Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и кибербезопасности

Высшая школа «Компьютерных технологий и информационных систем»

**ОТЧЕТ по лабораторной работе №8**

по дисциплине «Системный подход в разработке программного обеспечения»

**Выполнил:**

студент группы 5130902/20201 \_\_\_\_\_\_\_\_\_\_\_\_\_ А. И. Сафонов

подпись

**Проверил:**

кандидат тех. наук, доцент \_\_\_\_\_\_\_\_\_\_\_\_ С. А. Нестеров

подпись

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

Санкт-Петербург

2024

1. **По уроку 7.5**
2. **Упражнение 1**

Продолжайте редактировать проект PrisonTest. Вам предоставлена версия этой

программы. Создайте класс Cell, который содержит следующее:

– Строка name камеры.

– Значение Boolean, описывающее открытую дверь.

– Конструктор двух аргументов, который устанавливает оба поля.

Измените класс Prisoner следующим образом:

– Содержит поле Cell.

– Устанавливает поле Cell на основе параметра конструктора.

– Выводит name камеры в качестве метода display()

Создайте метод openDoor() в классе Prisoner. Получите доступ и измените

соответствующее поле в объекте Cell следующим образом:

• Если дверь закрыта, откройте ее.

• Если дверь открыта, закройте ее.

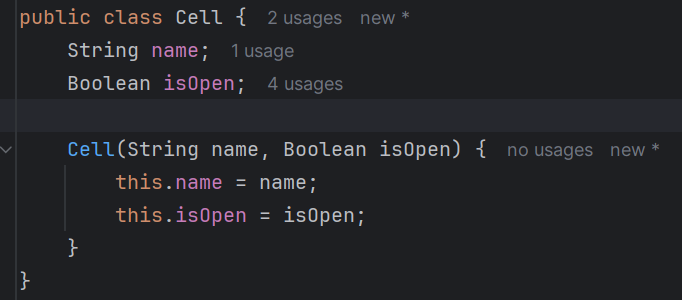
Выведите, открывается или закрывается ли дверь.

В главном методе:

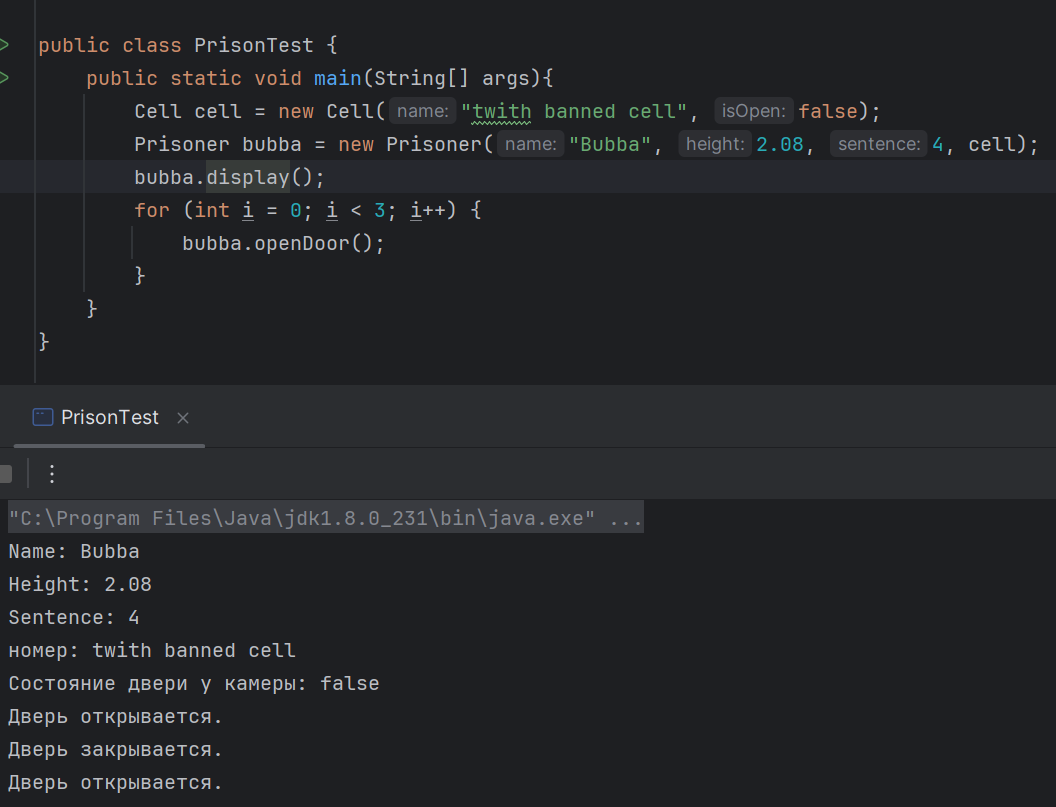
– Создание экземпляров объектов Cell и Prisoner.

– Вызовите метод display() заключенного один раз.

– Вызовите метод openDoor() несколько раз.

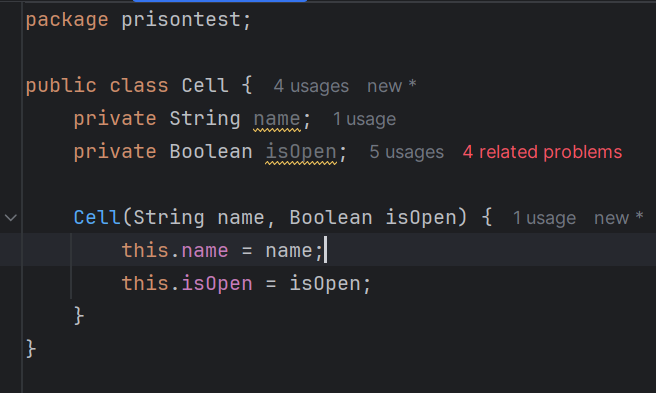
****

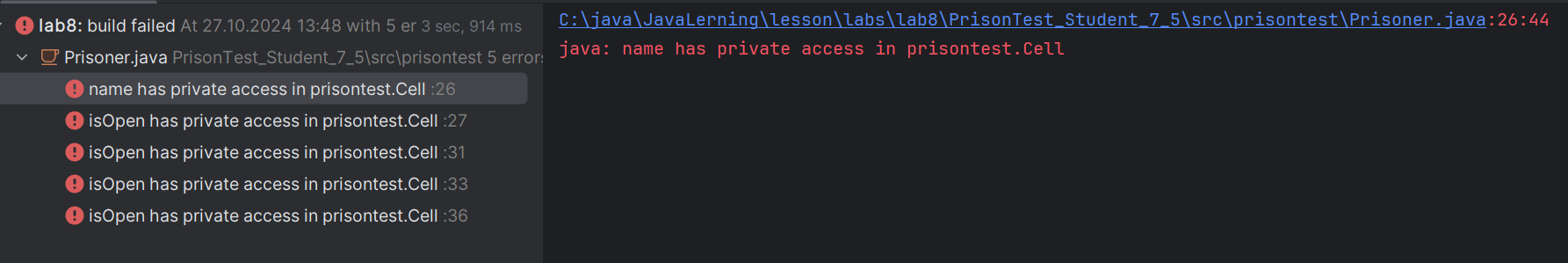
****

****

1. **Упражнение 2**

* Продолжайте редактировать проект PrisonTest.
* Изменение класса Cell:
* Измените значение его полей на значение private.
* Сохраните файл.
* Содержит ли NetBeans какие-либо неполадки?
* Что это за неполадки?
* Где они происходят?

****

****

Ошибка возникает из-за того, что доступ к полям private не может быть

получен за пределами класса Cell.

1. **Упражнение 3**

Продолжайте редактировать проект PrisonTest. Измените класс Cell следующим

образом…

– Устанавливающие методы существуют для полей name и isOpen.

– Имеется поле частного кода безопасности из 4-х цифр. Оно

инициализируется из конструктора и не имеет метода чтения.

