**7.9 Домашняя работа**

Ваша работа принята

Задача

**Цель задания**

* поработать с расширениями;
* поработать с object и companion object;
* поработать с enum и sealed class.

**Что нужно сделать**

1. Создайте перечисление (enum) Ammo для видов патронов.

Свойства:

* урон (Int);
* шанс критического урона (Int);
* коэффициент критического урона (Int).

Методы:

* получение текущего урона (рассчитывает текущий урон в соответствии с шансом критического урона и коэффициентом критического урона).

В перечислении должно быть минимум три вида патронов с различными параметрами. Для того чтобы проверить, что текущий урон является критическим, используйте метод Random.nextInt() — и сравните результат с шансом урона.

2. Создайте extension-метод для типа Int. Метод должен вычислять, реализовался ли шанс, и возвращать Boolean. Используйте этот метод, чтобы проверить, что урон является критическим у Ammo. Также используйте этот метод для будущих проверок, связанных с вероятностью. Метод необходимо реализовать с помощью вызова Random.nextInt().

3. Создайте sealed-класс для вида стрельбы FireType. Создайте наследников: одиночный вид (singleton object), стрельба очередями (data class, в конструктор принимает параметр “размер очереди”).

4. Создайте абстрактный класс оружия AbstractWeapon со следующими характеристиками:

* максимальное количество патронов в магазине (Int, устанавливается через конструктор);
* вид стрельбы (FireType, устанавливается через конструктор);
* текущий список патронов (List<Ammo>, при создании объекта список пуст);
* факт наличия патронов (Boolean, вычисляется на основе свойства текущего списка патронов).

Оружие имеет методы:

* создание патрона — абстрактный метод, возвращающий объект типа Ammo;
* перезарядка — текущему списку патронов присваивается новый список, который создаётся с использованием метода создания патрона с учётом максимального количества патронов в магазине;
* получение патронов для выстрела — из текущего списка получаются патроны в соответствии с типом стрельбы для выстрела (количество патронов в текущем списке должно уменьшиться).

5. Создайте синглтон-объект Weapons, в нём должны быть методы, создающие конкретные виды оружия. Методы должны создавать анонимные объекты, наследующиеся от класса AbstractWeapon с указанием необходимых свойств и реализацией абстрактного метода. Создайте минимум четыре метода для различных видов оружия.

Пример работы: *val pistol: AbstractWeapon = Weapons.createPistol().*

6. Создайте интерфейс Warrior. Cвойства:

* isKilled(Boolean);
* шанс избежать попадания(Int).

Методы:

* атаковать — на вход принимает врага (Warrior);
* понести урон — на вход принимает урон (Int).

7. Создайте абстрактный класс AbstractWarrior, реализуйте интерфейс Warrior. Свойства:

* максимальный уровень здоровья (Int, устанавливается через конструктор);
* шанс избежать попадания (override, Int, устанавливается через конструктор);
* точность — вероятность попадания (Int, устанавливается через конструктор);
* оружие (AbstractWeapon, устанавливается через конструктор);
* текущий уровень здоровья (Int, при создании объекта равен максимальному уровню здоровья).

Методы:

* Атаковать (override) — если патроны в оружии закончились — перезарядить оружие и не стрелять в этом ходу. Иначе получить патроны для выстрела. Для каждого патрона проверить, попадает ли он в соответствие с точностью воина и шансом избежать попадания врагом.  
  Для патронов, которые попадают во врага, посчитать суммарный урон. Нанести урон врагу.
* Понести урон (override) — отнять от текущего уровня жизни урон.

8. Создайте конкретные классы для воинов, наследуемых от AbstractWarrior.

Сделайте несколько классов воинов, разделите их, например, по старшинству. У более старшего воина должно быть больше жизней, большая точность, лучше оружие. *Примеры классов: General, Captain, Soldier.*

9. Создайте класс команды Team. Команда должна хранить список воинов (List<Warrior>).

При создании извне передаётся параметр — количество воинов в команде. Список воинов заполняется в соответствии с количеством c помощью приватного метода. В команду воины набираются по вероятности: чем выше вероятность, тем выше ранг.

*Пример: с вероятностью 10% создаётся генерал, иначе с вероятностью 40% создаётся капитан, иначе создаётся солдат.*

1o. Создайте sealed class состояние битвы Существуют четыре потомка:

* прогресс (содержит в себе текущие показатели команд для вывода в консоль в процессе битвы, например суммарное здоровье команды);
* команда 1 одержала победу;
* команда 2 одержала победу;
* ничья.

11. Создайте класс битвы Battle.

