|  |
| --- |
| Изображение выглядит как текст, коллекция картинок, фарфор  Автоматически созданное описание  МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |
| Институт кибербезопасности и цифровых технологий |
| Кафедра КБ-2 «Прикладные информационные технологии» |

**ЗАЧТЕНО**

\_\_\_\_\_\_\_\_\_\_ /Лесько С.А./

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_г.

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №4  
по дисциплине  
«Технологии и методы программирования»**

***(наименование дисциплины)***

**Выполнил**

Студент 3 курса: Жидков Г.С.

Группы: БИСО-01-20

Специальности: 10.05.04

Москва 2023

**Оглавление**

[Итератор (Iterator) 3](#_Toc133854710)

[Посетитель (Visitor) 4](#_Toc133854711)

[Приложение 5](#_Toc133854712)

Итератор (Iterator)

На Рисунке 1 представлена UML-диаграмма реализации паттерна «Итератор», код диаграммы представлен в Приложении. Исходный код реализации на C# представлен в GitHub-репозитории, который указан в Приложении.

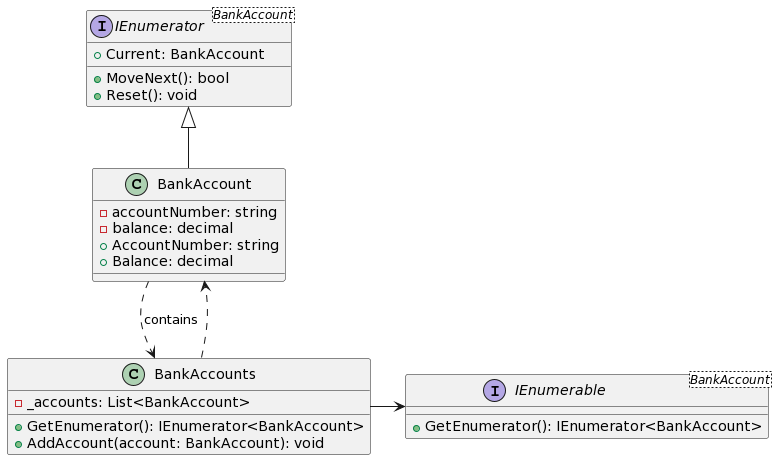


Рисунок 1 – Диаграмма для паттерна «Итератор»

*Описание.*

BankAccount - класс банковского счета, содержит баланс и номер счета.

BankAccountCollection - коллекция счетов, реализует интерфейс IEnumerable<BankAccount>, который содержит метод GetEnumerator() (он возвращает тип IEnumerator<BankAccount>, который используется для обхода коллекции).

Посетитель (Visitor)

На Рисунке 2 представлена UML-диаграмма реализации паттерны «Посетитель», код диаграммы представлен в Приложении. Исходный код реализации на C# представлен в GitHub-репозитории, который указан в Приложении.

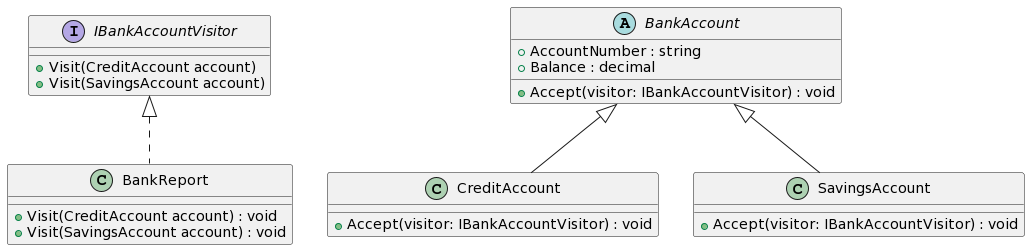


Рисунок 2 – Диаграмма для паттерна «Посетитель»

*Описание.*

BankAccount – базовый абстрактный класс банковского счета, содержит баланс и номер счета, а также абстрактный метод Accept.

CreditAccount - класс кредитного счета, который реализует базовый класс BankAccount

SavingsAccount - класс сберегательного счета, который реализует базовый класс BankAccount

IBankAccountVisitor - интерфейс посетителя с методами Visit для разных типов банковских счетов, которые необходимо реализовать наследнику.

BankReport - класс банковского счета, реализующего интерфейс IBankAccountVisitor.

Приложение

Ссылка на репозиторий GitHub:<https://github.com/st-georgy/TMP>

**Исходный код диаграммы паттерна «Итератор»**

@startuml

interface IEnumerator<BankAccount> {

+ Current: BankAccount

+ MoveNext(): bool

+ Reset(): void

}

interface IEnumerable<BankAccount> {

+ GetEnumerator(): IEnumerator<BankAccount>

}

class BankAccounts {

- accounts: List<BankAccount>

+ GetEnumerator(): IEnumerator<BankAccount>

+ AddAccount(account: BankAccount): void

}

class BankAccount {

- accountNumber: string

- balance: decimal

+ AccountNumber: string

+ Balance: decimal

}

BankAccounts -> IEnumerable

BankAccounts ..> BankAccount

BankAccount -up-|> IEnumerator

BankAccount ..> BankAccounts: contains

@enduml

**Исходный код диаграммы паттерна «Посетитель»**

@startuml

interface IBankAccountVisitor {

+ Visit(CreditAccount account)

+ Visit(SavingsAccount account)

}

abstract class BankAccount {

+ AccountNumber : string

+ Balance : decimal

+ Accept(visitor: IBankAccountVisitor) : void

}

class CreditAccount {

+ Accept(visitor: IBankAccountVisitor) : void

}

class SavingsAccount {

+ Accept(visitor: IBankAccountVisitor) : void

}

class BankReport {

+ Visit(CreditAccount account) : void

+ Visit(SavingsAccount account) : void

}

BankAccount <|-- CreditAccount

BankAccount <|-- SavingsAccount

IBankAccountVisitor <|.. BankReport

@enduml