–

Существует устанавливающий метод для открытия/закрытия выполняет следующие действия:

* Принимает код безопасности как аргумент.
* Выведите, если код неверен.
* Если код верен и дверь закрыта, открывает ее.
* Если код верен и дверь открыта, закрывает ее.
* Выводит, если дверь открыта или закрыта.

Измените класс Prisoner следующим образом…

– Метод display() выводит имя камеры.

– Метод openDoor() удален.

Измените главный метод следующим образом…

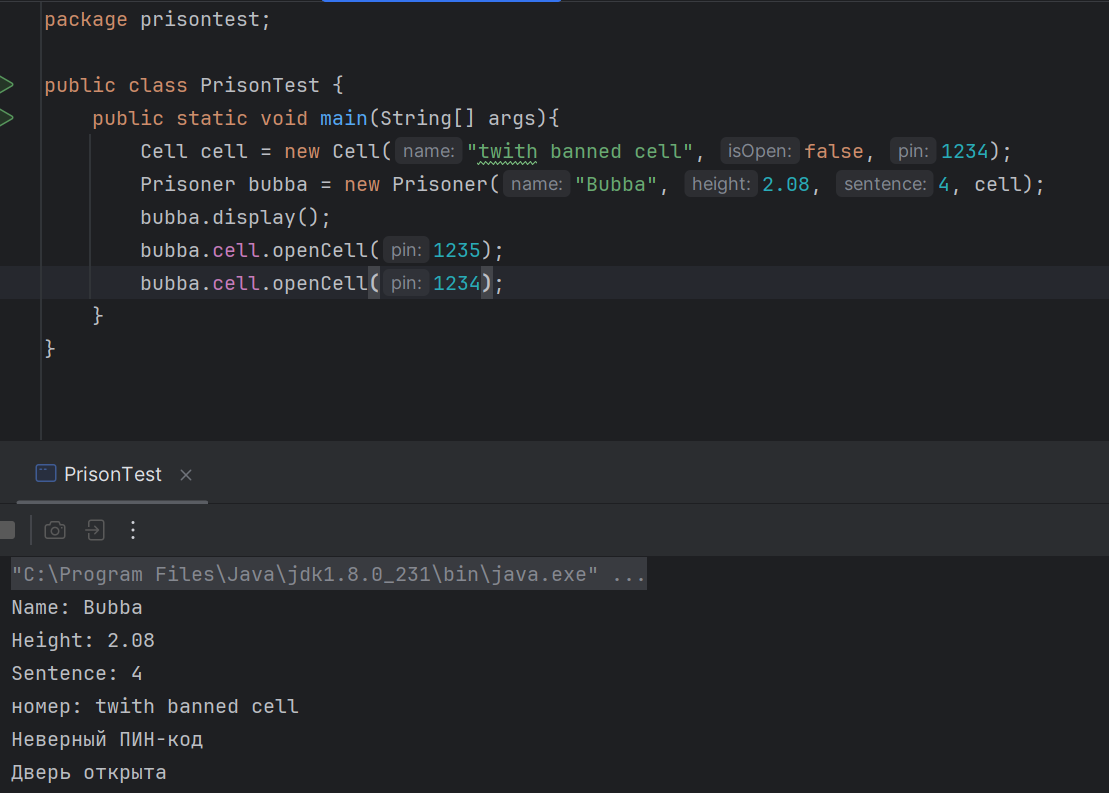
– Для объекта Cell экземпляр создан надлежащим образом.

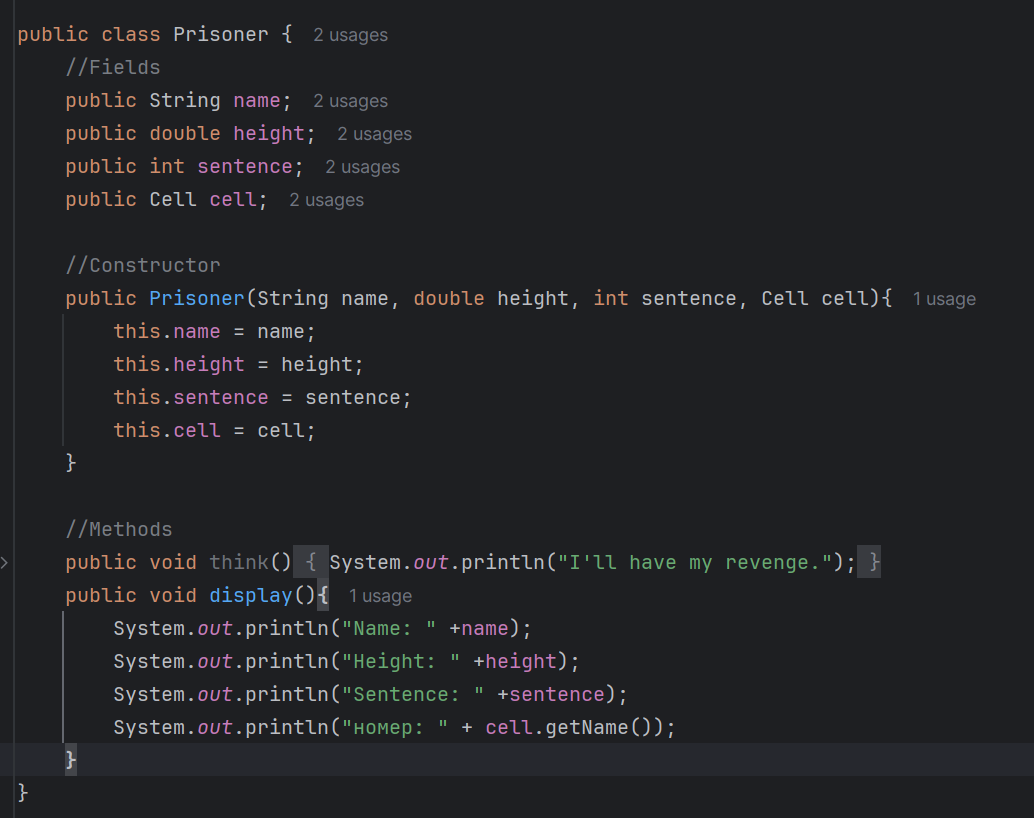
– Заключенный больше не пытается открыть дверь камеры.

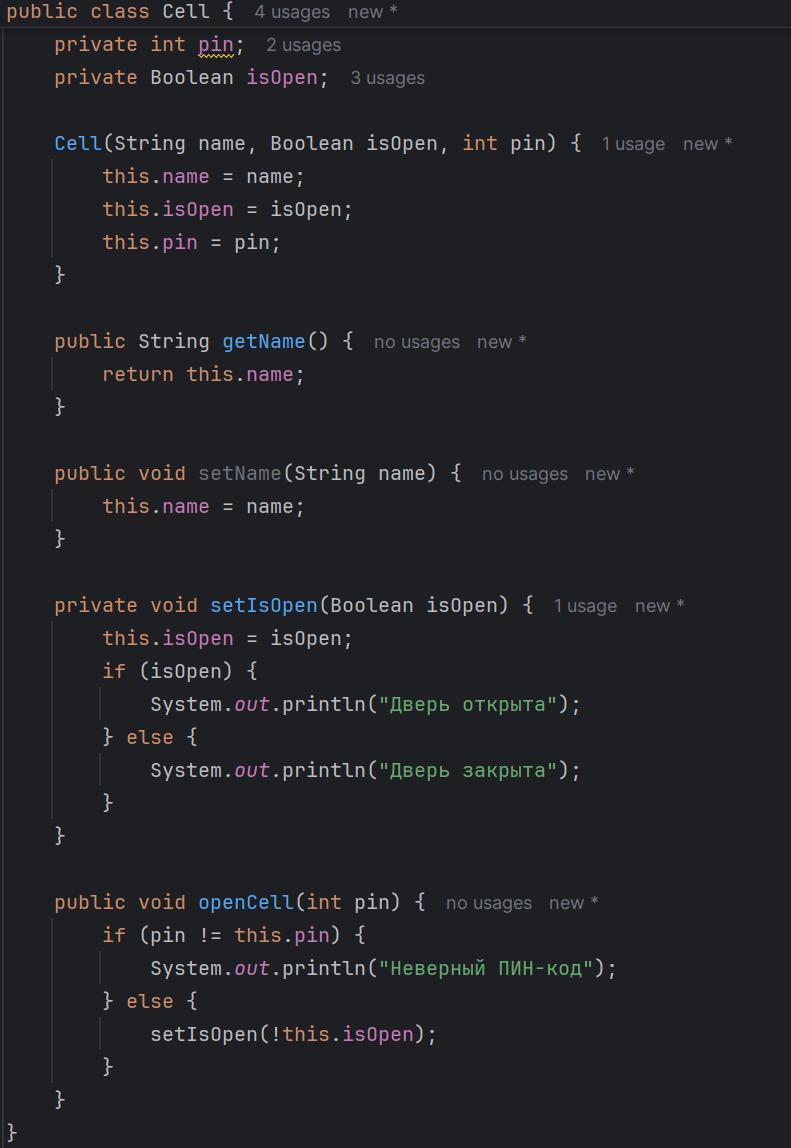
– Выполняется проверка возможности открытия и закрытия двери классом

