**Правительство Российской Федерации**

**Санкт-Петербургский государственный университет**

**Р А Б О Ч А Я П Р О Г Р А М М А**

**УЧЕБНОЙ ДИСЦИПЛИНЫ**

Проектирование и архитектура программного обеспечения

Software Architecture and Design

**Язык(и) обучения**

русский

Трудоемкость в зачетных единицах: 3

Регистрационный номер рабочей программы: 003669

Санкт-Петербург

2024

**Раздел 1. Характеристики учебных занятий**

**1.1. Цели и задачи учебных занятий**

Цель изучения дисциплины: ознакомление обучающихся с основными принципами проектирования крупных программных продуктов и комплексов.  
Задачи обучения:

1. приобретение знаний об архитектуре программного обеспечения в целом, роли архитектуры в жизненном цикле программного обеспечения, целях и содержании деятельности архитектора;
2. приобретение навыков объектно-ориентированного проектирования, проектирования информационных систем и распределённых приложений;
3. изучение основных архитектурных стилей, типовых приёмов проектирования, шаблонов проектирования и архитектурных шаблонов;
4. получение представления о языке UML и других формальных языках как основных средствах описания архитектуры программного обеспечения.

**1.2. Требования к подготовленности обучающегося к освоению содержания учебных занятий (пререквизиты)**

Программа курса предназначена для обучающихся 3 курса бакалавриата, освоивших программу курсов «Основы программирования», «Практикум на ЭВМ» и «Разработка программного обеспечения» (или аналогичных им). Максимальная эффективность программы будет обеспечена при условии, что обучающийся имел опыт разработки промышленных или исследовательских программных продуктов.

**1.2.1. Требуемые компетенции**

* ОПК-6 – способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов;
* ПКП-5 – способен использовать современные системные программные средства: операционные системы, операционные и сетевые оболочки, сервисные программы;
* УК-4 – Способен осуществлять деловую коммуникацию в устной и письменной формах на государственном языке Российской Федерации и иностранном(ых) языке(ах).

**1.3. Перечень результатов обучения (learning outcomes)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Наименование категории (группы) компетенций | Код и наименование компетенции | Планируемые результаты обучения, обеспечивающие формирование компетенции | Код индикатора и индикатор достижения универсальной компетенции |
|  | 1 | 2 | 3 | 4 |
| 1 | Профессиональные компетенции (практические) | ПКП-1. Способен разрабатывать требования к программной системе | * Умеет выполнить анализ требований и построить архитектурную модель требований в одной из стандартных нотаций | ПКП-1.03669.1. Создаёт диаграмму случаев использования UML, диаграмму активностей UML по данному списку пользовательских требований. |
| 2 | Профессиональные компетенции (практические) | ПКП-2. Способен проектировать архитектуру программной системы с учётом требований и ограничений в условиях цифровой экономики, используя современные инструментальные среды и стандарты документирования | * Умеет выбрать архитектурный стиль и создать проект программной системы по данному набору требований | ПКП-2.003669.1. Создаёт архитектуру системы по данному заданию с использованием диаграмм UML. |
| 3 | Профессиональные компетенции (практические) | ПКП-3. Способен декомпозировать поставленную задачу на составные части, оценить трудоёмкость и риски реализации | * Владеет методами декомпозиции и проектированием высокоуровневой структуры системы | ПКП-3.003669.1. Создаёт компонентную модель системы по заданным функциональным требованиям. |
| 4 | Системное и критическое мышление | УК-1. Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач | * Умеет выполнить декомпозицию проектируемой системы. * Умеет искать необходимую для решения задачи информацию. * Умеет обоснованно выбрать существующие компоненты и технологии для решения задачи. * Умеет грамотно оценить варианты архитектуры. | УК 1.1. Анализирует задачу, выделяя ее базовые составляющие;  УК 1.2. Определяет информацию, необходимую для решения поставленной задачи;  УК 1.3. Осуществляет по различным запросам поиск информации, необходимой для решения поставленной задачи;  УК 1.4. Оценивает достоинства, недостатки и последствия вариантов решения поставленных задач;  УК 1.5. Грамотно, логично, аргументированно формирует собственные суждения, решения и оценки. |
| 5 | Разработка и реализация проектов | УК-2. Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений | * Способен выполнить анализ способов решения поставленной задачи и обоснованно выбрать используемые технологии. * Знает основные архитектурные шаблоны и стили. * Умеет предложить архитектурный стиль, подходящий для решения поставленной задачи. * Умеет планировать сопровождение систем, принимая во внимание эффекты архитектурной эрозии. | УК-2.1. Определяет круг задач в рамках поставленной цели;  УК-2.2. Предлагает способы решения поставленных задач;  УК-2.3. Оценивает соответствие способов решения цели проекта;  УК-2.4. Планирует реализацию задач в зоне своей ответственности с учетом имеющихся ресурсов и ограничений, действующих правовых норм;  УК-2.5. Выполняет задачи в зоне своей ответственности в соответствии с запланированными результатами и точками контроля; |
| 6 | Исследовательская и (или) предпринимательская деятельность | УКБ-1. Способен участвовать в разработке и реализации проектов, в т.ч. предпринимательских | * Способен выполнить анализ способов решения поставленной задачи и обоснованно выбрать используемые технологии. * Знает основные архитектурные шаблоны и стили. * Умеет предложить архитектурный стиль, подходящий для решения поставленной задачи. * Умеет планировать сопровождение систем, принимая во внимание эффекты архитектурной эрозии. * Владеет стандартами описания архитектуры. * Владеет языками средствами визуального моделирования. | УКБ-1.1. Определяет круг задач в рамках поставленной цели;  УКБ-1.2. Предлагает способы решения поставленных задач;  УКБ-1.3. Оценивает соответствие способов решения цели проекта;  УКБ-1.4. Планирует реализацию задач в зоне своей ответственности с учетом имеющихся ресурсов и ограничений, действующих правовых норм;  УКБ-1.5. Выполняет задачи в зоне своей ответственности в соответствии с запланированными результатами и точками контроля;  УКБ-1.6. Представляет результаты проекта;  УКБ-1.7. Предлагает возможности использования результатов проекта и/или совершенствования. |
| 7 | Работа с информацией | УКБ-3. Способен понимать сущность и значение информации в развитии общества, использовать основные методы получения и работы с информацией с учетом современных технологий цифровой экономики и информационной безопасности | * Владеет навыками поиска и анализа существующих компонентов, которые могут быть использованы при решении задачи. | УКБ-3.1. Находит и использует различные источники информации;  УКБ-3.2. Точно определяет тип и форму необходимой информации;  УКБ-3.3. Получает информацию и сохраняет ее в удобном для работы формате;  УКБ-3.4. Проверяет достоверность собранной информации. |

