "))  
  
 sek\_amount = cur.convert(usd\_amount, 'USD', 'SEK')  
  
 print(f"Вы получите {sek\_amount} SEK")  
  
elif choice == '5':  
  
 sek\_amount = float(input("Введите сумму в SEK: "))  
  
 usd\_amount = cur.convert(sek\_amount, 'SEK', 'USD')  
  
 print(f"Вы получите {usd\_amount} USD")  
  
elif choice == '6':  
  
 usd\_amount = float(input("Введите сумму в USD: "))  
  
 jpy\_amount = cur.convert(usd\_amount, 'USD', 'JPY')  
  
 print(f"Вы получите {jpy\_amount} JPY")  
  
elif choice == '7':  
  
 jpy\_amount = float(input("Введите сумму в JPY: "))  
  
 usd\_amount = cur.convert(jpy\_amount, 'JPY', 'USD')  
  
 print(f"Вы получите {usd\_amount} USD")  
  
  
else:  
  
 print("Неправильный выбор. Пожалуйста, выберите 1, 2, 3, 4, 5, 6 или 7")

2.6 Описание тестовых наборов модулей

В этом разделе будут продемонстрированы результаты тестирования «черного ящика».

Тест 1

Цель: Проверить приветствие пользователя.

Тестовые данные:

Ввод номера: 20102006.

Ожидаемый результат: Сообщение "Добро пожаловать!".

Результат теста: Приложение 2 рисунок 1

Тест 2

Цель: Проверить правильность введенного номера.

Тестовые данные:

Ввод номера: 20.

Ожидаемый результат: Сообщение "Неправильный номер!".

Результат теста: Приложение 2 рисунок 2

Тест 3

Цель: Проверить правильность вывода баланса и меню операций на экран.

Тестовые данные:

Ввод номера: 20102006.

Ожидаемый результат: Сообщение " Добро пожаловать!

Ваш баланс (USD): 100000

Меню возможных действий:

Выберите операцию:".

Результат теста: Приложение 2 рисунок 3

Тест 4

Цель: Проверить операцию обмена RUB на USD.

Тестовые данные:

Ввод операции:1

Ввод суммы RUB: 1000

Ожидаемый результат: Сообщение " Вы получите 9.523809523809526 USD".

Результат теста: Приложение 2 рисунок 4

Тест 5

Цель: Проверить операцию обмена JPY на USD.

Тестовые данные: Приложение 2 рисунок 5

Ввод операции:7

Ввод суммы JPY: 1600

Ожидаемый результат: Сообщение "Вы получите 10.517293233082707 USD".

Результат теста: Приложение 2 рисунок 5

3. Эксплуатационная часть

3.1. Руководство оператора

АННОТАЦИЯ

В данном программном документе приведено руководство оператора по применению и эксплуатации программы.

В данном программном документе, в разделе «Назначение программы» указаны сведения о назначении программы и информация, достаточная для понимания функций программы и ее эксплуатации. В разделе «Условия выполнения программы» указаны условия, необходимые для выполнения программы (минимальный состав аппаратных и программных средств и т.п.). В данном программном документе, в разделе «Выполнение программы» указана последовательность действий оператора, обеспечивающих загрузку, запуск, выполнение и завершение программы, приведено описание функций, формата и возможных вариантов команд, с помощью которых оператор осуществляет загрузку и управляет выполнением программы, а также ответы программы на эти команды. Оформление программного документа «Руководство оператора» произведено по требованиям

Назначение программы 1.1. Функциональное назначение программы Специальное программное обеспечение используется для обмена разной валюты .

1.2. Эксплуатационное назначение программы Специальное программное обеспечение может эксплуатироваться на объектах любого масштаба в сфере банковского дела для облегчения работы персонала.

* 1. Состав функций

1.3.1 Функция ввода номера банковского счета.

Эта функция позволяет пользователю войти в систему под номером своего банковского счета.

1.3.2 Функция проверки правильности номера банковского счета

эта функция позволяет пользователю проверить правильность введенного им номера банковского счета.

1.3.3 Функция проверки баланса

эта функция позволяет пользователю проверить свой баланс

1.3.4 Функция демонстрации меню

эта функция позволяет пользователю увидеть меню возможных операций.

1.3.5 Функция обмена валюты

Эта функция позволяет пользователю выбрать и обменять нужную ему валюту.

2. Условия выполнения программы

2.1. Минимальный состав аппаратных средств ОС: Windows 10 Процессор: Как минимум 1 ГГц или SoC. ОЗУ: 1 ГБ (для 32-разрядных систем) или 2 ГБ (для 64-разрядных систем). 34 Место на жестком диске: 16 ГБ (для 32-разрядных систем) или 20 ГБ (для 64-разрядных систем). Видеоадаптер: DirectX версии не ниже 9 с драйвером WDDM 1.0. Дисплей: 800 x 600. 2.2. Минимальный состав программных средств Дополнительные программные средства не требуются.

2.3. Требование к персоналу (пользователю) Конечный пользователь программы должен обладать практическими навыками работы с графическим пользовательским интерфейсом операционной системы.