камеры.

Попробуйте указать правильный и неправильный коды безопасности.

****

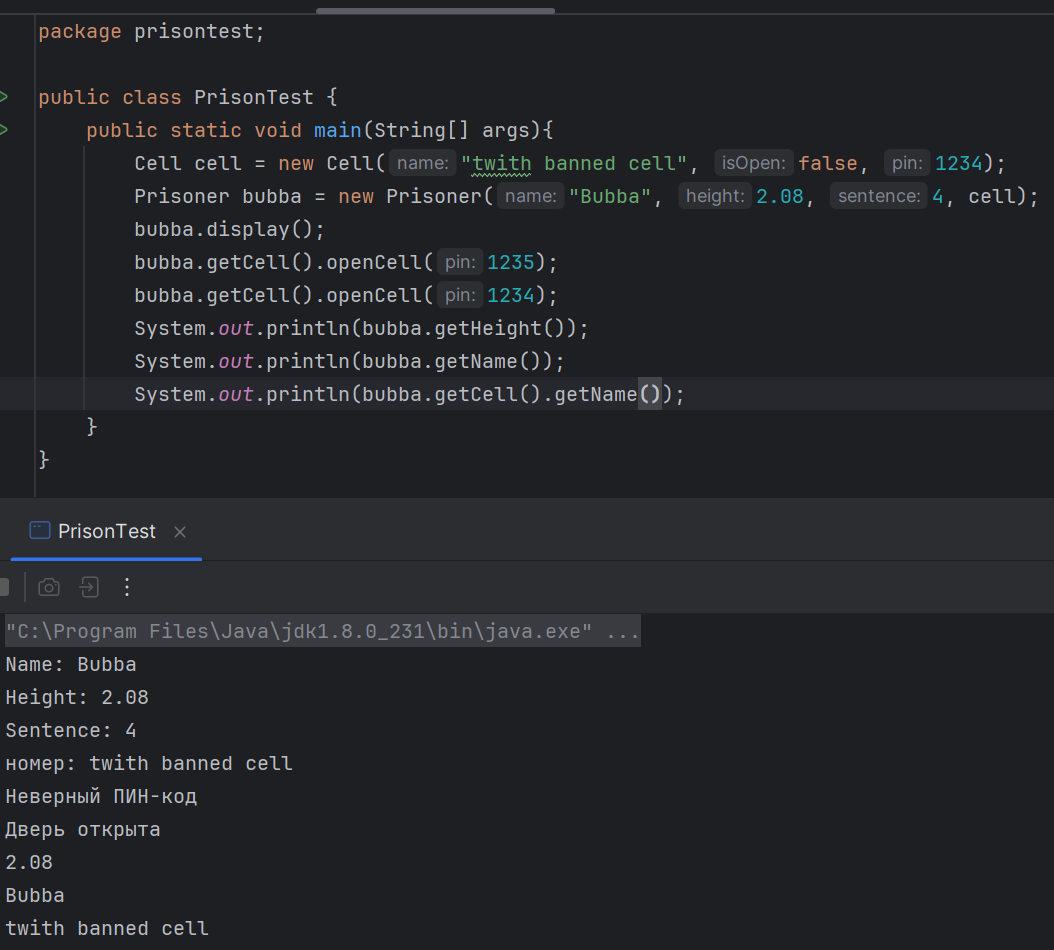
****

****

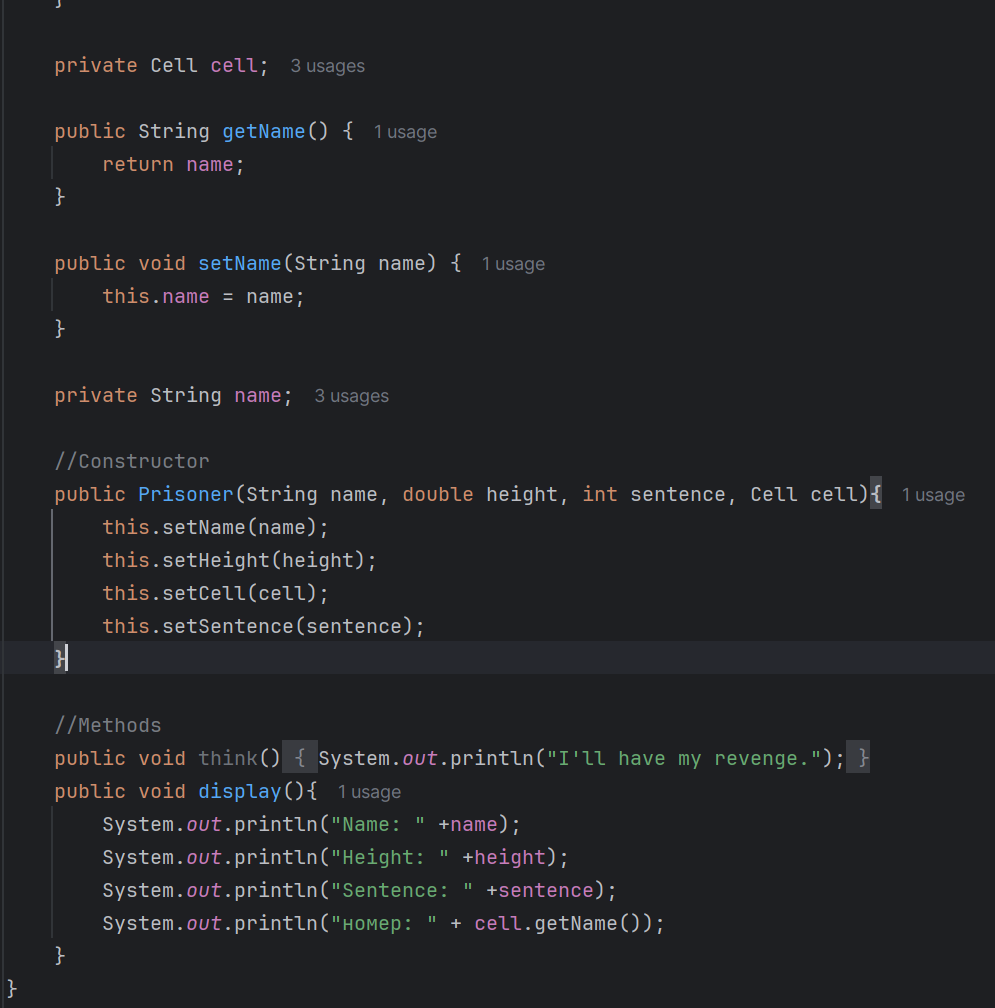
1. **Упражнение 4**

Продолжайте редактировать проект PrisonTest. Выполните инкапсуляцию

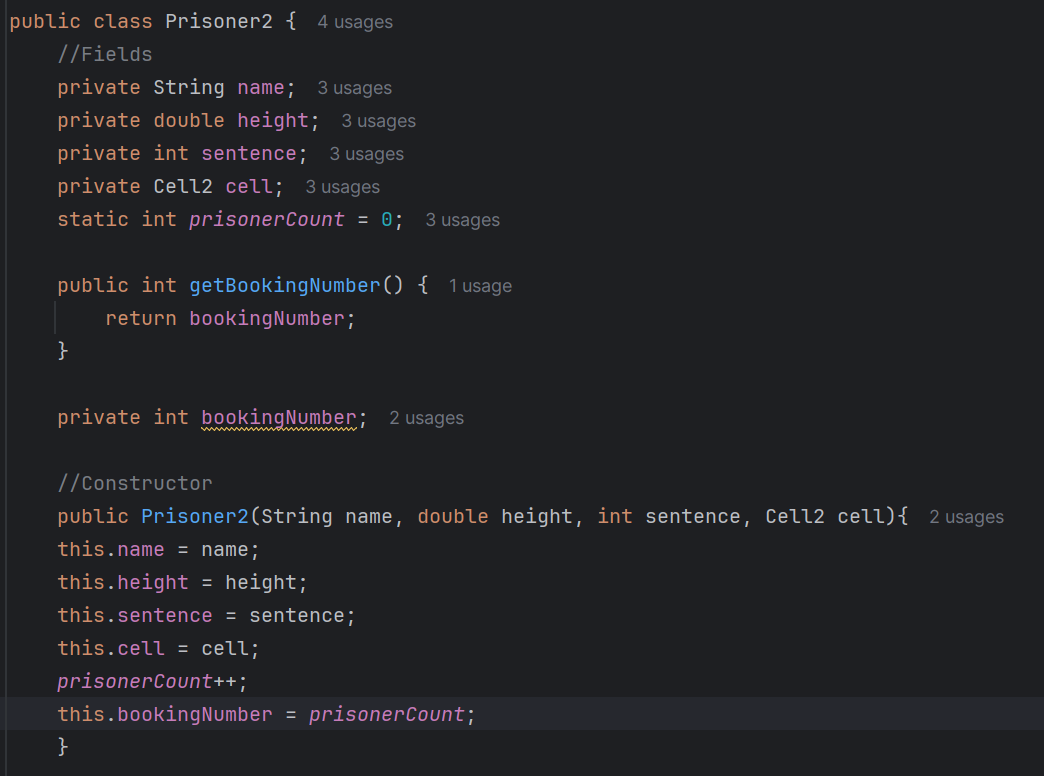
класса Prisoner. Установите для его полей значение private. Укажите методы чтения и устанавливающие методы для каждого поля.

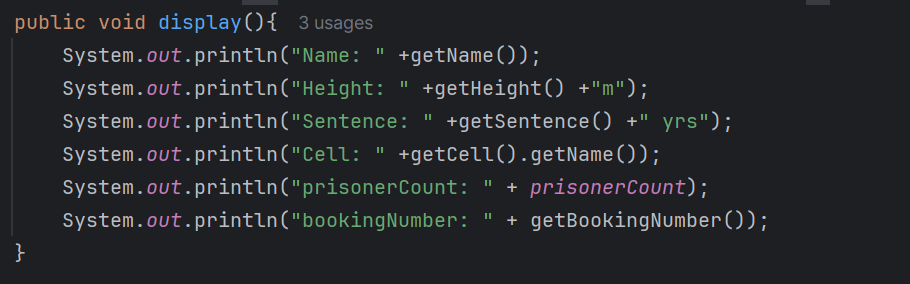


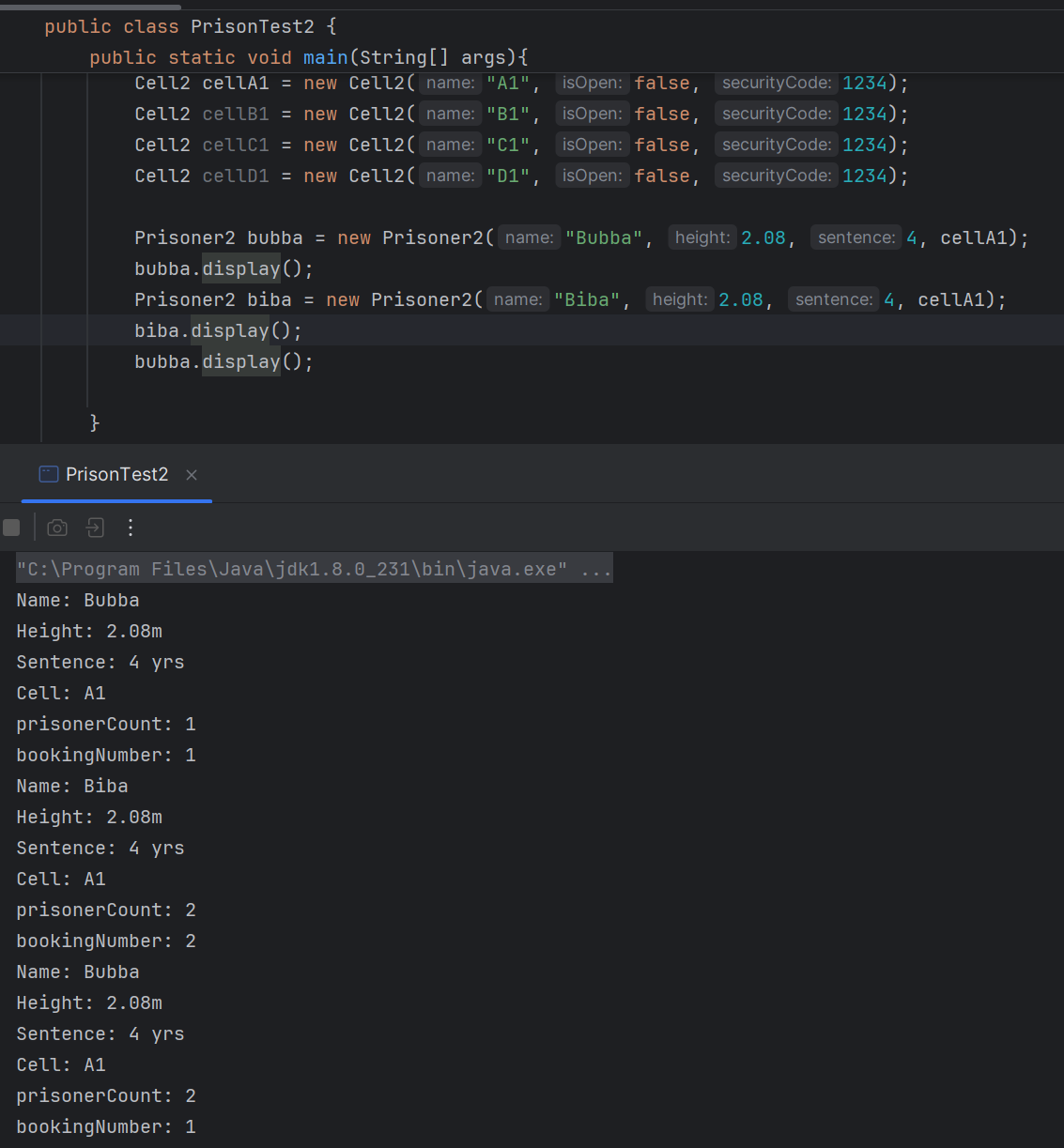




1. **По уроку 7.6**
2. **Упражнение 1**

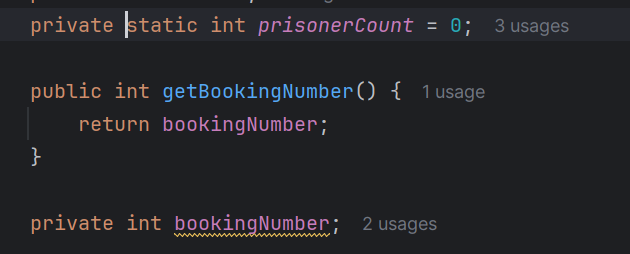
* Продолжайте редактировать проект PrisonTest.
* Измените класс Prisoner:
* Содержи статическое целочисленное поле prisonerCount. Данное поле подсчитывает общее количество заключенных, для которых созданы экземпляры. Выполните инициализацию этого поля до значения 0. Увеличивайте значение поля при каждом создании экземпляра заключенного.
* Включите целочисленное поле bookingNumber. Данное поле было инициализировано с помощью текущего значения prisonerCount.
* Выведите bookingNumber и prisonerCount как часть метода display().
* Создайте экземпляры нескольких заключенных и отобразите сведения о них.