**1.3.2. Знания, умения и навыки, формируемые дисциплиной**

Результатами обучения являются знание содержания программы курса и представление о возможностях применения ее разделов в практической деятельности, а именно:

1. умение самостоятельно выбрать подход к проектированию программной системы, создать или описать имеющуюся архитектуру программного продукта;
2. знание достоинств и недостатков основных существующих подходов к проектированию ПО;
3. владение типовыми шаблонами проектирования объектно-ориентированных программ;
4. владение языком UML на уровне, достаточном для создания архитектурных диаграмм и понимания диаграмм, созданных коллегами;
5. умение оформить и изложить результат проектирования программного обеспечения в виде архитектурного документа или набора документов;
6. владение принципами проектирования распределённых приложений.

**1.4. Перечень и объём активных и интерактивных форм учебных занятий**

Практические занятия – 15 академических часов.

Основной формой обучения проектированию программного обеспечения являются лекционные и практические занятия в аудитории. Практические занятия проводятся в группах по два-три человека и включают элементы групповых проблемных работ, используют технологии «мозговой штурм» и включают в себя анализ реальных ситуаций профессиональной деятельности.

**Раздел 2. Организация, структура и содержание учебных занятий**

**2.1. Организация учебных занятий**

**2.1.1 Основной курс**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Трудоёмкость, объёмы учебной работы и наполняемость групп обучающихся | | | | | | | | | | | | | | | | | | |
| Код модуля в составе дисциплины,  практики и т.п. | Контактная работа обучающихся с преподавателем | | | | | | | | | | | | Самостоятельная работа | | | | Объём активных и интерактивных  форм учебных занятий | Трудоёмкость |
| лекции | семинары | консультации | практические  занятия | лабораторные работы | контрольные работы | коллоквиумы | текущий контроль | промежуточная  аттестация | итоговая аттестация | под руководством преподавателя | в присутствии  преподавателя | сам. раб. с использованием  методических материалов | текущий контроль (сам.раб.) | промежуточная аттестация (сам.раб.) | итоговая аттестация  (сам.раб.) |
| ОСНОВНАЯ ТРАЕКТОРИЯ | | | | | | | | | | | | | | | | | | |
| Форма обучения: очная | | | | | | | | | | | | | | | | | | |
| Семестр 7 | 15 |  |  | 15 |  |  |  |  | 1 |  |  |  | 67 |  | 10 |  | 10 | 3 |
|  | 1-30 |  |  | 1-30 |  |  |  |  | 1-30 |  |  |  | 1-1 |  | 1-1 |  |  |  |
| ИТОГО | 15 |  |  | 15 |  |  |  |  | 1 |  |  |  | 67 |  | 10 |  |  | 3 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Виды, формы и сроки текущего контроля успеваемости и промежуточной аттестации | | | | | | |
| Код модуля в составе дисциплины, практики и т.п. | Формы текущего контроля успеваемости | | Виды промежуточной аттестации | | Виды итоговой аттестации  (только для программ итоговой аттестации и дополнительных образовательных программ) | |
| Формы | Сроки | Виды | Сроки | Виды | Сроки |
| ОСНОВНАЯ ТРАЕКТОРИЯ | | | | | | |
| Форма обучения: очная | | | | | | |
| Семестр 6 |  |  | зачёт, устно, традиционная форма | по графику промежуточной аттестации |  |  |