Этот класс отвечает за логику битвы.

Свойства:

* команда 1 (Team);
* команда 2 (Team);
* битва завершена (Boolean).

Методы:

* Получить состояние битвы(возвращает объект BattleState). У каждой команды проверяется, есть ли живые воины. Если у двух команд есть живые воины — возвращается состояние прогресса. Иначе либо победа одной из команд, либо ничья.
* Совершить следующую итерацию битвы. Воины двух команд перемешиваются (метод shuffled). Воины из каждой команды наносят друг другу урон по очереди, если они живы.

12. Создайте точку входа в программу.

* Пользователь вводит количество воинов в команде.
* Создаётся битва.
* Пока битва не завершена, происходит совершение следующей итерации.
* При итерациях выводится промежуточное состояние битвы.
* По окончании битвы выводится команда-победитель либо ничья.

**Критерии оценки**

* Названия переменных должны отражать суть данных, на которые они ссылаются.
* Переменные, которые не изменяются в программе, должны быть объявлены неизменяемыми.
* Различные выводы в консоль должны начинаться с новой строки.
* Методы должны иметь корректную область видимости. Например, если метод или свойство не должно быть доступно извне, то у них должен быть модификатор private.
* Названия классов и интерфейсов должны отражать их назначение.
* Классы, свойства и методы должны именоваться в соответствии с соглашениями <https://kotlinlang.org/docs/reference/coding-conventions.html#naming-rules>.
* Классы должны быть открытыми для наследования (open) только в случае необходимости.
* Методы должны быть открытыми для переопределения (open) только в случае необходимости.
* Выполнены все пункты задания.

**Рекомендации**

Разбивайте код на более мелкие функции, если этого требует ваша программа. Даже если это не указано в задании.

По желанию вы можете добавить в программу дополнительный функционал.

Настоятельно рекомендуем выполнять домашнее задание поэтапно, консультируясь с преподавателем.

Вы можете отправлять на проверку частично выполненное задание. Например, в три итерации: шаги 1–5, шаги 6–8, шаги 9–12. Таким образом, вам не нужно будет переделывать работу из-за проблем в начальных пунктах.

Если написанная программа по какой-то причине не работает и вы не можете разобраться, в чём дело, — воспользуйтесь отладчиком.

Основную информацию о нём вы можете найти тут:

<https://medium.com/@artem_shevchenko/android-studio-debugging-%D0%B1%D0%B0%D0%B7%D0%BE%D0%B2%D1%8B%D0%B5-%D0%BF%D0%BE%D0%BD%D1%8F%D1%82%D0%B8%D1%8F-%D0%B8-%D0%B2%D0%BE%D0%B7%D0%BC%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D0%B8-658ee6dcc641>

С помощью точек останова и просмотра информации о состоянии объектов вы сможете выявить ошибку в коде.

Мы будем рассматривать работу с отладчиком в одном из следующих модулей.

**Как отправить задание на проверку**

* Используйте репозиторий **[learning\_materials / android\_basic](https://gitlab.skillbox.ru/learning_materials/android_basic)**.
* Скачайте изменения в репозитории на локальную машину.
* Выполните домашнее задание в папке **Extensions\_Objects\_Enums**.
* Отправьте коммиты в удалённый репозиторий.

Формат сдачи

[Перейти на GitLab](https://gitlab.skillbox.ru/markov_aleksei_2/android_basic)

Начало формы

Оцените взаимодействие с преподавателем

Конец формы

**Проверка домашнего задания**

Общение с преподавателем продолжится в следующем уроке.7.9. Основы Kotlin: ООП, классы и объекты. Часть 313сообщений



Октавиан Прокопов

09 января 2021, 12:36

Здравствуйте!

И еще прошу помочь разобраться в вопросе: когда в классе Battle  battleIsOver была переменной, то в Main цикл уходил в бесконечность. Когда поменяли на функцию, то все стало работать корректно. Не могу понять почему не работало с переменной?

Без кода сказать не просто, но обычно с переменными решения сложнее, так как ее надо постоянно поддерживать в актульном состоянии, не забыть при любом изменении поменять ее значение,а вот с функцией попроще, она обычно берет текущее состояние нужных полей и вычисляет значение в рантайме. Это не всегда оптимально, но ошибок меньше.

Проверил работу, все хорошо! Модуль принят.

Отлично что подсчет общего состояния здоровья команды вынесли в отдельный метод. Чтобы улучшить структуру проекта, можно разнести файлы по пакетам, например в пакет warriors можно вынести все классы касающиейся бойцов, в weapons - все что касается оружия. Так будет проще разбираться с проектом. Объект Weapons можно вынести в отдельный файл. В большинстве случаев один файл для одного класса - улучшает читаемость кода.

