OTUS

Архитектура и шаблоны проектирования

Курс для разработчиков, которые хотят изучить основные паттерны проектирования и научиться применять их, находить им замену в сложных ситуация и научиться мыслить, как архитектор программного обеспечения

Длительность курса: 112 академических часов

1 Основные принципы

1	**ООП or not ООП**	реализовывать принципы, составляющие ООП и без ООП; проанализировать принципы вне скрытой реализации языков, использующих подход ООП.
2	**SOLID (часть 1)**	описывать абстракции, устойчивые к изменениям, чтобы писать код, соотвествующий OCP принципу.
3	**SOLID (часть 2)**	вносить исправления в абстракции, чтобы повысить их устойчивость к изменениям. Домашние задания 1 Движение игровых объектов по полю. Цель: Выработка навыка применения SOLID принципов на примере игры "Танки". В результате выполнения ДЗ будет получен код, отвечающий за движение объектов по игровому полю, устойчивый к появлению новых игровых объектов и дополнительных ограничений, накладываемых на это движение. Описание игры по ссылке https://docs.google.com/document/d/19QXXaUEAIMkYsZZceSCkZ8jkkryMPpqJUotwV3GGIgQ/edit?usp=sharing Реализовать движение объектов на игровом поле в рамках подсистемы Игровой сервер.

4 **Многопоточность и архитектуры приложений**

показать основные архитектурные решения для приложений, использующих несколько потоков.

Домашние задания

1 Многопоточное выполнение команд

Цель: Разработка многопоточной системы выполнения команд на примере игры "Танки".

В результате выполнения ДЗ будет получен код, отвечающий за выполнение множества команд в несколько потоков, устойчивый к появлению новых видов команд и дополнительных ограничений, накладываемых на них.

Предположим, что у нас есть набор команд, которые необходимо выполнить. Выполнение команд организуем в несколько потоков.
Для этого будем считать, что у каждого потока есть своя потокобезопасная очередь.
Для того, чтобы выполнить команду, ее необходимо добавить в очередь. Поток читает очередную команду из очереди и выполняет ее.
Если выполнение команды прерывается выброшенным исключением, то поток должен отловить его и продолжить работу.
Если сообщений нет в очереди, то поток засыпает до тех пор, пока очередь пуста.

Последовательность шагов решения:

1. Реализовать код, который запускается в отдельном потоке и делает следующее
В цикле получает из потокобезопасной очереди команду и запускает ее.
Выброс исключения из команды не должен прерывать выполнение потока.
2. Написать команду, которая стартует код, написанный в пункте 1 в отдельном потоке.
3. Написать команду, которая останавливает цикл выполнения команд из пункта 1, не дожидаясь их полного завершения (hard stop).
4. Написать команду, которая останавливает цикл выполнения команд из пункта 1, только после того, как все команды завершат свою работу (soft stop).
5. Написать тесты на команду запуска и остановки потока.

2 Методологии разработки ПО

1	**Методологии разработки ПО**	объяснить различные подходы к разработке ПО, их особенности и ограничения.
2	**Что предшествует программированию**	объяснить понятие архитектуры; рассмотреть архитектурные стили.
3	**Общие шаблоны распределения ответственностей**	проанализировать функциональное разделение функционала; рассмотреть 9 шаблонов GRASP (они понадобятся для пояснения более широких понятий, таких как микросервисная архитектура).
4	**Процесс-всему голова**	построить бизнес-процесс на примере задачи; разделить на функциональные процессы; обсудить, как можно решить подобную задачу, какими подходами; разобраться в законе "Конвея".
5	**"Как это выглядит?" или как нарисовать процесс**	разобраться, как рисуется процесс, его составные части; использовать оба вида нотаций. Домашние задания 1 Создание модели любого процесса в нотациях EPC или BPMN Цель: Научиться создавать графическое отображение модели простейшего бизнес-процесса. 1. Выбрать какую-то из нотаций EPC или BPMN; 2. Выбрать любой процесс на усмотрение учащегося (из рабочих операций, бытовых и т.п.); 3. Устно или письменно кратко описать суть процесса-прототипа; 4. Начертить модель бизнес-процесса.
6	**Знакомство с "кирпичиками" построения систем**	рассмотреть на примере ракурсы, на которые смотрит архитектор ПО; строить схемы видов.
7	**Чистый код и рефакторинг**	видеть недостатки кода, смогут корректировать его, превращая в корректный удобный код.

3 Шаблоны проектирования - 1

1 **Команда**

применять шаблон;
построить процесс выполнения задачи с использованием шаблона "Команда".

Домашние задания

1 Макрокоманды

Цель: Цель: Научиться обрабатывать ситуации с точки зрения SOLID, когда требуется уточнить существующее поведение без модификации существующего кода.

Предположим, что у нас уже написаны команды MoveCommand и RotateCommand. Теперь возникло новое требование: пользователи в игре могут устанавливать правило - во время движение расходуется топливо, двигаться можно только при наличии топлива.