**2.2. Структура и содержание учебных занятий**

Период обучения (модуль): семестр 6.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. | Введение в проектирование ПО | лекции | 2 |
| по методическим материалам | 4 |
| II. | Объектно-ориентированное проектирование | лекции | 2 |
| практические занятия | 2 |
| по методическим материалам | 4 |
| III. | Моделирование, язык UML | лекции | 1 |
| практические занятия | 3 |
| по методическим материалам | 10 |
| IV. | Архитектурные стили | лекции | 2 |
| по методическим материалам | 4 |
| V. | Предметно-ориентированное проектирование | лекции | 2 |
| практические занятия | 2 |
| по методическим материалам | 8 |
| VI. | Шаблоны проектирования | лекции | 3 |
| практические занятия | 3 |
| по методическим материалам | 25 |
| VII. | Проектирование распределённых приложений | лекции | 3 |
| практические занятия | 5 |
| по методическим материалам | 12 |
| VIII. | Промежуточная аттестация | зачёт | 1 |
| самостоятельная работа | 10 |

**2.2.1. Содержание учебных занятий**

Раздел 1: Введение в проектирование ПО.

1. Программа и программное обеспечение. Понятие архитектуры. Роль проектирования и архитектора в производстве ПО. Понятие архитектурного стиля, архитектурного вида, деградации архитектуры. Design Document, стандарт IEEE 1016-2009. Архитектура и жизненный цикл ПО.

Раздел 2: Объектно-ориентированное проектирование.

1. Сложность, существенная и случайная. Восходящее и нисходящее проектирование. Модульность, сопряжение и связность. Понятие объекта, абстракция, инкапсуляция, наследование, композиция, мутабельность. Методы выделения объектов. Принципы SOLID. Закон Деметры. Абстрактные типы данных, основные принципы абстракции, некоторые принципы написания качественного объектно-ориентированного кода.
2. Практика по проектированию: проектирование интерпретатора командной строки.

Раздел 3: Моделирование, язык UML.

1. Модели в различных областях человеческой деятельности. Роль моделирования в процессе разработки. Виды моделей. Язык UML и его назначение, предпосылки появления. Краткий обзор диаграмм UML.
2. Практика по UML: по данному техническому заданию нарисовать диаграмму случаев использования разрабатываемого приложения и диаграмму активностей для бизнес-процесса предприятия, для которого разрабатывается приложение.
3. Практика по UML: по данному техническому заданию построить модель данных разрабатываемого приложения в виде диаграммы классов, диаграмму компонентов и диаграмму развёртывания.

Раздел 4: Архитектурные стили.

1. Понятие архитектурного шаблона и архитектурного стиля. Примеры архитектурных шаблонов. Основные архитектурные стили: «Главная программа и подпрограммы», объектно-ориентированный стиль, слоистый стиль, «Клиент-сервер», «Пакетная обработка», «Каналы и фильтры», «Blackboard», стили с неявным вызовом: «Издатель-подписчик», событийно-ориентированные стили, «Peer-to-peer».

Раздел 5: Предметно-ориентированное проектирование.

1. Понятие предметно-ориентированного проектирования. Модель предметной области, единый язык, элементы модели: сущность, объект-значение, служба, модуль. Жизненный цикл объекта. Шаблоны поддержки жизненного цикла: «Агрегат», «Фабрика», «Репозиторий». Принципы разбиения объектов по модулям. Реализация ограничений предметной области, паттерн «Спецификация».
2. Практика по проектированию: провести анализ и построить модель предметной области для компьютерной игры в жанре Roguelike по данному техническому заданию.

Раздел 6: Шаблоны проектирования.

