Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Российская академия народного хозяйства и государственной службы при Президенте Российской Федерации»

Нижегородский институт управления

Кафедра Информатики и информационных технологий

ОТЧЕТ

ПО ЗАДАНИЮ К ЛЕКЦИИ №9

«Классы. Часть 2»

Выполнил: студент группы: ИБ-321

Борисков Дмитрий Андреевич

Преподаватель: Окулич Виктор Иванович

Нижний Новгород

2023 г.

Оглавление

[Задание 9.1 3](#_Toc130750716)

[1.1 Реализовать проекты с программами рисунков 9.1 и 9.2. 3](#_Toc130750717)

[1.2 Определить (идентифицировать) причины ошибок проекта рисунка 9.1 и исправить их. 5](#_Toc130750718)

[1.3 В проекте рисунка 9.2 объяснить работу кодов строчек 6-10 и добиться вывода другого значения возраста по методу **.setAge**. 5](#_Toc130750719)

[1.4 Разобраться в коде, приведённом в https://3ty.ru/languages/java/gettery-i-settery-v-java и реализовать его в соответствующем проекте. 5](#_Toc130750720)

[1.5 Сформулировать преимущества и недостатки инкапсуляции 6](#_Toc130750721)

[Задание 9.2 7](#_Toc130750722)

[1. Программно реализовать ошибки и их исправление из раздела 9.3.3; 7](#_Toc130750723)

[2. Разобрать разделы 9.3.5.1 и 9.3.5.2 8](#_Toc130750724)

[3. Программно реализовать геттеры и сеттеры собственного класса с использованием метода clone(). 8](#_Toc130750725)

# Задание 9.1

* 1. Реализовать проекты с программами рисунков 9.1 и 9.2.

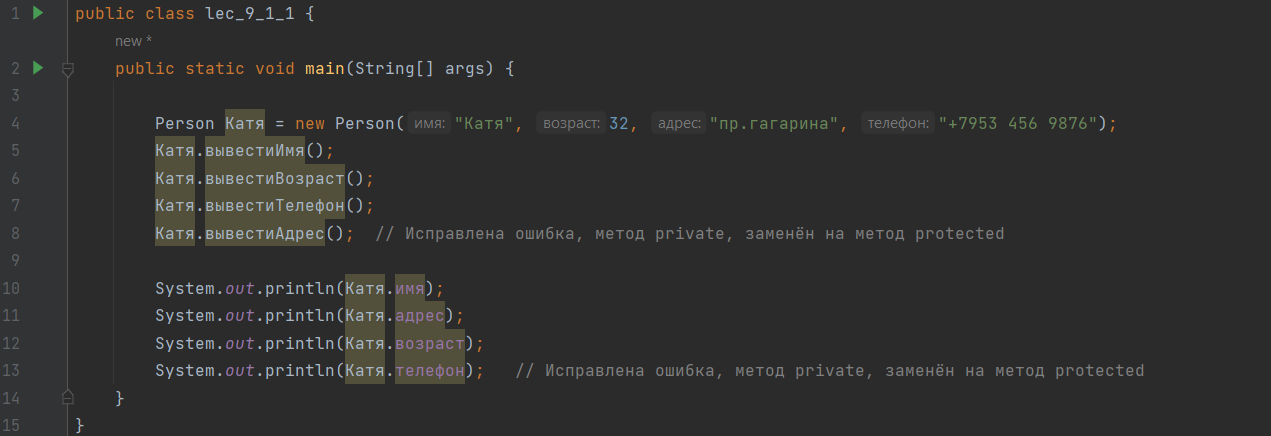


Рисунок 1. Реализация программы (с внесёнными исправлениями) из рисунка 9.1.