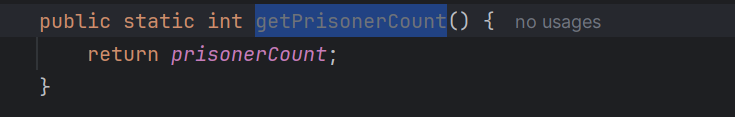


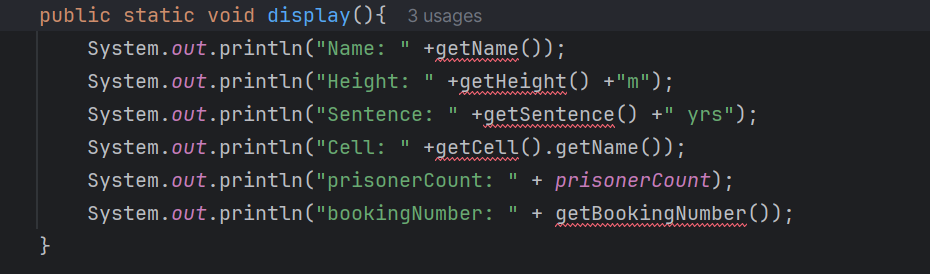


1. **Упражнение3**

* Продолжайте редактировать проект PrisonTest.
* Измените класс Prisoner:
* Выполните инкапсуляцию поля prisonerCount. Сделайте это поле частным и создайте статический метод чтения.
* Попробуйте сделать метод отображения статичным. Что представляют собой неполадки NetBeans?
* В главном методе:
* Вызовите метод чтения, который только что был создан, и выведите возвращенное значение.

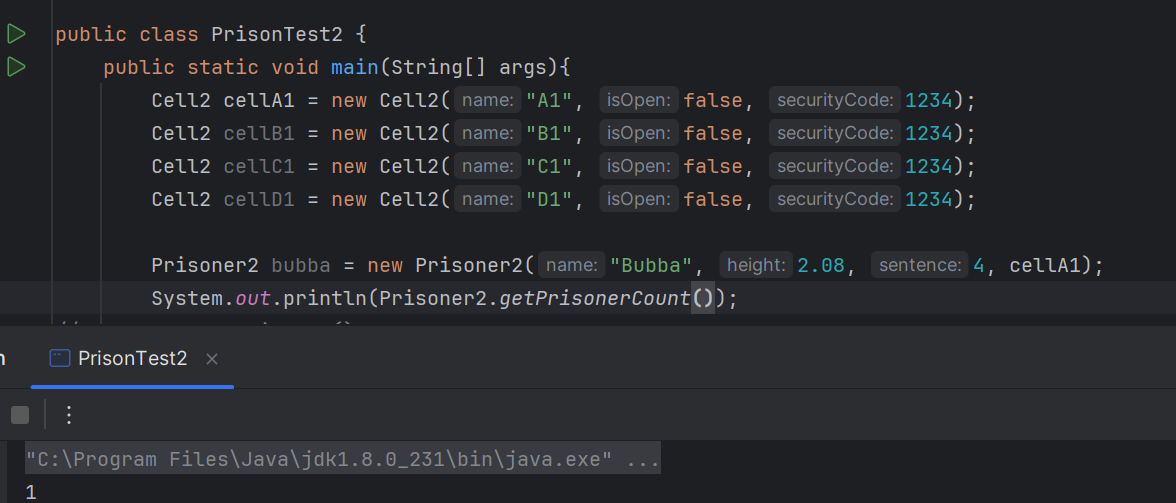






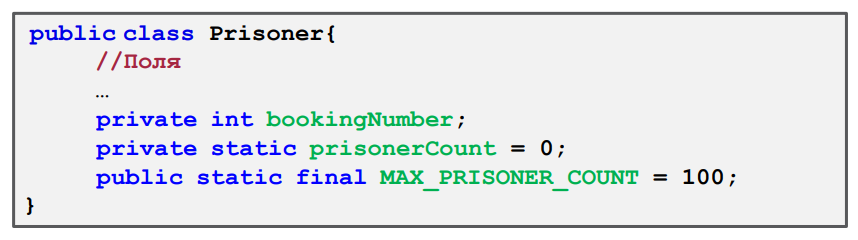
Видим, что возникают ошибки из-за того, что используем нестатические переменные и методы в статичном методе.

Статические поля и статические методы можно вызвать без создания экземпляра объекта. Но переменные экземпляров должны быть связаны с определенным экземпляром. Создается парадокс, если статический метод пытается получить доступ к информации об экземпляре до его создания. Поэтому Java не разрешает, чтобы статические методы содержали переменные или методы экземпляров.



1. **Создание полей static final**

* Вы можете указать для переменных static значение final.
* Помните, что имена переменных final…
* По правилам именования пишутся с заглавных букв.
* Используют нижнее подчеркивание (\_) для разделения слов.



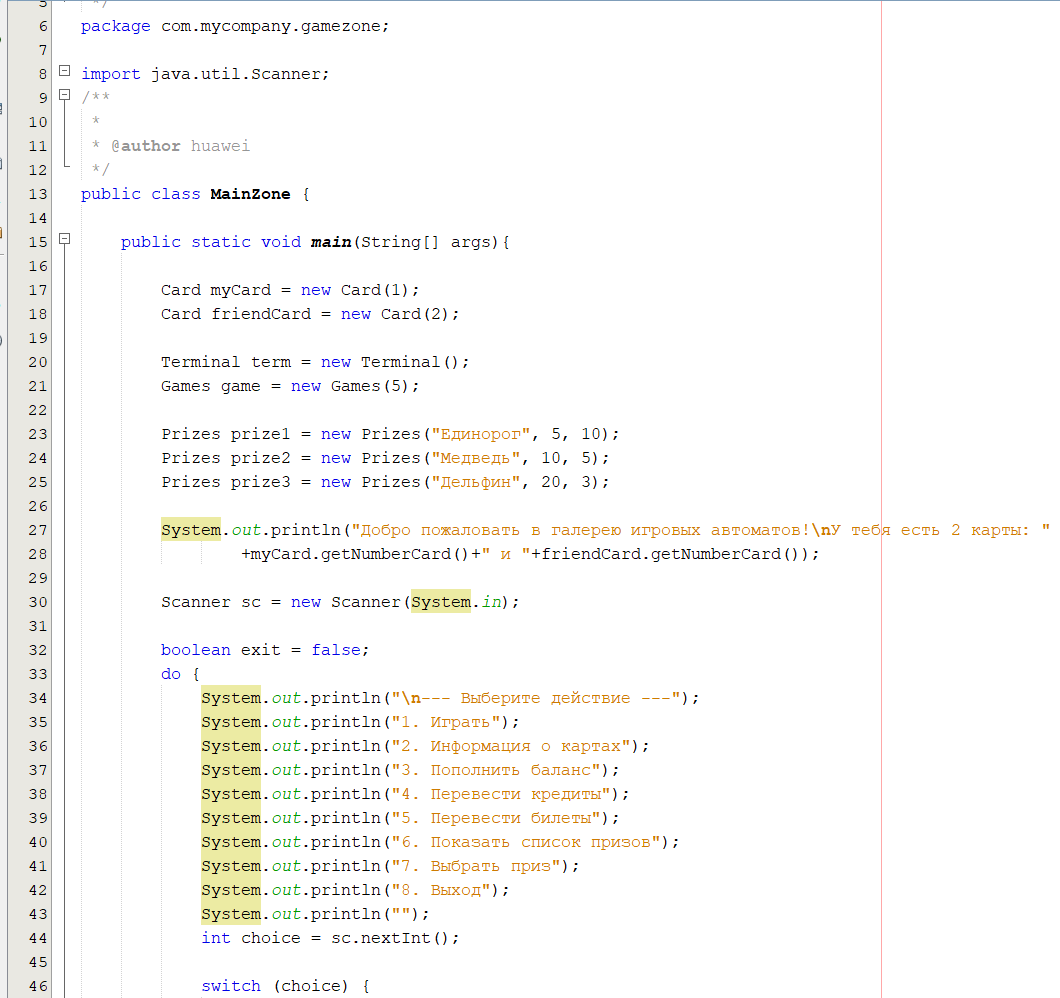
В Java константы должны быть инициализированы при объявлении, вы не можете просто присвоить значение.