1. Понятие шаблона проектирования. Структурные шаблоны. Рекурсивная композиция, шаблоны «Компоновщик» и «Декоратор». Выделение алгоритма в объект, шаблон «Стратегия».
2. Практическое занятие: уточнить модель компьютерной игры Roguelike, построенной на предыдущем занятии, используя шаблоны «Стратегия» для поддержки различных поведений мобов и «Декоратор» для поддержки временных эффектов, накладываемых на мобов.
3. Порождающие шаблоны: «Абстрактная фабрика», «Фабричный метод». Клонирование объектов, шаблон «Прототип». Поэтапное конструирование объекта, шаблон «Строитель».
4. Практическое занятие: уточнить модель компьютерной игры Roguelike, построенной на предыдущем занятии, используя шаблоны «Строитель» для инициализации карты, «Абстрактная фабрика» для создания мобов и предметов на карте и «Прототип» для поддержки клонирования персонажей и предметов.
5. Поведенческие шаблоны. Шаблон «Цепочка обязанностей». Отложенное выполнение. Шаблон «Команда». Подходы к реализации поведения, основанного на модели конечного автомата, шаблон «Состояние».
6. Практическое занятие: уточнить модель компьютерной игры Roguelike, построенной на предыдущем занятии, используя шаблон «Команда» для поддержки взаимодействия с пользователем. Оформить итоговую архитектуру системы в виде диаграмм компонентов и классов.

Раздел 7: Проектирование распределённых приложений.

1. Понятие распределённой системы. Виды сущностей в распределённой системе, способы организации взаимодействия, варианты размещения, типичные архитектурные стили. Удалённые вызовы: RPC, RMI, примеры: protobuf, gRPC. Понятие веб-сервиса. SOAP. Пример технологии разработки веб-сервисов: Windows Communication Foundation. Очереди сообщений, пример: RabbitMQ. Архитектурный стиль REST. Микросервисы. Архитектурный стиль Peer-to-Peer, примеры архитектур.
2. Практическое занятие: реализовать консольный сетевой чат, используя технологию gRPC для обеспечения сетевого взаимодействия.
3. Практическое занятие: оформить сетевой чат, разработанный на предыдущем занятии, в виде Docker-контейнера.

**Раздел 3. Обеспечение учебных занятий**

**3.1. Методическое обеспечение**

**3.1.1 Методические указания по освоению дисциплины**

Для успешного усвоения дисциплины необходимо посещение занятий и самостоятельная работа с материалами лекций и рекомендованной литературой. Особое внимание следует уделить части курса, посвящённой шаблонам проектирования – на лекциях будут разобраны только те шаблоны, которые потребуются на практиках, остальные шаблоны, включённые в список вопросов к зачёту, оставлены для самостоятельного изучения.

Практические занятия рекомендуется посещать все, поскольку задания на них связаны и зависят друг от друга. Практические занятия проводятся в группах по 2-3 человека, при этом допустимо, чтобы состав групп менялся от занятия к занятию.

**3.1.2 Методическое обеспечение самостоятельной работы**

Основная и дополнительная литература.

**3.1.3 Методика проведения текущего контроля успеваемости и промежуточной аттестации и критерии оценивания**

***3.1.3.1. Методика проведения текущего контроля успеваемости и промежуточной аттестации***

Для получения зачёта необходимо сдать теоретическую часть курса в виде устного теоретического зачёта, а также сдать задачи с практических занятий. В случае пропуска практического занятия задачу, дававшуюся на нём, разрешается досдать в течение семестра или на зачёте. На решение задач в течение зачёта отводится два академических часа, разрешается пользоваться литературой.

Теоретическая часть зачёта проводится в устной форме. Обучающемуся задаётся два вопроса, на которые он отвечает без выделения времени на подготовку. Преподаватель вправе задавать уточняющие вопросы, а также, в случае, если после ответа на два вопроса есть сомнения в оценке, вопросы по всему содержанию курса.

По желанию преподавателя на экзамен допустимо приглашать других преподавателей с квалификацией не ниже изложенной в п. 3.2.1.

Допускается проведение устного зачёта с использованием ИКТ.

***3.1.3.2. Критерии оценивания итогового процента освоения дисциплины***

Задачи с практических занятий оцениваются по шкалам и критериям, индивидуальным для каждой задачи. Критерии и максимальные баллы приведены в разделе 3.1.4 и доводятся до обучающихся вместе с условием задачи. Результирующая оценка находится в диапазоне от 0 до 100 и вычисляется по формуле MAX(0, (n/N – 0.6)) \* 2.5 \* 100, где n – суммарный балл, набранный обучающимся на момент аттестации, N – максимально возможный суммарный балл за курс. Например, обучающийся, успешно сдавший 60% заданий, получает 0 баллов, 80% – 50, 100% – 100 баллов.