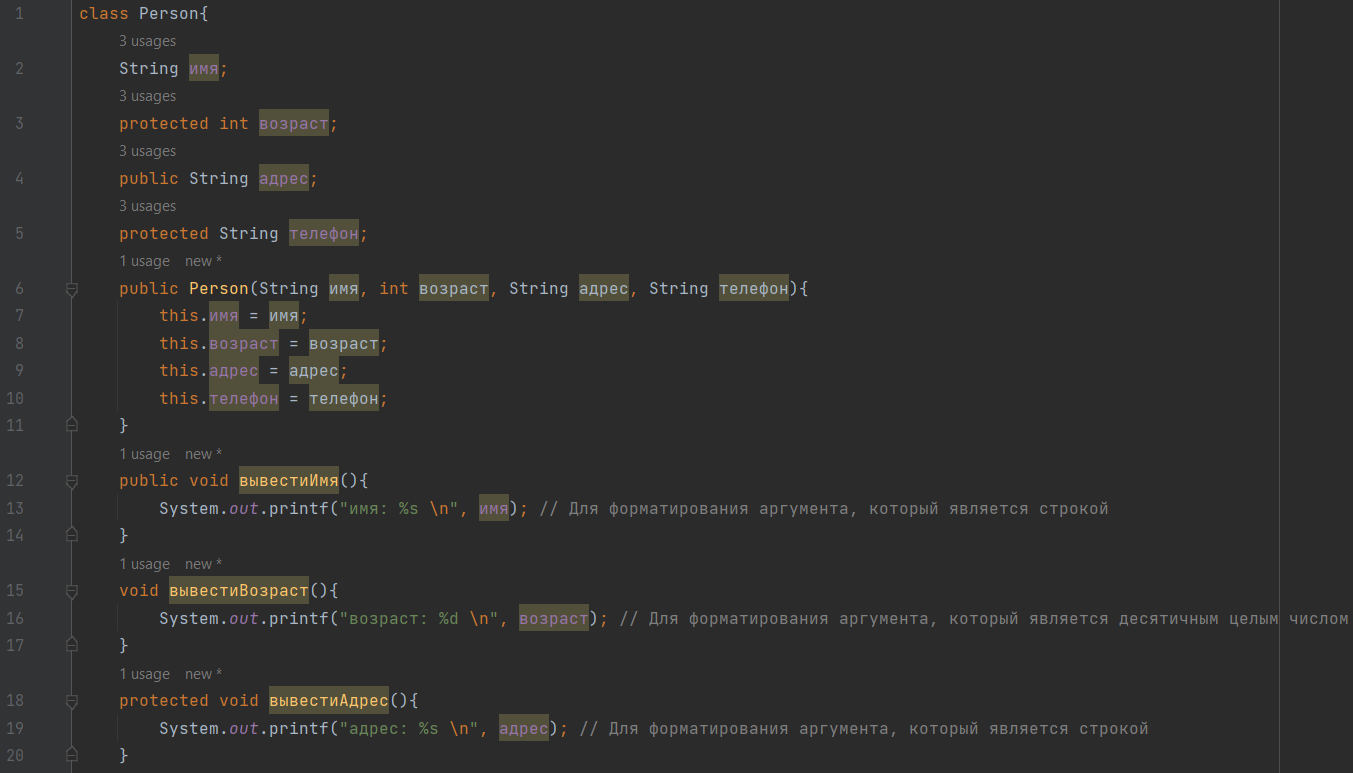
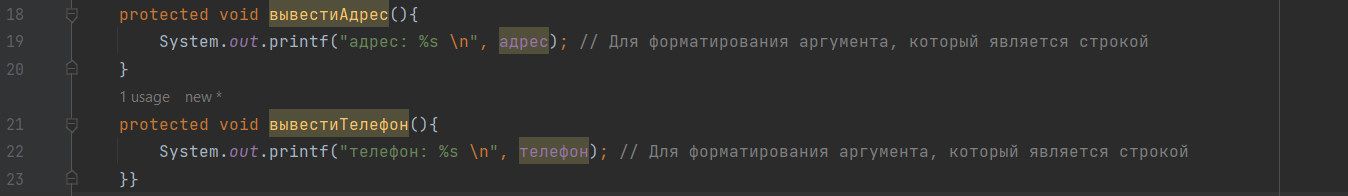
 

Рисунок 2. Реализация программы (с внесёнными исправлениями) из рисунка 9.1.

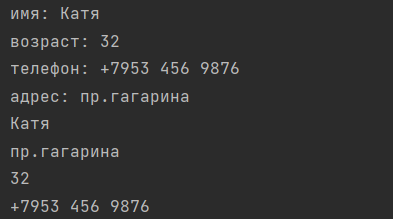


Рисунок 3. Результат работы исправленной программы из рисунка 9.1.

