**Чек-лист к ДЗ**

**Свойства хорошего кода. Abstraction, Composition, Coupling, Cohesion**

*(заполняется спикером урока)*

Общая структура и оформление кода:

- [ ] Репозиторий GitHub содержит все необходимые файлы, включая README.md с инструкциями.

- [ ] Код оформлен единообразно и читабельно.

- [ ] Присутствуют комментарии и документация к методам, классам и модулям, если это необходимо.

Абстракция и ООП-практики:

- [ ] Код использует объектно-ориентированные принципы, такие как инкапсуляция, наследование и полиморфизм.

- [ ] Код следует принципам абстракции, скрывая сложные детали реализации.

- [ ] Классы и методы выполняют единственную обязанность и логически организованы.

Композиция и структура кода:

- [ ] Код использует композицию для построения сложных объектов из более простых.

- [ ] Проект содержит адекватное разделение на модули и пакеты.

- [ ] Зависимости между модулями минимальны и четко определены.

Связность и когезия:

- [ ] Классы имеют высокую когезию и ясно определенные задачи.

- [ ] Связность между классами минимальна, насколько это возможно.

- [ ] Код не содержит заметных антипаттернов, таких как жесткая связность или God Object.

Безопасность и конфиденциальность:

- [ ] Конфиденциальные данные (такие как токены и ключи API) отсутствуют в коде.

- [ ] (дополнительные но не обязательные требования) Используется система управления конфигурацией для чувствительных данных (секретов и токенов).

Тестирование:

- [ ] (дополнительные но не обязательные требования) Присутствуют автоматизированные тесты для ключевых функций бота (если требуется).

- [ ] (дополнительные но не обязательные требования) Тесты полные и проверяют различные сценарии использования бота.

Функциональность бота:

- [ ] Бот корректно реагирует на команды и сообщения.

- [ ] Функции бота соответствуют заявленной теме и функциональности.

- [ ] Проверено соответствие бота заданным пользовательским сценариям.

Дополнительные пункты:

- [ ] Работоспособность бота проверена в реальных условиях.

- [ ] Инструкции и команды бота информативны и понятны конечному пользователю.

**Само ДЗ для вашего понимания:**

В домашнем задании к этому уроку необходимо разработать функциональный телеграм-бот, исходя из изученных принципов хорошего кода, таких как абстракция, композиция, связность и когезия. Проверьте устойчивость, надежность и возможность переиспользования компонентов вашего бота.

Примеры тем для телеграм-бота: [Примеры тем для телеграм-бота. Свойства хорошего кода](https://docs.google.com/document/d/1G3SF0zb8nWYHy-uTRgfzHGrrQx9xEgk-gkFN49FzIOU/edit?usp=sharing)

Этапы выполнения домашнего задания: [Описание домашнего задания. Свойства хорошего кода](https://docs.google.com/document/d/1APIiN0UQRlqc2um019i_qdw0bdFPVCNjOF5TND8KxXc/edit?usp=sharing)

На проверку нужно выслать ссылку на репозиторий GitHub, содержащий исходный код бота.

Пожалуйста, обеспечьте соблюдение стандартов хорошего кода. Комментарии и документация к коду помогут ускорить процесс проверки и улучшат его понимание.  
  
Ссылка на репозиторий, которая также может помочь в решение некоторых задач - <https://github.com/EkaterinaMikrukova/Java-OOP-PS>