На теоретическом зачёте ответ на каждый вопрос оценивается по шкале от 0 (нет ответа) до 10 (очень хороший ответ), далее оценка усредняется. Результат переводится в диапазон от 0 до 100.

Итоговый процент выполнения целей изучения дисциплины вычисляется как минимум из оценки за практические занятия и оценки за теоретический зачёт. Далее применяется следующее правило выставления оценки:

|  |  |  |
| --- | --- | --- |
| Итоговый процент  выполнения, % | Оценка СПбГУ при  проведении зачёта | Оценка  ECTS |
| 90-100 | зачтено | A |
| 80-89 | зачтено | B |
| 70-79 | зачтено | C |
| 61-69 | зачтено | D |
| 50-60 | зачтено | E |
| менее 50 | не зачтено | F |

**3.1.4 Методические материалы для проведения текущего контроля успеваемости и промежуточной аттестации (контрольно-измерительные материалы, оценочные средства)**

***3.1.4.1. Формируемые дисциплиной компетенции***

**Компетенции, впервые формируемые дисциплиной:**

Нет

**Компетенции, развиваемые дисциплиной:**

* **ПКП-1** — способен проектировать программные системы.
* **УК-1** — способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач.
* **УК-2** — способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений
* **УКБ-1** — способен участвовать в разработке и реализации проектов, в т.ч. предпринимательских.
* **УКБ-3** — способен понимать сущность и значение информации в развитии общества, использовать основные методы получения и работы с информацией с учетом современных технологий цифровой экономики и информационной безопасности.

**Компетенции, полностью сформированные по результатам освоения дисциплины:**

Нет.

Для каждой компетенции применяется линейная шкала оценивания, определяемая долей успешно выполненных заданий, проверяющих данную компетенцию.

***3.1.4.2. Контрольно-измерительные материалы (примеры)***

***Пример списка вопросов для устного зачёта:***

1. Понятие архитектуры, профессия «Архитектор».
2. Архитектурные виды.
3. Роль архитектуры в жизненном цикле программного обеспечения.
4. Понятие декомпозиции. Модульность, связность, сопряжение, сложность.
5. Понятия класса и объекта, абстракция, инкапсуляция, наследование.
6. Принципы выделения абстракций предметной области.
7. Принципы SOLID, закон Деметры.
8. Моделирование, визуальные модели, виды моделей.
9. Язык UML. Проектирование структуры системы, диаграммы классов.
10. Диаграммы объектов, диаграммы пакетов UML.
11. Диаграммы компонентов, диаграммы развёртывания UML.
12. Диаграмма случаев использования UML.
13. Диаграмма активностей UML.
14. Диаграммы конечных автоматов UML.
15. Диаграммы последовательностей UML.
16. Диаграммы коммуникации UML.
17. Диаграммы составных структур, коопераций, временные диаграммы.
18. Диаграммы обзора взаимодействия, диаграммы потоков данных.
19. Диаграммы «Сущность-связь».
20. Понятие архитектурного стиля, трёхзвенная архитектура,
21. Model-View-Controller, Sense-Compute-Control.
22. Слоистый стиль, «Клиент-сервер».
23. Пакетная обработка, каналы и фильтры
24. Стиль Blackboard.
25. Стили с неявным вызовом, Publish-Subscribe.
26. Стиль Peer-to-peer.
27. Понятие Domain-Driven Design, единый язык.
28. Изоляция предметной области в DDD, антипаттерн «Умный GUI».
29. DDD, основные структурные элементы модели предметной области.
30. DDD, паттерн «Агрегат».
31. DDD, паттерны «Фабрика», «Репозиторий».
32. Моделирование ограничений, паттерн «Спецификация».
33. Паттерн «Компоновщик».
34. Паттерн «Декоратор».
35. Паттерн «Стратегия».
36. Паттерн «Адаптер».
37. Паттерн «Заместитель».
38. Паттерн «Фасад».
39. Паттерн «Мост».
40. Паттерн «Приспособленец».
41. Паттерн «Фабричный метод».
42. Паттерн «Шаблонный метод».
43. Паттерн «Абстрактная фабрика».
44. Паттерн «Одиночка».
45. Паттерн «Прототип».
46. Паттерн «Строитель».
47. Паттерн «Посредник».
48. Паттерн «Команда».
49. Паттерн «Цепочка ответственности».
50. Паттерн «Наблюдатель».
51. Паттерн «Состояние».
52. Паттерн «Посетитель».
53. Паттерн «Хранитель».
54. Понятие распределённой системы, заблуждения при проектировании распределённых систем.
55. RPC, RMI. Пример: gRPC.
56. Веб-сервисы, SOAP. WCF.
57. Веб-сервисы, REST. ASP.NET Web APIs.
58. Архитектурные стили распределённых приложений: Big Compute, Big Data.
59. Web-queue-worker, N-звенная архитектура.
60. Микросервисная архитектура.
61. Дизайн REST-интерфейса.
62. Принципы дизайна распределённых приложений: самовосстановление.
63. Паттерн «Circuit Breaker».
64. Принципы дизайна распределённых приложений: избыточность.
65. Принципы дизайна распределённых приложений: минимизация координации.
66. Command and Query Responsibility Segregation.
67. CAP-теорема, модели ACID и BASE.
68. Принципы дизайна распределённых приложений: проектирование для обслуживания.