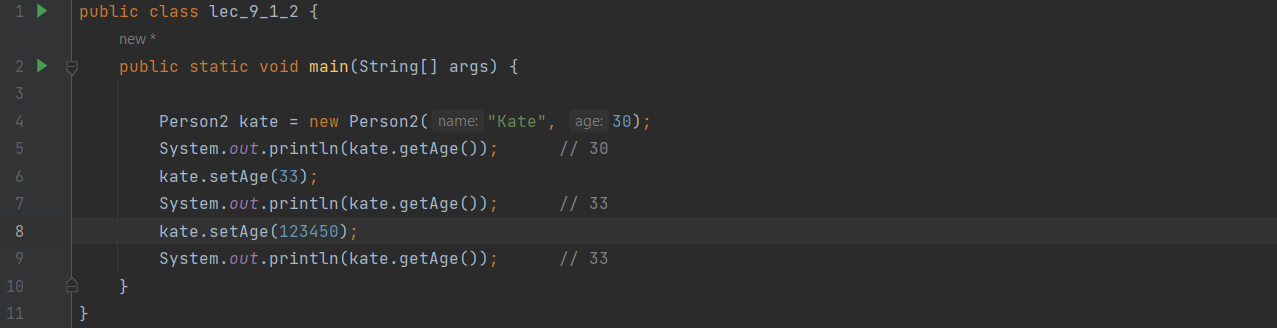
а

Рисунок 4. Реализация программы из рисунка 9.2.

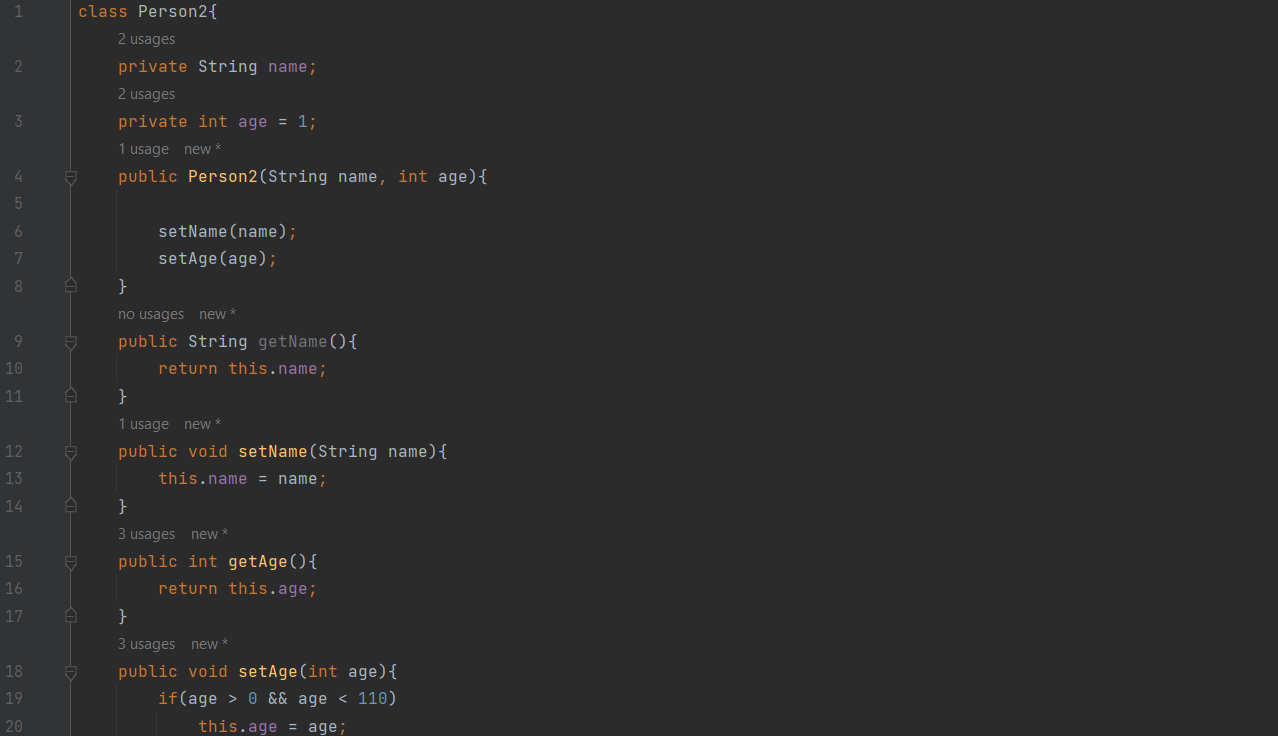


Рисунок 5. Реализация программы из рисунка 9.2.