Реализовать новую возможность можно введя две новые команды.
CheckFuelCommand и BurnFuelCommand.
CheckFuelCommand проверяет, что топлива достаточно, если нет, то выбрасывает исключение CommandException.
BurnFuelCommand уменьшает количество топлива на скорость расхода топлива.

После этого мы можем три команды выстроить в цепочку.
CheckFuelCommand MoveCommand BurnFuelCommand

Чтобы это было прозрачно для пользователя реализуем Макрокоманду - специальную разновидность команды, которая в конструкторе принимает массив команда, а методе execute их все последовательно выполняет.

При повороте движущегося объекта меняется вектор мгновенной скорости. Напишите команду, которая модифицирует вектор мгновенной скорости, в случае поворота.
Постройте цепочку команд для поворота.

1. Реализовать класс CheckFuelComamnd и тесты к нему.
2. Реализовать класс BurnFuelCommand и тесты к нему.
3. Реализовать простейшую макрокоманду и тесты к ней. Здесь простейшая - это значит, что при выбросе исключения вся последовательность команд приостанавливает свое выполнение, а макрокоманда выбрасывает CommandException.
4. Реализовать команду движения по прямой с расходом топлива, используя команды с предыдущих шагов.
5. Реализовать команду для модификации вектора мгновенной скорости при повороте. Необходимо учесть, что не каждый разворачивающийся объект движется.
6. Реализовать команду поворота, которая еще и меняет вектор мгновенной скорости, если есть.

---

2 **Расширяемая**

узнать и научиться применять шаблон.

фабрика и IoC

Домашние задания

1 Реализация выбора подходящего метода сортировки (выбором, вставки, слиянием) набора данных с использованием абстрактной фабрики и описание применения шаблона в проекте

Цель: Получите навык работы с абстрактной фабрикой.

Данные задаются в файле. Результат также помещается в файл.
1. Выбрать массив размером 50 элементов.
2. Создать программу, которая в качестве входного параметра получает вариант сортировки (выбором, вставки, слиянием), имя файла со входным набором данных и имя файла с выходными данными.
3. Реализовать в программе абстрактную фабрику и конкретные фабрики, отвечающие за каждый вариант сортировки как продукты.
4. Программа записывает результаты в выходной файл данных. В содержании в пишется тип сортировки и результаты.
5. Если потребуется использовать абстрактную фабрику или фабричный метод в проектной работе, предоставить описание в текстовом файле в GitHub репозитории где конкретно и в какой роли используется этот шаблон.
6. нарисовать диаграмму классов.
Срок 2 недели от занятия
ДЗ сдается в виде ссылки на GitHub репозиторий с проектом.
По вопросам обращаться в Slack к студентам, преподавателям и наставникам в канал группы

---

3 **Стратегии разрешения зависимостей IoC**

применять шаблон "Singleton".

---

4 **Адаптер и мост**

узнать и научиться применять шаблоны "адаптер" и "мост"

Домашние задания

1 Адаптер для работы двух независимых программ.
Описание применения шаблона в проекте

Цель: 1. Вы напишете адаптер, чтобы связать функционал двух отдельных программ в единый процесс. Разберётесь с тем, как адаптер работает в случае вызова отдельных программ. Получите навыки работы с формальными и фактическими параметрами передачи данных.
2. Получите навык анализа системы - использовать или нет этот шаблон в проектной работе.

Написать простую консольную программу П1, с интерфейсом вызова И1, которая читает данные о двух матрицах A и B из файла F0, складывает матрицы и сохраняет результат A+B в другой файл F1.
Написать вторую консольную программу П2, которая

может генерить данные матриц A и B и писать их в файл с именем F2.
Чтобы она могла их просуммировать, следует сделать адаптер для программы П1, который позволит программе П2 вызвать П1.
1. Написать программу П1
2. Написать программу П2, включив туда адаптер вызова и использования программы П1
3. Написать автотест для проверки функционирования
4. Если потребуется использовать адаптер в проектной работе, предоставить описание в текстовом файле в GitHub репозитории где конкретно и в какой роли используется этот шаблон.
5. Нарисовать диаграмму классов.
Срок - 2 недели от занятия.
ДЗ сдается в виде ссылки на GitHub репозиторий с проектом.
По вопросам обращаться в Slack к преподавателям и наставникам в канал группы.

4 Архитектура ПО

1	**Создание микросервиса**	познакомиться с паттернами декомпозиции системы на микросервисы.
2	**Интеграция программного обеспечения**	объяснить, как работать с интеграцией программного обеспечения и какие существуют подходы.
3	**Системы обмена сообщениями**	рассмотреть архитектурные концепции построения систем обмена сообщений.
4	**DevOps**	рассмотреть процесс создания и поставки ПО в рамках devops.
5	**Микросервисная архитектура**	рассмотреть описание, характерные свойства и характеристики; оценить порядок перехода от "монолита".

5 Шаблоны проектирования - 2

1 **Итератор**

объяснить, как инкапсулировать этот процесс;
проанализировать один общий интерфейс для перебора, множество реализаций;
объяснить как работает итератор;
рассмотреть на примере.