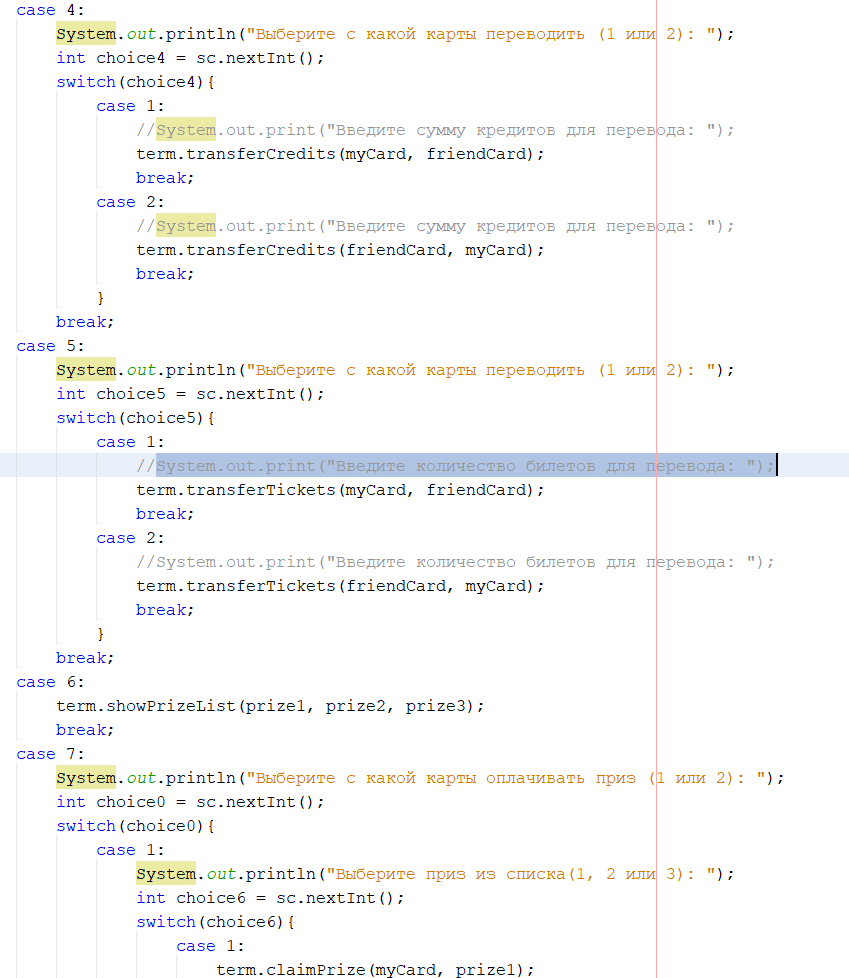
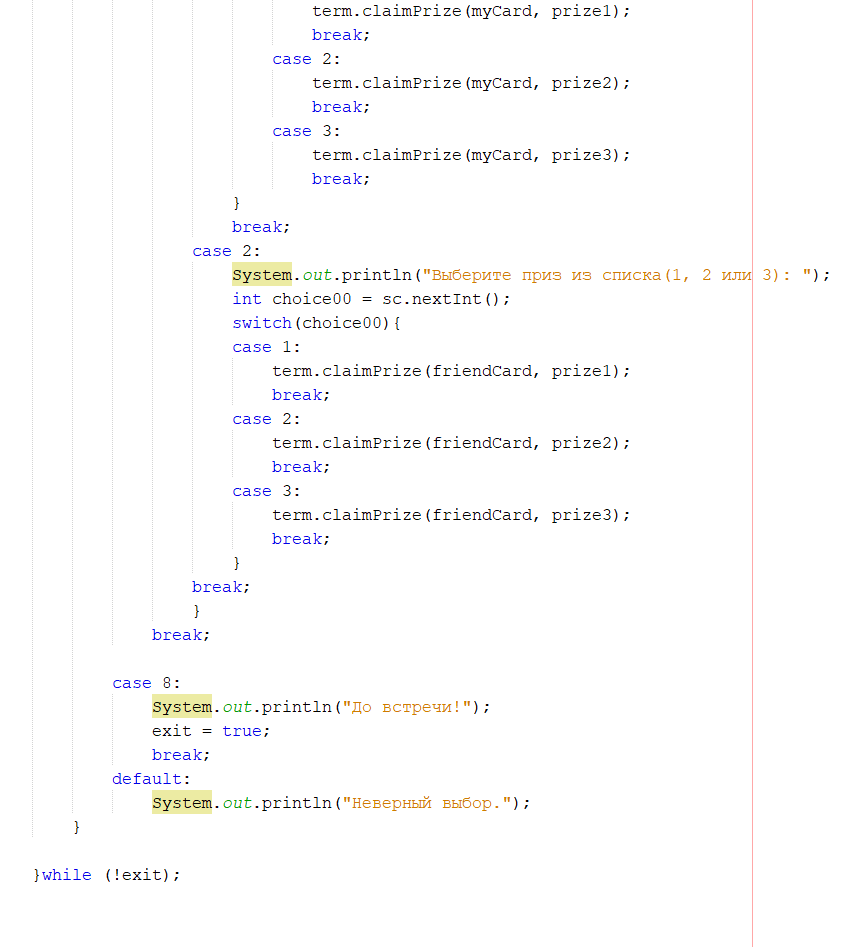
* Инкапсуляция не дает манипулировать переменными нежелательным способом.
* Но риска искажения примитивов public static final нет, так как изменить их значения невозможно.
* Это полезно для констант, например π, e, или других значений, которые часто используются в расчетах.
* Эти переменные вызываются непосредственно вместо методов чтения.

1. **Практика 7.**

Написать программу Java, которая моделирует свойства, поведение и взаимодействие объектов в этой галерее игровых автоматов. Вам также потребуется тестовый класс, который содержит главный метод. Используйте главный метод для моделирования определяющих действий программы, таких как создание экземпляров объектов и считывание карт. Все поля должны иметь значение private. Предоставьте метод получения и все необходимые методы установки.