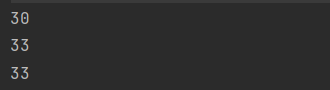


Рисунок 6. Результат работы программы из рисунка 9.2.

* 1. Определить (идентифицировать) причины ошибок проекта рисунка 9.1 и исправить их.

Была выявлена ошибка, связанная с методом private, он ограничивал доступ к переменным и методам. Данная ошибка была исправлена путём замены метода private, на метод protected/

Исправлена ошибка, связанная с форматированием аргумента, в строках 13, 19 и 22 %d изменён на %s.

* 1. В проекте рисунка 9.2 объяснить работу кодов строчек 6-10 и добиться вывода другого значения возраста по методу **.setAge**.

Согласно заданному условию, представленному в строках 6-10, при превышении присваиваемым значением от-метки в 110 – значение будет проигнорируемо. Для изменения возраста сеттером достаточно в доступном диапазоне задать произвольное значение.

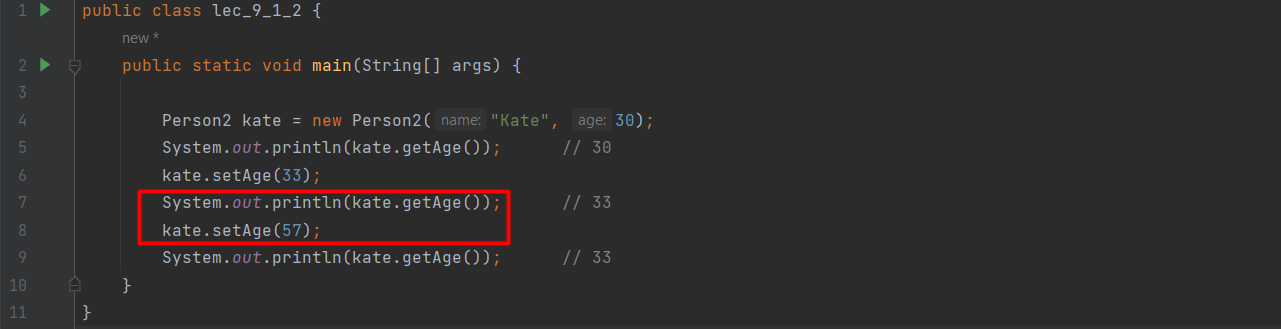


Рисунок 7. Изменение возраста.



Рисунок 8. Вывод с изменённым возрастом.

* 1. Разобраться в коде, приведённом в <https://3ty.ru/languages/java/gettery-i-settery-v-java> и реализовать его в соответствующем проекте.

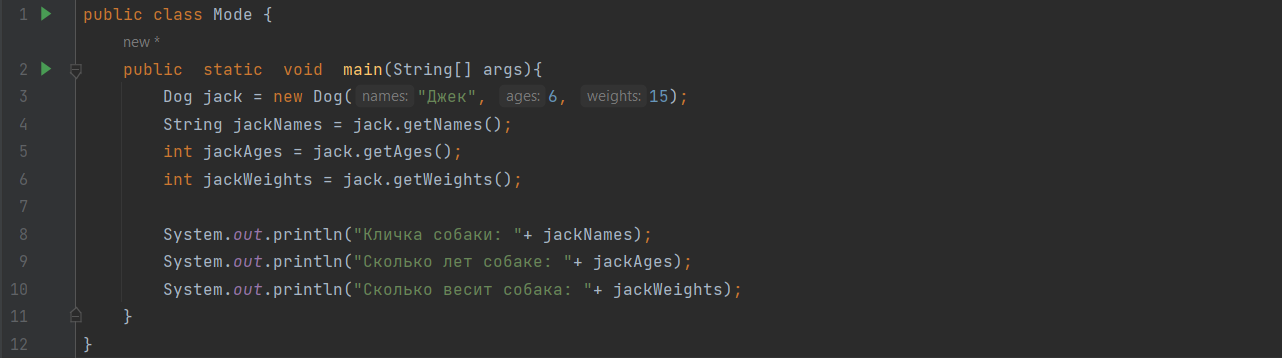


Рисунок 9. Реализация проекта Dog.