С уважением, Октавиан



Алексей Марков

08 января 2021, 14:05

Добрый день, Октавиан!

Прошу проверить домашнюю работу. Добавил задания 11-12.

<https://gitlab.skillbox.ru/markov_aleksei_2/android_basic/-/tree/master/Extensions_Objects_Enums/app/src/main/java/com/markov/extensions_objects_enums>

И еще прошу помочь разобраться в вопросе: когда в классе Battle  battleIsOver была переменной, то в Main цикл уходил в бесконечность. Когда поменяли на функцию, то все стало работать корректно. Не могу понять почему не работало с переменной?

С уважением, Марков Алексей.



Октавиан Прокопов

24 декабря 2020, 00:37

Здравствуйте!

Отлично, буду ждать доработки, мысли или вопросы :)

С уважением, Октавиан



Алексей Марков

23 декабря 2020, 14:19

Добрый день, Октавиан!

Я тоже рад знакомству! Замечания посмотрю и исправлю.

С уважением, Алексей Марков.



Октавиан Прокопов

23 декабря 2020, 11:00

Здравствуйте, Алексей!

Рад знакомству! С данного момента буду вашим проверяющим преподавателем :)

По 9 и 10 пунктам все верно, принимается.

Немного смущает метод getAmmo(), при вызове на пустом магазине он вызовет ошибку, это стоит иметь в виду. Я так понял метод атаки за раз выпускает весь магазин в противника? В принципе можно и так, только тогда нет смысла в FireType, все выстрелы будут одной очередью.

С уважением, Октавиан



Алексей Марков

22 декабря 2020, 17:00

Добрый день, Октавиан!

Прошу проверить домашнее задание, добавил задания 9-10

<https://gitlab.skillbox.ru/markov_aleksei_2/android_basic/-/tree/master/Extensions_Objects_Enums/app/src/main/java/com/markov/extensions_objects_enums>

С уважением, Алексей Марков.



Виктор Кузьмин

13 декабря 2020, 16:15

Здравствуйте,

Отлично идете по заданиям!

Все правильно делаете. Можно доделывать задание и дебажить, если будет необходимость...

С уважением, Виктор.



Алексей Марков

11 декабря 2020, 18:29

Добрый день, Виктор!

Прошу проверить домашнее задание, добавил пункты 6 - 8.

<https://gitlab.skillbox.ru/markov_aleksei_2/android_basic/-/tree/master/Extensions_Objects_Enums/app/src/main/java/com/markov/extensions_objects_enums>

С уважением, Алексей Марков.



Виктор Кузьмин

19 ноября 2020, 12:45

Здравствуйте,

Все отлично сделано.

Единственное, здесь маленькая поправка:

fun getAmmo(): Ammo {  
 currentListAmmo.*last*() *//****Это лишнее. Это просто лишний вызов.***return currentListAmmo.*removeLast*() *//у вас здесь уже патрон возвращается, при удалении элемента или Исключение (Exception). Но мы ошибки и баги будем отлавливать к концу программы, если что. Тут, возможно, в метод придется условие добавить, чтобы программа не вылетела. Но мы до этого дойдем, чтоб вы самостоятельно ошибки поотлавливали...*}

Можно следующие шаги делать!

С уважением, Виктор.



Алексей Марков

18 ноября 2020, 21:42

Добрый день, Виктор!

Загрузил проект, прошу проверить домашнюю работу (задания 1-5).

<https://gitlab.skillbox.ru/markov_aleksei_2/android_basic/-/tree/master/Extensions_Objects_Enums>

С уважением, Марков Алексей.



Алексей Марков

18 ноября 2020, 21:39

Добрый день, Виктор!

Загрузил проект, прошу проверить домашнюю работу (задания 1-5).

С уважением, Марков Алексей.



Виктор Кузьмин

18 ноября 2020, 13:46

Здравствуйте,

Прошу проверить домашнюю работу (задания 1 - 5)

У вас в репозитории только папка app с исходниками. Нужен проект Студии, чтобы его сбилдить и проверить!

Жду проект!

С уважением, Виктор.



Алексей Марков

17 ноября 2020, 17:32

Добрый день, Виктор!

Прошу проверить домашнюю работу (задания 1 - 5) <https://gitlab.skillbox.ru/markov_aleksei_2/android_basic/-/tree/master/Extensions_Objects_Enums/app/src/main/java/com/markov/extensions_objects_enums>

С уважением, Марков Алексей.