1. Подготовка окружения

Перед запуском программы выполните следующие действия:

Убедитесь, что на вашем компьютере установлена библиотека CurrencyConverter

2. Настройка программы

Скачайте или скопируйте файл программы в каталог на вашем компьютере.

Убедитесь, что каталог программы имеет доступ на чтение и запись.

1. Запуск программы

При первом запуске программы вам нужно будет ввести номер вашего счета

4. Основные функции программы

После запуска программы вам будет предложено выбрать одно из следующих действий:

Ввести номер вашего счета

Введите номер счета -20102006.

Приложение 2. Рисунок 6

Обмен валюты (например USD на SEK)

Выберите операцию под номером 4. Введите желаемую сумму в USD.

После ввода суммы автоматически программа обменяет валюту и выведет на экран результат.

Приложение 2. Рисунок 7

Заключение

В результате выполнения проекта была разработана программа, предназначенная для упрощения процесса обмена валюты. Основные возможности программы включают выбор валют для обмена и расчет суммы обмена с учетом текущего курса.

В процессе работы над проектом были использованы новые технологии для работы с курсами валют. Было изучено и применено использование API для получения актуальных курсов валют.

Дальнейшие планы по развитию проекта включают в себя:

To-do лист:

1. Расширение списка доступных валют для обмена
2. Добавление функционала автоматического обновления курсов валют
3. Создание возможности отслеживания изменения курсов валют в реальном времени

Список литературы и интернет-источников

1. Методические рекомендации по организации выполнения курсового проекта – КМПО РАНХиГС – Москва 2024.

2. ГОСТ 7.32-2001. Межгосударственный стандарт. Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления.

3. ГОСТ 19.701-90 Единая система программной документации. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения.

4.Статья про библиотеку CurrencyConverter

<https://github.com/alexprengere/currencyconverter>

5.Подключение библиотеки

<https://www.geeksforgeeks.org/currency-converter-in-python/>

6.Официальный сайт DORS-1250

<https://dors.com/catalog/dors-1250/>

7.Официальный сайт Hitachi iH-110

<http://hitachi110.com/>

Приложение 1.Код программы

import sys  
from currency\_converter import CurrencyConverter  
  
  
cur = CurrencyConverter()  
class Currency:  
 def \_\_init\_\_(self, name, rate):  
 self.name = name  
 self.rate = rate  
  
class Wallet:  
 def \_\_init\_\_(self, currency, amount):  
 self.currency = currency  
 self.amount = amount  
  
 def convert\_to(self, new\_currency):  
 exchange\_rate = new\_currency.rate / self.currency.rate  
 self.currency = new\_currency  
 self.amount \*= exchange\_rate  
  
class Exchange:  
 def \_\_init\_\_(self):  
 self.currencies = []  
  
 def add\_currency(self, currency):  
 self.currencies.append(currency)  
  
 def exchange\_currency(self, wallet, new\_currency):  
 if new\_currency in self.currencies:  
 wallet.convert\_to(new\_currency)  
 else:  
 print("Данная валюта не поддерживается!")  
  
  
class CurrencyExchange:  
 def \_\_init\_\_(self,cur):  
 self.cur = cur  
 self.c = cur.currencies  
  
 def exchange\_usd\_to\_rub(self, amount):  
  
 return amount \* self.usd\_rate  
  
 def exchange\_rub\_to\_usd(self, amount):  
  
 return amount / self.rub\_rate  
  
 def exchange\_usd\_to\_eur(self, amount):  
  
 return amount \* self.usd\_rate  
  
 def exchange\_eur\_to\_usd(self, amount):  
  
 return amount / self.eur\_rate  
  
 def exchange\_usd\_to\_sek(self, amount):  
  
 return amount \* self.usd\_rate  
  
 def exchange\_sek\_to\_usd(self, amount):  
  
 return amount / self.byn\_rate  
  
 def exchange\_usd\_to\_jpy(self, amount):  
  
 return amount \* self.usd\_rate  
  
 def exchange\_jpy\_to\_usd(self, amount):  
  
 return amount / self.jpy\_rate  
  
  
usd = Currency("USD", 1)  
  
  
rub = Currency("RUB", 73)  
  
  
eur = Currency("EUR", 0.85)  
  
  
sek = Currency("SEK", 2.6)  
  
  
jpy = Currency("JPY", 109)  
  
wallet = Wallet(usd, 100000)  
  
exchange = Exchange()  
  
exchange.add\_currency(usd)  
  
  
exchange.add\_currency(rub)  
  
  
exchange.add\_currency(eur)  
  
  
exchange.add\_currency(sek)  
  
  
exchange.add\_currency(jpy)  
  
  
bank\_number = input("Введите номер вашего счета ")  
  
if bank\_number == '20102006' :  
  
 print('Добро пожаловать!')  
  
else:  
 print('Неправильный номер!')  
  
 sys.exit()  
  
  
  
print("Ваш баланс (USD):", wallet.amount)  
  