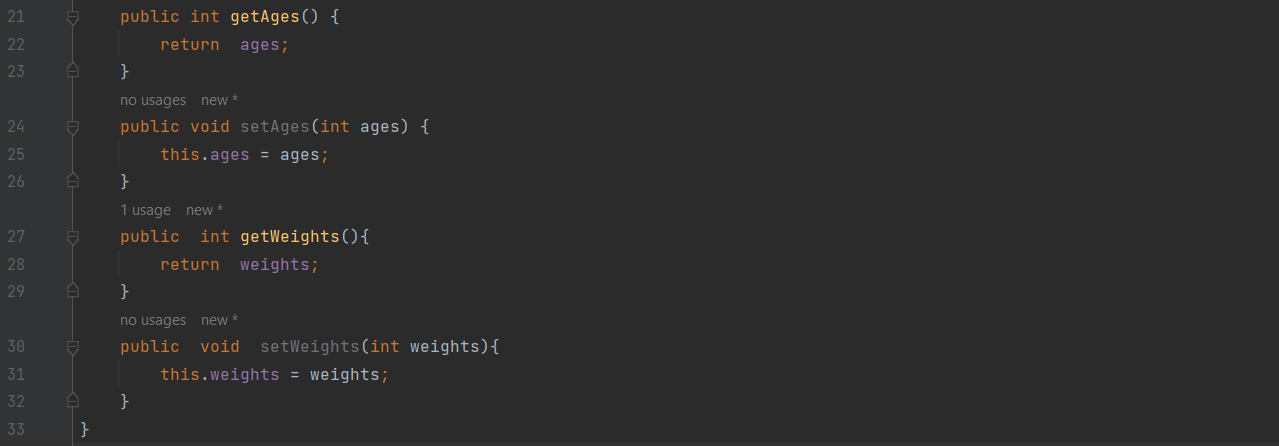


Рисунок 10. Реализация проекта Dog.

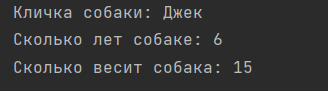


Рисунок 11. Результат работы проекта Dog.

* 1. Сформулировать преимущества и недостатки инкапсуляции

Преимущества инкапсуляции:

1. Повышает безопасность внутреннего состояния объекта, скрывая его от внешнего мира.
2. Повышает модульность и ремонтопригодность, упрощая изменение реализации, не затрагивая другие части кода.
3. Включает абстракцию данных, позволяя рассматривать объекты как единое целое.
4. Позволяет легко добавлять новые методы и поля, не затрагивая существующий код.
5. Поддерживает объектно-ориентированный принцип сокрытия информации, что упрощает изменение реализации, не затрагивая остальной код.

Недостатки инкапсуляции:

1. Может привести к увеличению сложности, особенно при неправильном использовании.
2. Может затруднить понимание того, как работает система.
3. Может ограничить гибкость реализации.

# Задание 9.2

1. Программно реализовать ошибки и их исправление из раздела 9.3.3;

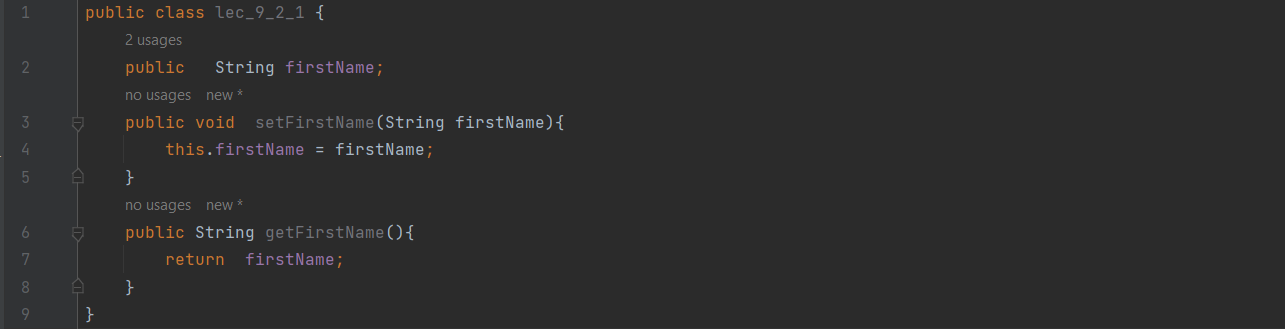


Рисунок 12. Скрипт "Ошибка 1".