Главный метод:

****

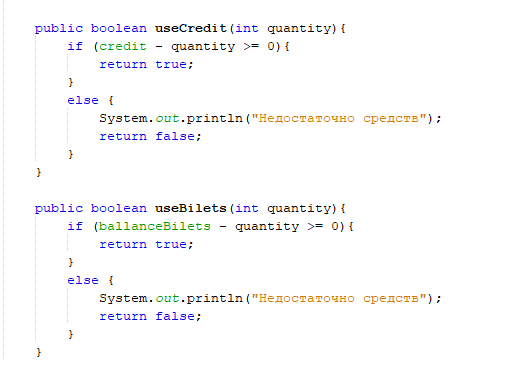
**** **** 

Games:



Card:

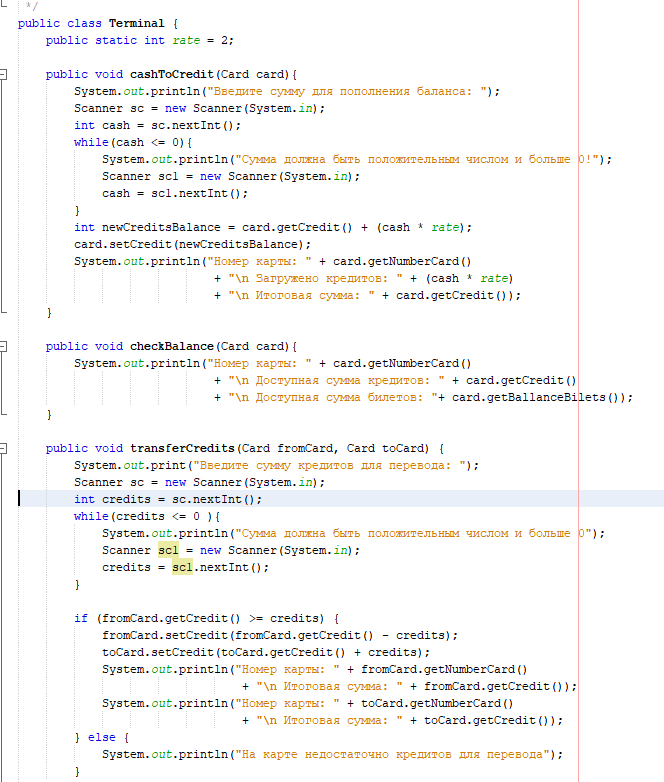




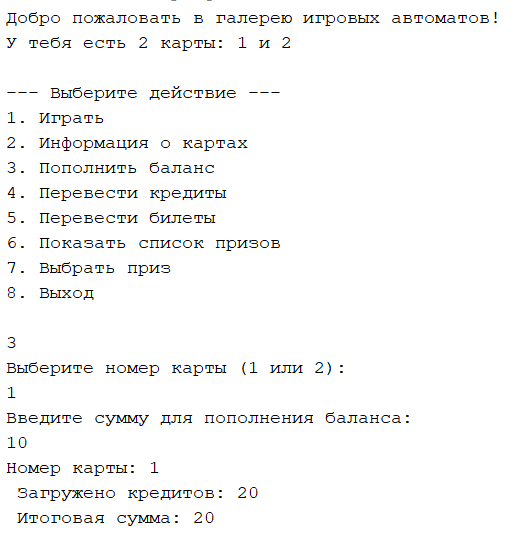
Prizes:



Terminal:

Результат:



UML-диаграмма классов:

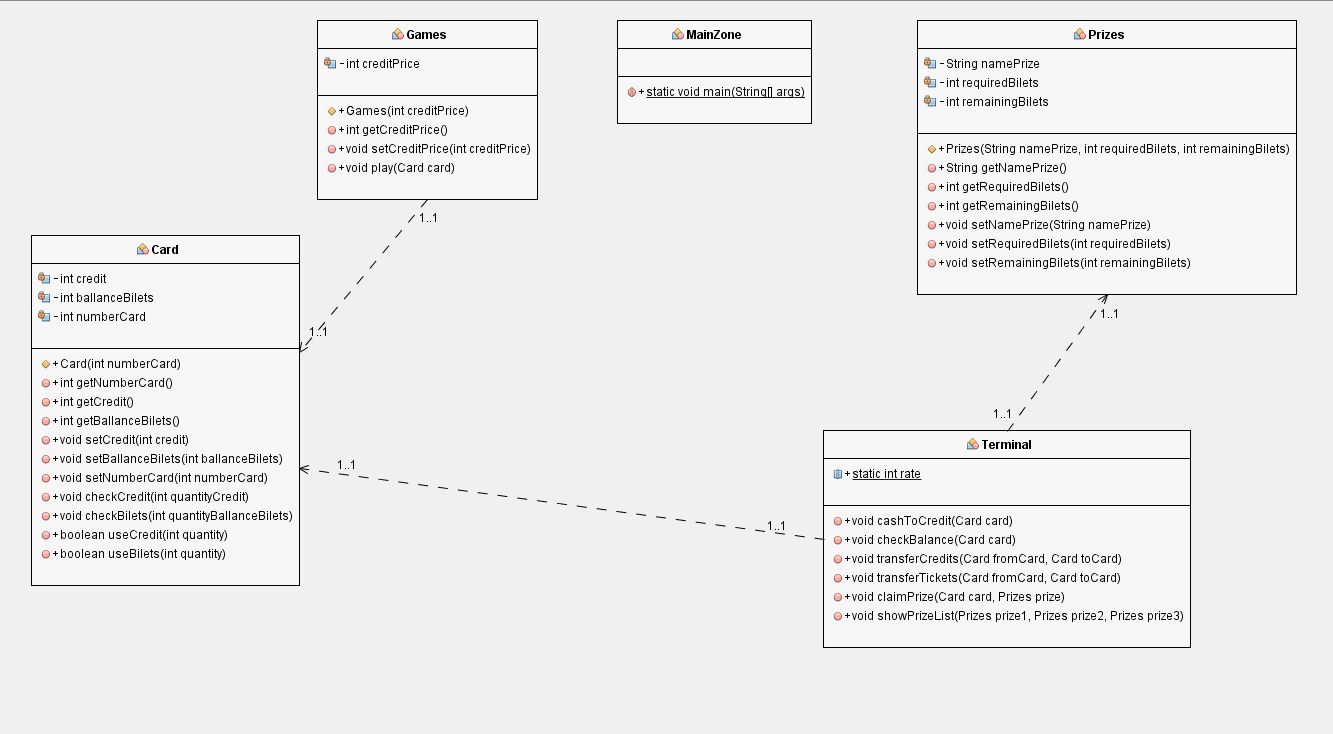
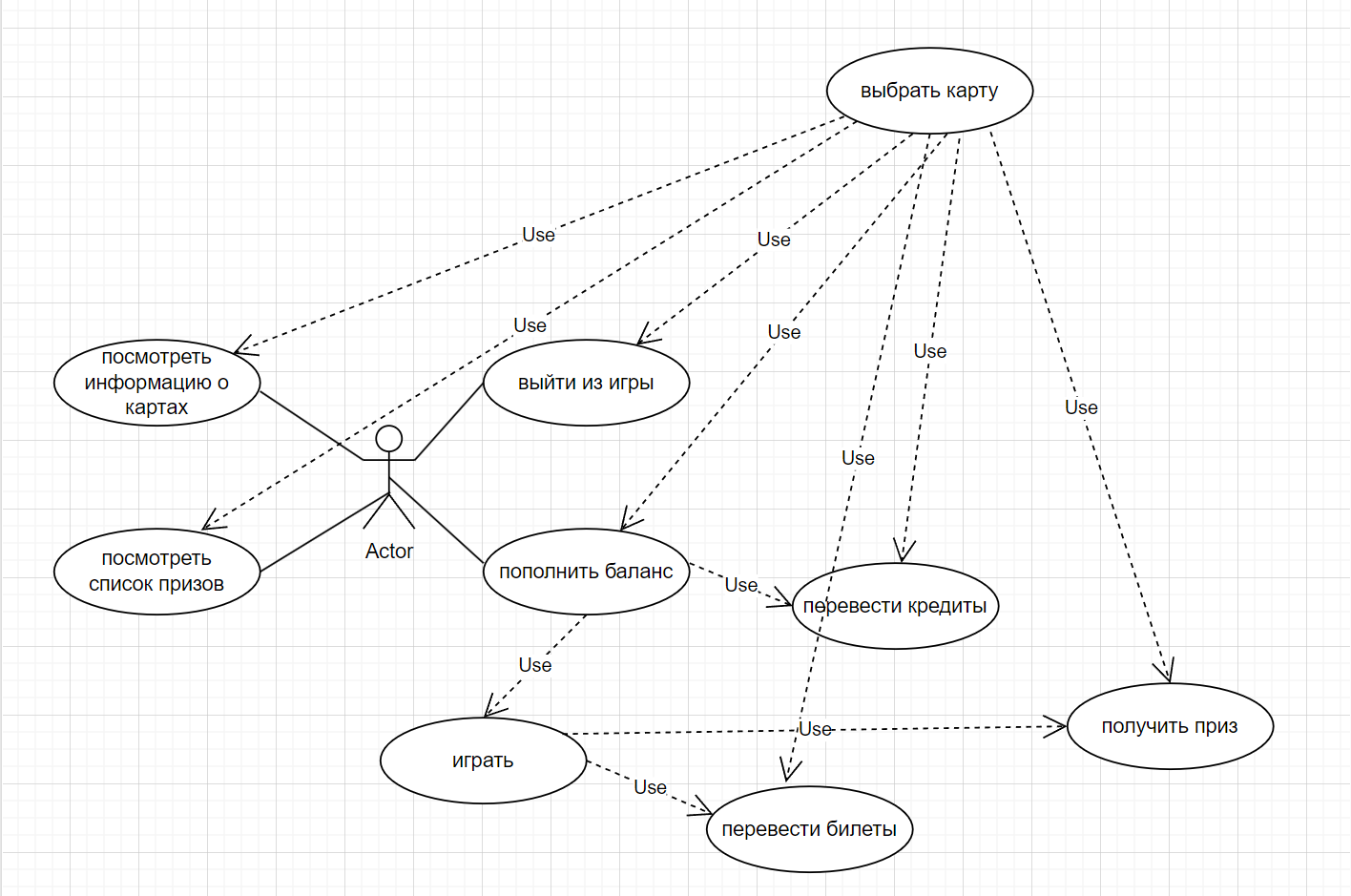
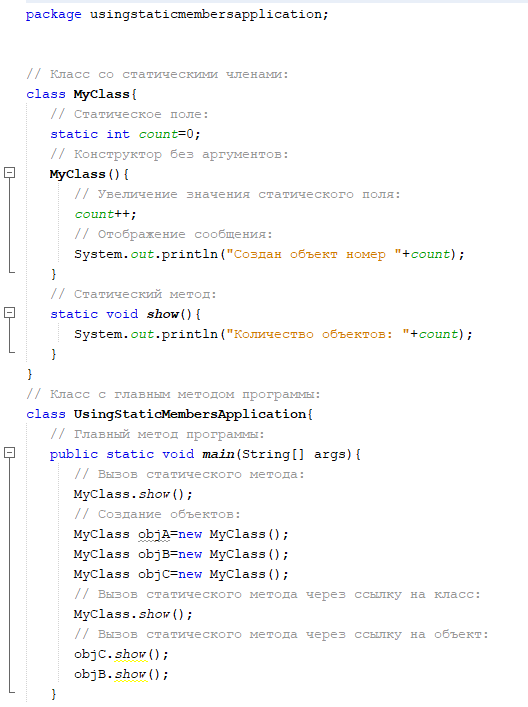


