Коллоквиум

Паттерны проектирования, ИСиТ-2

1. **Перечислите основные парадигмы объектно-ориентированного программирования(ООП).**

Инкапсуляция, наследование, полиморфизм, абстракция.

1. **Поясните парадигму ООП «абстракция».**

Абстракция - выделение важных характеристик объекта и игнорирование прочих.

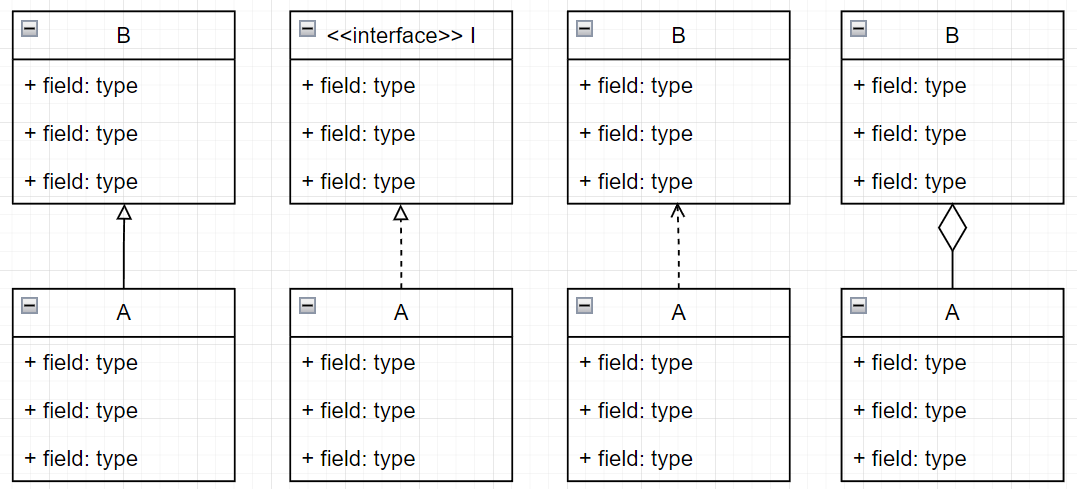
1. **Поясните принцип «Бритва Оккама».**

Всякое простое объяснение является наиболее вероятным.

1. **Поясните понятие ООП «интерфейс».**

Интерфейс - набор методов, доступных для взаимодействия с объектом.

1. **Нарисуйте UML-диаграмму: класс A является производным от класса В.**
2. **Нарисуйте UML-диаграмму: класс А реализует интерфейс I.**
3. **Нарисуйте UML-диаграмму: класс А использует тип класса B.**
4. **Нарисуйте UML-диаграмму: объект класса А содержит объект класса B.**

****

1. **Переведите с английского языка слово pattern.**

Паттерн, шаблон.

1. **Дайте простое определение (максимум 4 слова) понятия паттерна.**

Повторяемое архитектурное решение.

1. **Что такое «реализация паттерна»?**

Реализация паттерна - применение паттерна в конкретном контексте.

1. **Поясните понятие «идиома».**

Идиома - специфичный для языка программирования паттерн.

1. **Поясните понятие «фреймворк».**

Фреймворк - структура для разработки приложений.

1. **Поясните (расшифруйте) аббревиатуру SOLID.**

SOLID - **S**ingle Responsibility, **O**pen/Closed, **L**iskov Substitution, **I**nterface Segregation, **D**ependency Inversion.

1. **Дайте простое определение (максимум 5 слов) SOLID.**

Принципы проектирования ООП.

1. **Поясните кратко Single Responsibility Principe (максимум 7 слов).**

На каждый объект возложена одна единственная ответственность.

1. **Поясните кратко Open/Close Principe (максимум 7 слов).**

Сущности открыты для расширения, закрыты для модификаций.

1. **Поясните кратко Liskov Substitution Principe (LSP) (максимум 7 слов).**

Наследники заменяют объекты без изменения свойств программы.

1. **Поясните кратко 1 правило LSP.**

Функции, использующие базовый тип, должны работать с подтипами.

1. **Поясните кратко 2 правило LSP.**

Подклассы должны расширять, а не заменять свойства базового класса.

1. **Поясните кратко 3 правило LSP.**

Подклассы должны уточнять поведение базового класса.

1. **Поясните кратко 4 правило LSP.**

Новые исключения в подклассах должны быть подмножеством исключений базового класса.

1. **Поясните кратко 5 правило LSP.**

Подклассы должны быть заменяемыми на базовый класс без изменения корректности.

1. **Поясните кратко 6 правило LSP.**

Контракты подклассов должны быть согласованы с контрактами базового класса.

1. **Поясните кратко 7 правило LSP.**

Постоянство свойств и инвариантов базового класса должно быть сохранено в подклассах.

1. **Поясните кратко Interface Segregation Principle (максимум 7 слов).**

Много специализированных интерфейсов лучше, чем один универсальный.

1. **Поясните кратко Dependency Inversion Principle максимум 7 слов).**

Зависимость от абстракций.

1. **Перечислите наименование 3 группы паттернов. Укажите количество паттернов в каждой группе.**

Порождающие (5), структурные (7), поведенческие (11).

1. **Библиотека классов реализует Abstract Factory. Что библиотека предоставляет программисту-пользователю? Что разрабатывает программист-разработчик библиотеки.**

Abstract Factory: программист-пользователь получает интерфейс для создания семейств связанных объектов, программист-разработчик реализует конкретные фабрики.

1. **Библиотека классов реализует Facade. Что библиотека предоставляет программисту-пользователю? Что разрабатывает программист-разработчик библиотеки?**

Facade: программист-пользователь получает упрощенный интерфейс для работы со сложной подсистемой, программист-разработчик реализует упрощенный интерфейс, инкапсулируя сложную логику.

1. **Библиотека классов реализует Decorator. Что библиотека предоставляет программисту-пользователю? Что разрабатывает программист-разработчик библиотеки?**

Decorator: программист-пользователь получает возможность динамически добавлять новые обязанности к объектам, программист-разработчик реализует декораторы, расширяющие функциональность базовых объектов.

1. **Библиотека классов реализует Observer. Что библиотека предоставляет программисту-пользователю? Что разрабатывает программист-разработчик библиотеки?**

Observer: программист-пользователь получает возможность создания системы с механизмом подписки на события, программист-разработчик реализует объекты "наблюдателей" и "субъектов".