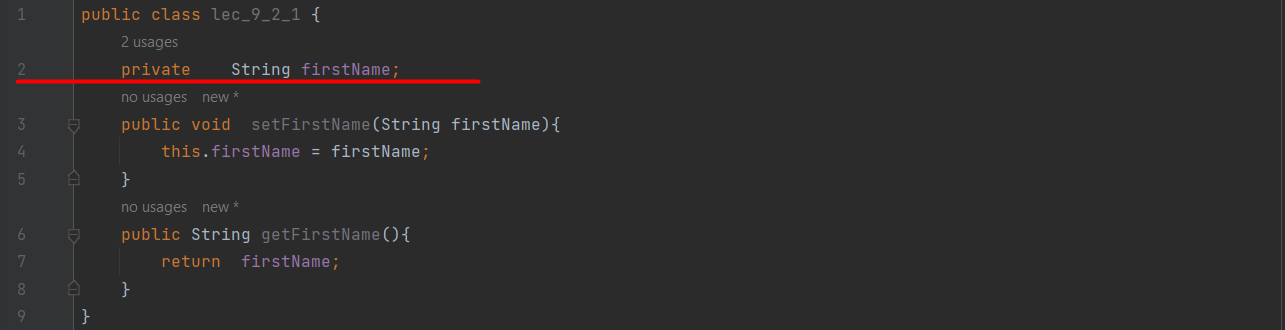


Рисунок 13. Cкрипт "Ошибка 1" с внесёнными исправлениями.

Первая ошибка — методы геттеры и сеттеры для атрибутов с режимом доступа public. Мы меняем режим доступа на private.

