**Прокопчук Роман Ігорович**

**Студент групи ШІ-1**

**Формальна специфікація «Термінал оплати проїзду в транспорті»**

Метод специфікації: моделе-орієнтований

Тип специфікації: поведінковий – було описано обмеження на поведінку об’єкта специфікації.

Функції, які було описано:

* «Оплата за проїзд з використанням проїзного квитка»
* «Оплата за проїзд з використанням кредитної картки»

**«Оплата за проїзд з використанням проїзного квитка»**

**Вхідні дані:**

1. Проїзний квиток, з якого буде здійснюватися оплата
2. Кількість пасажирів

Проїзний квиток містить у собі наступну інформацію:

1. Кількість поїздок
2. Ідентифікатор

Алгоритм:

*Якщо кількість поїздок < кількість пасажирів*

* *Оплата неможлива*

*Інакше знімаємо з картки к-ть поїздок, яка = к-ті пасажирів та інформуємо про успішну оплату*

**«Оплата за проїзд з використанням кредитної картки»**

**Вхідні дані:**

1. Кредитна картка, з якої здійснюватиметься оплата
2. Кількість пасажирів
3. Ціна проїзду за одного пасажира

Кредитна картка містить у собі наступну інформацію:

1. Баланс рахунку

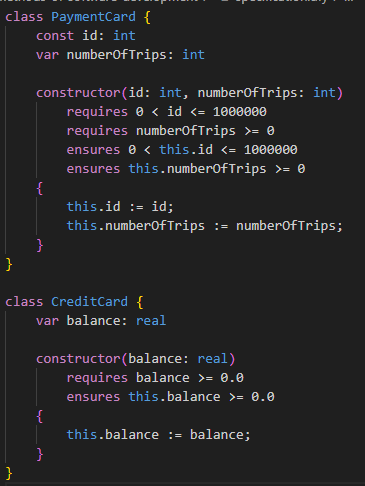
Алгоритм:

* Обраховуємо ціну проїзду за задану кількість пасажирів за формулою: [К-ть пасажирів] \* [Ціна проїзду за одного пасажира]
* Якщо ціна проїзду > баланс на карті
  + Оплата неможлива
* Інакше знімаємо з картки к-ть грошей, яка = ціні проїзду за задану к-ть пасажирів та інформуємо про успішну оплату

**Інструменти, які були використані для реалізації:** VSCode, Dafny

**Реалізація:**

Спочатку було створено класи, які описують проїзний квиток(PaymentCard) та кредитну картку(CreditCard):

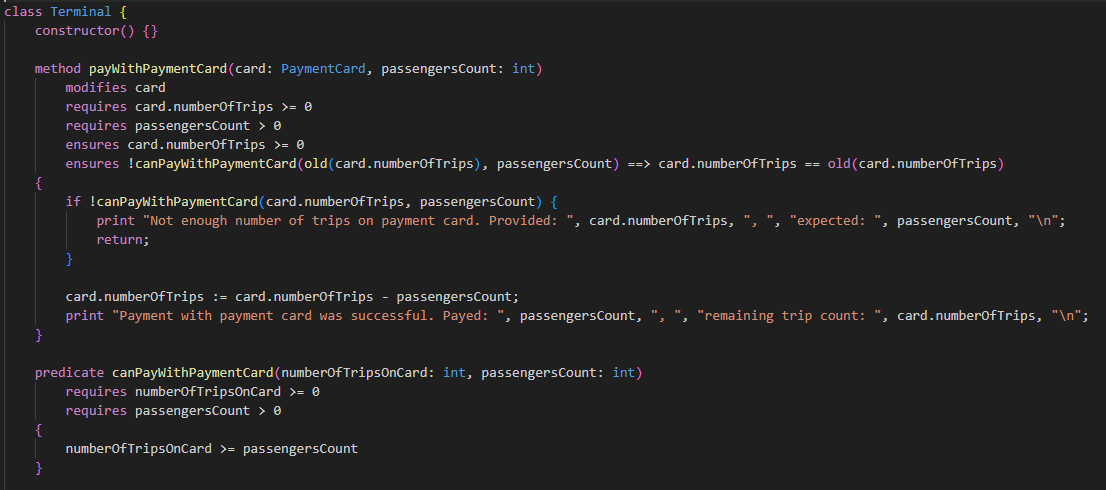


У конструкторах класів було наведено передумови та постумови, зокрема:

* PaymentCard
  + Передумова: id має бути у інтервалі (0, 100000)
  + Передумова: к-ть поїздок має бути >= 0
  + Постумова: id після ініціалізації поля має бути у інтервалі (0, 100000)
  + Постумова : к-ть поїздок після ініціалізації поля має бути >= 0
* CreditCard:
  + Передумова: баланс має бути >= 0
  + Постумова: баланс після ініціалізації має бути >= 0

Далі, було створено клас Terminal та два методи, що описують функції терміналу для оплати проїзду

Метод **«Оплата за проїзд з використанням проїзного квитка»**

****

Передумови:

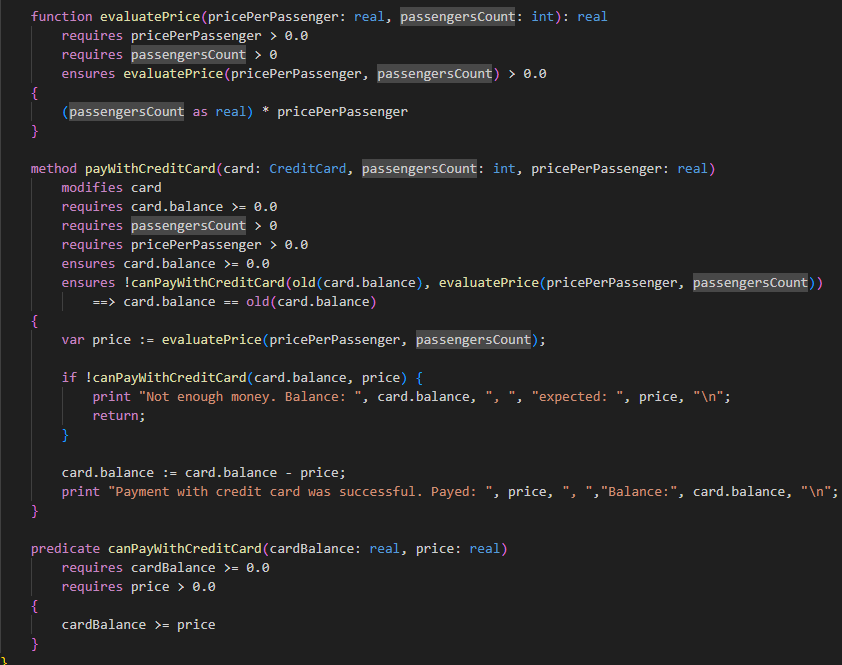
* К-ть поїздок у проїзному квитку має бути >= 0
* К-ть пасажирів > 0

Постумови:

* К-ть поїздок у проїзному квитку після спроби оплати має бути >= 0
* Якщо оплата не була успішною, то к-ть поїздок у проїзному квитку до спроби оплати = к-ті поїздок у проїзному квитку після спроби оплати

Цей метод використовує предикат **cayPayWithPaymentCard** для перевірки можливості оплати проїзним квитком для заданої к-ті пасажирів

**Метод «Оплата за проїзд з використанням кредитної картки»**



Метод використовує функцію **evaluatePrice** для обчислення ціни проїзду для заданої кількості пасажирів. Також, є предикат **cayPayWithCreditCard** для перевірки можливості оплати

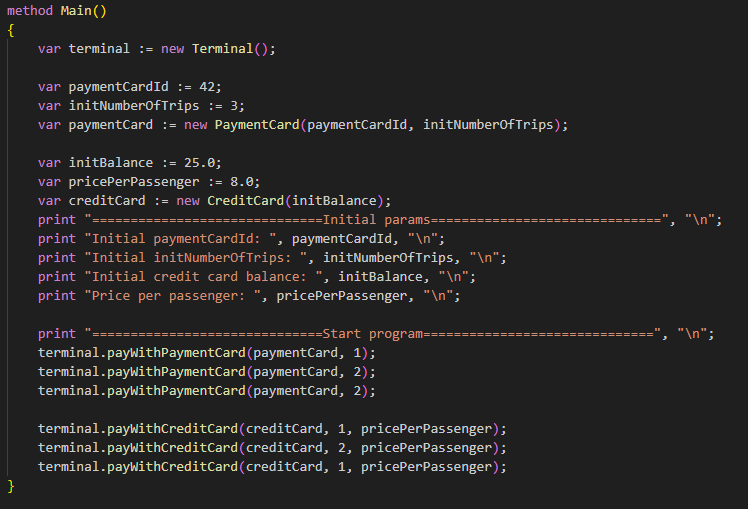
Передумови:

* Баланс на карті >= 0
* К-ть пасажирів > 0
* Ціна проїзду за одного пасажира > 0

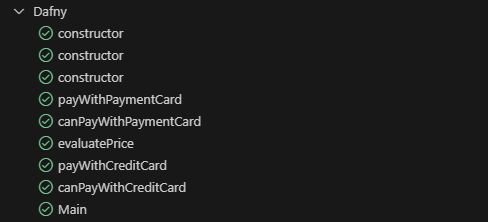
Постумови:

* Баланс на карті після спроби оплати має бути >= 0
* Якщо оплата не була успішною, то баланс на карті до спроби оплати = балансу на карті після спроби оплати

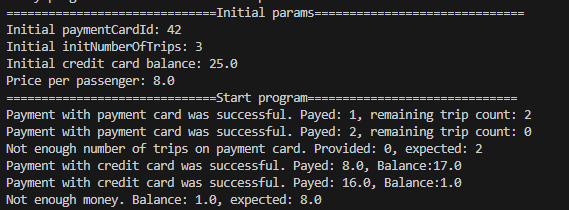
**Симуляція поведінки у методі Main()**

****

**Результат специфікації**

****

**Результат роботи програми**

****