***Проверяемые компетенции*:** УК-1, УКБ-3

***Критерии оценивания*:** обучающемуся даётся два билета и задаётся несколько дополнительных вопросов по курсу. Ответ на каждый вопрос билета и на дополнительные вопросы оценивается по шкале от 0 (нет ответа) до 10 (очень хороший ответ), далее оценка усредняется. Результат переводится в диапазон от 0 до 100.

***Примеры задач для практик:***

**Практика 1: проектирование CLI**

В командах по два-три человека спроектировать простой интерпретатор командной строки, поддерживающий команды:

* cat [FILE] — вывести на экран содержимое файла
* echo — вывести на экран свой аргумент (или аргументы)
* wc [FILE] — вывести количество строк, слов и байт в файле
* pwd — распечатать текущую директорию
* exit — выйти из интерпретатора

При этом:

* должны поддерживаться одинарные и двойные кавычки (full and weak quoting);
* окружение (команды вида “имя=значение”), оператор $;
* вызов внешней программы через Process (или его аналоги);
  + если введено что-то, чего интерпретатор не знает;
* пайплайны (оператор «|»).

Задача выполняется в командах по 2-3 человека в течение пары, результатом является диаграмма классов, описывающая структуру системы, и рассказ о том, как оно должно работать. В конце пары одно из решений защищают у доски.

***Проверяемые компетенции*:** ПКП-1, УК-2, УКБ-1

***Критерии оценивания*:** решение оценивается по шкале от 0 (нет решения) до 10 (продемонстрировано знание формальных нотаций, умение выполнить грамотную декомпозицию предметной области, разработана и грамотно описана качественная архитектура).

**Практики 2 и 3: автомобильный завод**

Крупному автомобильному заводу требуется информационная система для учёта дефектов сборки, возникающих при сборке автомобиля на конвейере.

Работа завода организована следующим образом:

* Завод получает набор заказов от дилеров на производство определённого количества автомобилей в определённой комплектации и определённого цвета
* Отдел планирования производства составляет производственный план, определяющий с учётом загруженности конвейера и поставок комплектующих, в каком порядке автомобили будут запущены в производство
* Процесс производства начинается со сборки кузова, после сборки кузова попадают в покраску, где находятся относительно долгое время (порядка нескольких дней), после покраски происходит сборка всех остальных комплектующих автомобиля на конвейерной линии
* После сборки автомобиль попадает в отдел контроля качества, где проходит серию стендовых испытаний и ходовые испытания
* В случае успешного прохождения проверок автомобиль отгружается заказчику

На каждом из этапов производства и тестирования могут быть выявлены дефекты сборки (от погнутой детали кузова или царапин до сбоев в работе программы управления инжектором). В случае обнаружения дефекта автомобиль снимается с конвейера и отправляется в ремонтную зону, где дефект устраняется и автомобиль возвращается на конвейер либо в зону тестирования. Каждый участок конвейера имеет несколько (от одной до четырёх) ремонтных зон, в каждой зоне имеется несколько (от одного до шести) ремонтных мест, где может находиться один автомобиль. Каждая ремонтная зона обслуживается ремонтной бригадой, имеющей бригадира и нескольких ремонтников. Бригады работают в ремонтных зонах посменно.