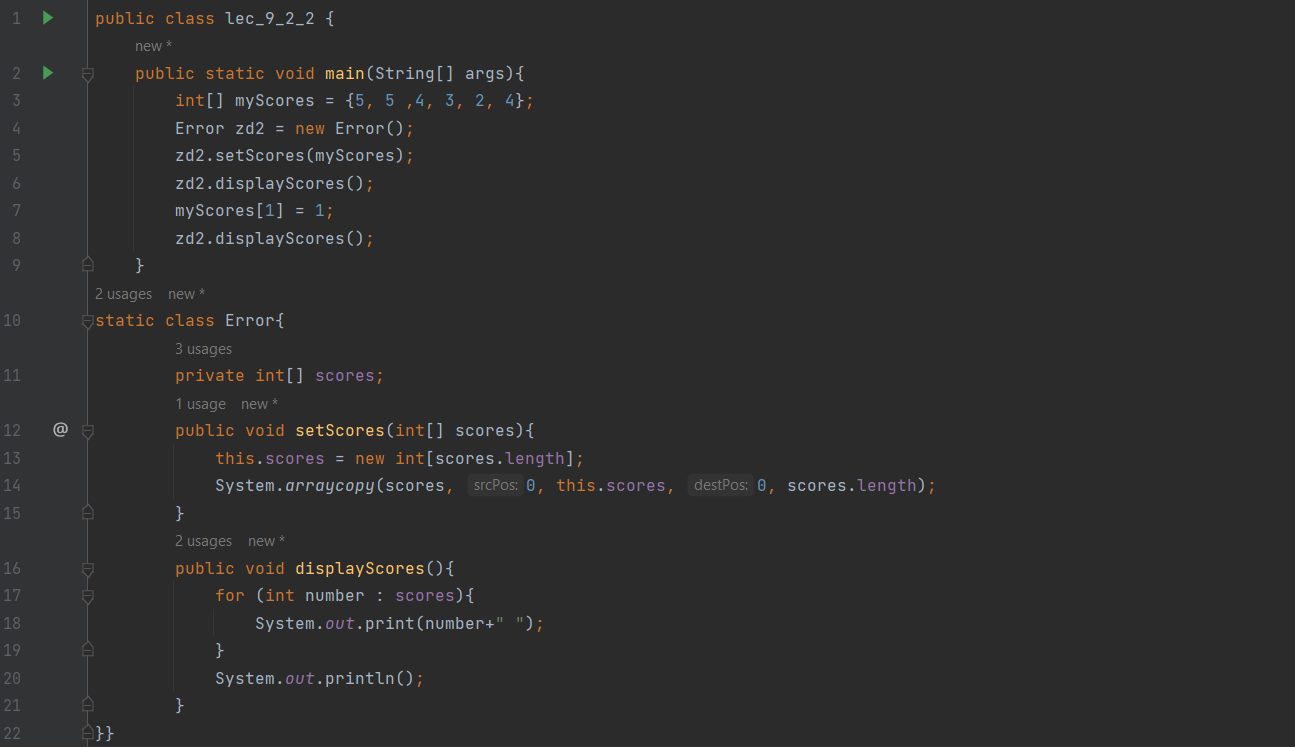


Рисунок 14. Скрипт "Ошибка 2" с внесёнными исправлениями.

Рисунок 15. Скрипт "Ошибка 3", с внесёнными исправлениями.

Вторая, а также третья ошибка — работа методов геттеров и сеттеров при работе с массивами. Мы вместо возврата массива и присвоения ему значения, будем делать это с копией массива.

1. Разобрать разделы 9.3.5.1 и 9.3.5.2

Геттеры и сеттеры для String

String — это immutable-тип. Это означает, что после создания объекта этого типа, его значение нельзя изменить. Любые изменения будут приводить к созданию нового объекта String.

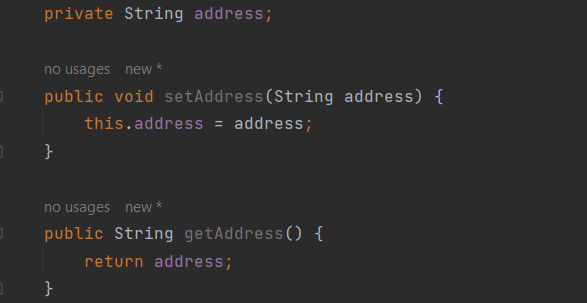


Рисунок 16. Пример использования геттеров и сеттеров с String.

Геттеры и сеттеры для объектов типа Date

Т.к. объекты класса java.util.Date являются изменяемыми, то внешние классы не должны иметь доступ к их оригиналам. Данный класс реализует метод clone() из класса Object, который возвращает копию объекта, но использовать его для этих целей не стоит.

Поскольку Date не является окончательным классом, нет гарантии, что метод clone() возвратит объект, класс которого именно java.util.Date: он может вернуть экземпляр ненадежного подкласса, созданного специально для нанесения ущерба. Такой подкласс может, например, записывать ссылку на каждый экземпляр в момент создания последнего в закрытый статический список, а затем предоставить злоумышленнику доступ к этому списку. В результате злоумышленник получит полный контроль над всеми экземплярами копий. Чтобы предотвратить атаки такого рода, не используйте метод clone() для создания копии параметра, тип которого позволяет ненадежным сторонам создавать подклассы.

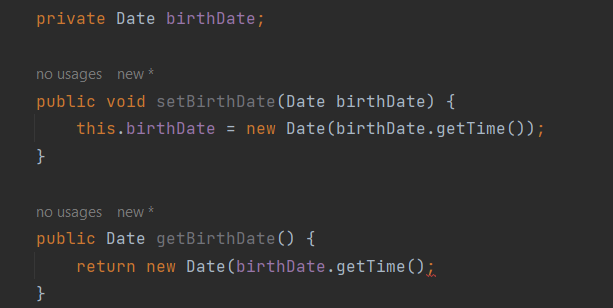


Рисунок 17. Пример использования геттеров и сеттеров с Date.

1. Программно реализовать геттеры и сеттеры собственного класса с использованием метода clone().

Рисунок 18. Реализация геттер и сеттер с использованием метода clone().

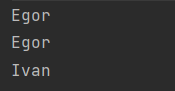


Рисунок .Вывод программы.

Класс Person содержит конструктор, метод геттер и метод сеттер а так же метод clone(), взятый из лекции.