Диаграмма вариантов использования:

****

1. **По книге Васильев, А. Н. Программирование на Java для начинающих**
2. **Листинг 3.5.**

**Программный код проекта UsingStaticMembersApplication**



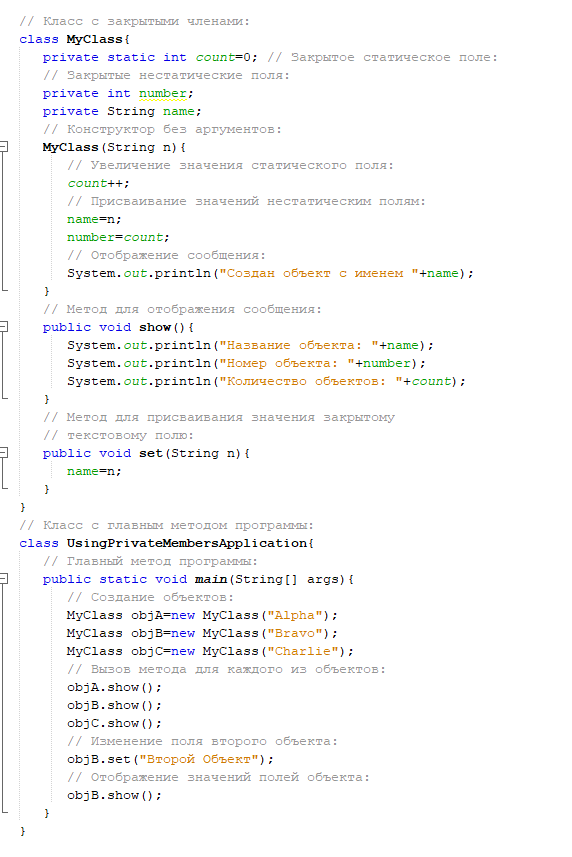
В программе описывается класс MyClass, у которого есть статическое целочисленное поле count, конструктор без аргументов и статический метод show(). Поле count инициализируется с начальным нулевым значением. При вызове статического метода show()в консольном окне отображается сообщение с указанием значения статического поля count. Статическое поле count изменяется при вызове конструктора. В теле конструктора командой count++ на единицу увеличивается значение статического поля, после чего отображается сообщение в окне вывода.

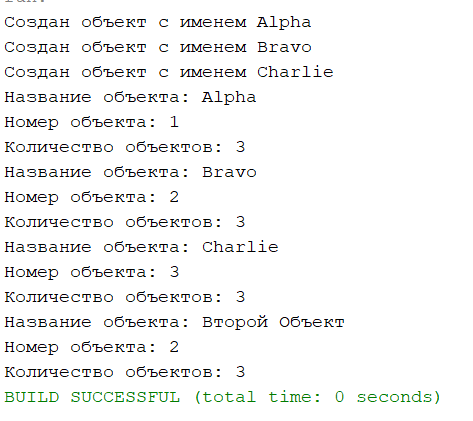
Таким образом, статическое поле count используется для подсчета созданных в программе объектов класса MyClass.

В главном методе программы командой MyClass.show() вызывается статический метод show(). В результате вызова метода в консольном окне появляется сообщение о том, что количество объектов нулевое. На момент вызова метода еще ни один объект не создан. Тем не менее статический метод show(), равно как и статическое поле count, «существуют», и их можно использовать, даже не создавая объект. При обращении к методу show() мы перед именем метода указываем имя класса. Метод show() можно вызвать и «обычным» образом, через ссылку на объект.

1. **Листинг 3.6.**

**Программный код проекта UsingPrivateMembersApplication**





В классе MyClass описано закрытое статическое поле count с начальным нулевым значением, два закрытых нестатических поля (целочисленное number и текстовое name). Также в классе описаны нестатические открытые методы:

* методом show() отображаются значения полей name и number объекта, из которого вызывается метод, а также значение статического поля count;
* метод set() предназначен для присваивания значения закрытому полю name.

Общая ситуация:

* Статическое поле count закрытое и его значение изменяется (увеличивается на единицу) при создании объекта. Других способов (кроме создания объекта) изменить значение поля count нет.
* Закрытое поле number получает значение при создании объекта, после этого нет возможности изменить значение поля.
* Закрытое текстовое поле name получает значение на основе аргумента конструктора. Впоследствии изменить значение поля можно с помощью метода set().