Целью разработки информационной системы учёта дефектов является повышение качества сборки, минимизация числа дефектов и минимизация времени, проводимого автомобилем в ремонте. Для достижения этой цели разрабатываемая система должна обладать следующей функциональностью:

* Возможностью вносить данные о ремонтных зонах, ремонтных местах, ремонтных бригадах и их бригадирах
* Возможностью зарегистрировать дефект после поступления автомобиля в ремонтную зону и выполнения первоначальной диагностики:
* Место дефекта, отмечаемое на схеме автомобиля
* Возможная причина дефекта
* Номер автомобиля, дата поступления
* Ремонтник, выполнявший диагностику и ремонт
* Возможностью зафиксировать время начала и конца ремонта, рабочего, выполнявшего ремонт и ремонтное место, занимаемое автомобилем в процессе ремонта
* Возможностью для бригадира назначить ремонт конкретному рабочему
* Возможностью для рабочего зарегистрироваться в системе как доступного для получения заданий в начале смены
* Диспетчер должен иметь возможность видеть свободные и занятые места в ремонтных зонах своего участка конвейера, а также наличие свободных рабочих в зоне, и возможность направить автомобиль для ремонта в свободную ремонтную зону
* Отдел контроля качества должен получать в конце каждой смены сводный отчёт по выполненным ремонтам по каждому участку конвейера, включая общее количество ремонтов, статистику по местам дефекта, среднее время ремонта, детальную информацию о каждом ремонте
* Начальник цеха должен получать в конце каждой смены отчёт об общем количестве ремонтов и суммарном времени ремонтов
* Бригадир каждой бригады в конце каждой смены должен получать информацию по каждому ремонтнику бригады о количестве выполненных ремонтов и времени, затраченном на каждый ремонт, а также суммарному времени, которое работник занимался ремонтами.
* Бухгалтерия в конце каждого месяца должна получать отчёт по каждому ремонтнику всех ремонтных бригад о количестве отработанных смен.

Вашей задачей будет в командах по два-три человека проанализировать данное предложение и построить на языке UML следующие модели:

1. диаграмму случаев использования разрабатываемого приложения;
2. диаграмму активностей для всего бизнес-процесса предприятия, для которого разрабатывается приложение;
3. модель данных разрабатываемого приложения в виде диаграммы классов;
4. диаграмму компонентов разрабатываемого приложения;
5. диаграмму развёртывания разрабатываемого приложения.

***Проверяемые компетенции*:** УК-1, УК-2, УКБ-1

***Критерии оценивания*:** каждаяиз пяти задач оценивается по шкале от 0 (нет решения) до 4 (продемонстрировано знание UML, предложенное решение адекватно как часть описания архитектуры системы).

**Практика 4: Roguelike**

Roguelike – это довольно популярный жанр компьютерных игр, названный в честь игры Rogue, 1980 года выхода. Характеризуется:

* простой тайловой или консольной графикой;
* активным использованием случайной генерации;
* перманентной смертью персонажа и невозможностью загрузить предыдущее сохранение после смерти персонажа;
* чрезвычайно развитым набором игровых правил (чем они и интересны с точки зрения архитектуры);
* высокой свободой действий персонажа (так называемые «игры-песочницы»).

Примеры:

* https://en.wikipedia.org/wiki/NetHack
* https://en.wikipedia.org/wiki/Angband\_(video\_game)
* https://en.wikipedia.org/wiki/Ancient\_Domains\_of\_Mystery

Вашей задачей будет в командах по два-три человека провести анализ и построить модель предметной области согласно принципам предметно-ориентированного проектирования для такой компьютерной игры.

При этом должны быть выполнены следующие функциональные требования:

* персонаж игрока, способный перемещаться по карте, управляемый с клавиатуры;
* карта обычно генерируется, но для некоторых уровней грузится из файла;
* характеристики — здоровье, сила атаки и т.д.;
* экспа и уровни персонажа, с ростом уровня повышаются характеристики;
* инвентарь персонажа, включающий элементы, влияющие на его характеристики, которые можно надеть и снять;
* несколько разных видов мобов, способных перемещаться по карте;
* боевая система — движущиеся объекты, пытающиеся занять одну клетку карты, атакуют друг друга;
* консольная графика, традиционная для этого жанра игр.

***Проверяемые компетенции*:** ПКП-1, УК-1, УК-2, УКБ-1

***Критерии оценивания*:** задача оценивается по шкале от 0 (нет решения) до 10 (продемонстрировано знание UML, умение выполнить грамотную декомпозицию предметной области, разработана и грамотно описана качественная архитектура).

**Практика 5: Roguelike, структурные шаблоны**

Модифицировать модель игры, построенную на предыдущей практике, чтобы:

* использовать в Roguelike паттерн "Стратегия" для поддержки различных поведений мобов;
* используя паттерн "Декоратор", реализовать для игрока возможность конфузить мобов. Моб, находящийся под эффектом конфузии, перемещается, случайно выбирая соседнюю клетку. Эффект должен быть временным.