  
  
  
print('Меню возможных действий:')  
  
  
print("1. Обмен RUB на USD")  
  
print("2. Обмен USD на EUR")  
  
print("3. Обмен EUR на USD")  
  
print("4. Обмен USD на SEK")  
  
print("5. Обмен SEK на USD")  
  
print("6. Обмен USD на JPY")  
  
print("7. Обмен JPY на USD")  
  
  
choice = input("Выберите операцию: ")  
  
exchange = CurrencyExchange(cur)  
  
if choice == '1':  
  
 rub\_amount = float(input("Введите сумму в RUB: "))  
  
 usd\_amount = cur.convert(rub\_amount, 'RUB', 'USD')  
  
 print(f"Вы получите {usd\_amount} USD")  
  
elif choice == '2':  
  
 usd\_amount = float(input("Введите сумму в USD: "))  
  
 eur\_amount = cur.convert(usd\_amount, 'USD', 'EUR')  
  
 print(f"Вы получите {eur\_amount} EUR")  
  
  
elif choice == '3':  
  
 eur\_amount = float(input("Введите сумму в EUR: "))  
  
 usd\_amount = cur.convert(eur\_amount, 'EUR', 'USD')  
  
 print(f"Вы получите {usd\_amount} USD")  
  
elif choice == '4':  
  
 usd\_amount = float(input("Введите сумму в USD: "))  
  
 sek\_amount = cur.convert(usd\_amount, 'USD', 'SEK')  
  
 print(f"Вы получите {sek\_amount} SEK")  
  
elif choice == '5':  
  
 sek\_amount = float(input("Введите сумму в SEK: "))  
  
 usd\_amount = cur.convert(sek\_amount, 'SEK', 'USD')  
  
 print(f"Вы получите {usd\_amount} USD")  
  
elif choice == '6':  
  
 usd\_amount = float(input("Введите сумму в USD: "))  
  
 jpy\_amount = cur.convert(usd\_amount, 'USD', 'JPY')  
  
 print(f"Вы получите {jpy\_amount} JPY")  
  
elif choice == '7':  
  
 jpy\_amount = float(input("Введите сумму в JPY: "))  
  
 usd\_amount = cur.convert(jpy\_amount, 'JPY', 'USD')  
  
 print(f"Вы получите {usd\_amount} USD")  
  
  
else:  
print("Неправильный выбор. Пожалуйста, выберите 1, 2, 3, 4, 5, 6 или 7")

Приложение 2. Результаты тестирования

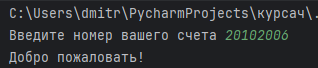


Рисунок 1. Результат теста 1

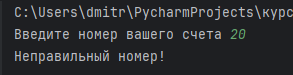


Рисунок 2. Результат теста 2

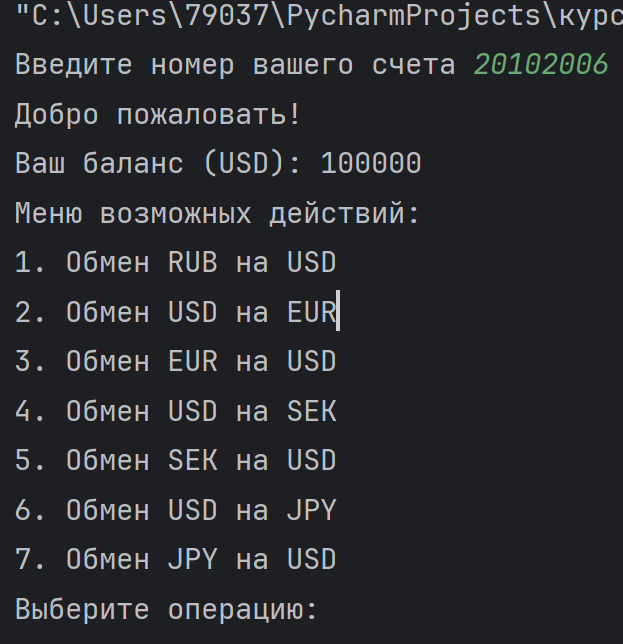


Рисунок 3. Результат теста 3

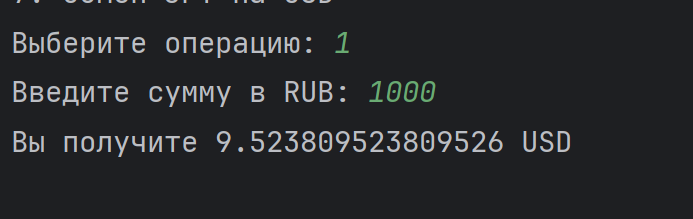


Рисунок 4. Результат теста 4

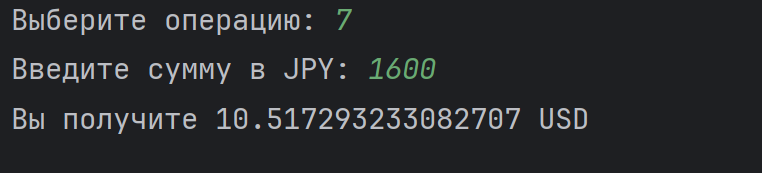


Рисунок 5. Результат теста 5

Приложение 3. Диаграмма прецедентов. Рисунок 6