Домашние задания

1 Генератор чисел Фибоначчи и описание применения шаблона в проекте

Цель: Получите навыки применения шаблона "итератор" и знания формировании чисел Фибоначчи.

1. Создать программу, которая генерит числа Фибоначчи в указанном диапазоне в указанный файл. Предусмотреть возможность движения в обратном направлении (например, с использованием формулы Binet).
2. Реализовать в программе абстрактную фабрику и конкретные фабрики, отвечающие за каждый вариант сортировки как продукты.
3. Программа записывает результаты в выходной файл данных.
4. Если потребуется использовать Итератор в проектной работе, предоставить описание в текстовом файле в GitHub репозитории, где конкретно и в какой роли используется этот шаблон.
5. Нарисовать диаграмму классов.

Срок - 2 недели от занятия
ДЗ сдается в виде ссылки на GitHub репозиторий с проектом.
По вопросам обращаться в Slack к студентам, преподавателям и наставникам в канал группы.

---

2 **Состояние**

объяснить:
состояние не так однозначно, как кажется;
диаграмма состояний и переходов;
обзор конечных автоматов;
простая прямая реализация;
расширение функциональности;
от простой реализации объектов к интерфейсам.

---

3 **Цепочка обязанностей**

изучить шаблон и научиться его применять.

Домашние задания

1 Парсер файлов в зависимости от их типа и описание применения шаблона в проекте

Цель: Получите навыки применения шаблона "цепочка ответственности"

Программа, реализующая алгоритм, получает на вход список файлов. И каждый попадает в обработку алгоритма.

На вход алгоритма передаётся ряд файлов, которые имеют различный тип (Xml, JSON, CSV, txt)
Требуется создать цепочку обработки этих файлов, где отдельный обработчик отвечает за обработку конкретного типа документа.
Обработчик логирует получение подходящего ему файла в виде "обработчик TXT получил файл filename.txt" и копирует содержимое в выходной файл.

требуется:
1. создать программу, где на вход подаётся путь файла со списком обрабатываемых файлов и путь выходного файла;
2. реализовать алгоритм обработки с помощью шаблона "Цепочка ответственности";
3. нарисовать диаграмму классов;
Если потребуется использовать шаблон в проектной работе, предоставить описание в текстовом файле в GitHub репозитории где конкретно и в какой роли используется этот шаблон.
Срок - 2 недели от занятия
ДЗ сдается в виде ссылки на GitHub репозиторий с проектом.
По вопросам обращаться в Slack к студентам, преподавателям и наставникам в канал группы

---

4 **Заместитель**

научиться делегировать;
научиться давать команду, чтобы "Заместитель" (Proxy) выполнил задачу.

---

5 **Декоратор**

применять шаблон.

---

6 **Шаблонный метод**

объяснить, что цели можно достичь разными путями, порой важно оперировать не объектами, а процессами;

проанализировать:
от бизнес процесса к абстрактным общим шагам;
декомпозиция;
временная диаграмма;
один процесс, разные шаги;
необходимость контроля процесса;
методы-перехватчики.

Домашние задания

1 Реализация выбора подходящего метода матричных операций с применением шаблонного метода и описание применения шаблона в проекте

Цель: Получите навыки в программировании алгоритмов матричных операций, применении шаблонного метода.

Есть несколько операций над матрицами:
1. транспонирование матрицы;

		2. сложение матриц; 3. нахождение определителя матрицы. Написать программу, которая выполняет следующее: 0. На входе получает название входного файла, выходного файла и вид операции; 1. Получает данные из файла; 2. Выполняет указанную операцию над данными; 3. Формирует данные для вывода в необходимом формате; 4. Записывает данные в выходной файл; 5. Если потребуется использовать Шаблонный метод в проектной работе, предоставить описание в текстовом файле в GitHub репозитории где конкретно и в какой роли используется этот шаблон; 6. Нарисовать диаграмму классов. Срок - 2 недели от занятия ДЗ сдается в виде ссылки на GitHub репозиторий с проектом. По вопросам обращаться в Slack к студентам, преподавателям и наставникам в канал группы
7	**Интерпретатор**	изучить шаблон и научиться его применять.

6 Проектная работа

1	**Вводное занятие по проектной работе**	определиться с финальной архитектурой проекта; выбрать и обсудить тему проектной работы; спланировать работу над проектом; ознакомиться с регламентом работы над проектом. Домашние задания 1 Обсуждаем вопросы проектирования, корректируем принятые решения Цель: выбрать темы; закрепить тему в чат с преподавателем. 1. выбрать стиль архитектуры; 2. описать бизнес-процессы; 3. отрисовать функциональные процессы; 4. копонентная схема; 5. информационная схема; 6. используемые шаблоны; 7. описание интеграций.
2	**Консультация по проектам и домашним заданиям**	получить ответы на вопросы по проекту, ДЗ и по курсу.
3	**Защита проектных работ**	защитить проект и получить рекомендации экспертов.