Результатом работы должна быть диаграмма классов, описывающая применение данных шаблонов в архитектуре игры.

***Проверяемые компетенции*:** ПКП-1, УК-2, УКБ-1

***Критерии оценивания*:** задача оценивается по шкале от 0 (нет решения) до 10 (продемонстрировано знание UML, умение выполнить грамотную декомпозицию предметной области, разработана и грамотно описана качественная архитектура).

**Практика 6: Roguelike, порождающие шаблоны**

Модифицировать модель игры, построенную на предыдущей практике, чтобы:

* использовать шаблон «Строитель» для инициализации карты;
* использовать шаблон «Абстрактная фабрика» для создания мобов и предметов на карте;
* использовать шаблон «Прототип» для поддержки клонирования персонажей и предметов (например, как результат применения персонажем заклинания либо как свойство некоторых мобов – для игр подобного жанра типично «размножение» слабых противников на карте, например, грибов или мышей, при этом клонированный объект получает характеристики своего «родителя»).

Результатом работы должна быть диаграмма классов, описывающая применение данных шаблонов в архитектуре игры.

***Проверяемые компетенции*:** ПКП-1, УК-2, УКБ-1

***Критерии оценивания*:** задача оценивается по шкале от 0 (нет решения) до 10 (продемонстрировано знание UML, умение выполнить грамотную декомпозицию предметной области, разработана и грамотно описана качественная архитектура).

**Практика 7: Roguelike, поведенческие шаблоны**

Модифицировать модель игры, построенную на предыдущей практике, чтобы использовать в Roguelike шаблон "Команда" для поддержки взаимодействия с пользователем. Результатом работы должна быть диаграмма классов, описывающая применение этого шаблона в архитектуре игры.

***Проверяемые компетенции*:** ПКП-1, УК-2, УКБ-1

***Критерии оценивания*:** задача оценивается по шкале от 0 (нет решения) до 10 (продемонстрировано знание UML, умение выполнить грамотную декомпозицию предметной области, разработана и грамотно описана качественная архитектура).

**Практика 8: сетевой чат на gRPC**

В командах по два-три человека спроектировать и реализовать сетевой чат (наподобие Telegram) с помощью gRPC:

* peer-to-peer, то есть соединение напрямую;
* графический пользовательский интерфейс;
* отображение имени отправителя, даты и текста сообщения;
* при запуске указываются:
  + адрес peer-а и порт, если хотим подключиться;
  + должно быть можно не указывать, тогда работаем в режиме сервера;
  + наше имя пользователя.

Реализация допустима на любом языке программирования из поддержанных gRPC.

***Проверяемые компетенции*:** УК-1, УК-2, УКБ-1

***Критерии оценивания*:** задача оценивается по шкале от 0 (нет решения) до 10 (продемонстрировано работоспособное решение с качественным исходным кодом и грамотное архитектурное описание).

**Практика 9: Docker**

Оформить сетевой чат, разработанный на предыдущем занятии, в виде Docker-контейнера.

***Проверяемые компетенции*:** УК-1, УК-2, УКБ-1

***Критерии оценивания*:** задача оценивается по шкале от 0 (нет решения) до 5 (продемонстрировано работоспособное решение).

***3.1.4.3. Соответствие индикаторов достижения компетенций и контрольно-измерительных материалов***

|  |  |  |
| --- | --- | --- |
| № | Код индикатора и индикатор достижения универсальной компетенции | Контрольно-измерительные материалы (КИМ) (тестовые вопросы, контрольные задания, кейсы и пр.) |
|  | 1 | 2 |
| 1 | ПКП-1.03669.1. Создаёт диаграмму случаев использования UML, диаграмму активностей UML по данному списку пользовательских требований. | Практика 2 |
| 2 | ПКП-2.003669.1. Создаёт архитектуру системы по данному заданию с использованием диаграмм UML. | Практики 1, 4-7 |
| 3 | ПКП-3.003669.1. Создаёт компонентную модель системы по заданным функциональным требованиям. | Практики 1, 4 |
| 4 | УК 1.1. Анализирует задачу, выделяя ее базовые составляющие. | Практики 1-4, 8 |
| 5 | УК 1.2. Определяет информацию, необходимую для решения поставленной задачи. | Практики 1-4, 8 |
| 6 | УК 1.3. Осуществляет по различным запросам поиск информации, необходимой для решения поставленной задачи. | Список вопросов для устного экзамена, практики 1-9 |
| 7 | УК 1.4. Оценивает достоинства, недостатки и последствия вариантов решения поставленных задач. | Практики 1-4, 8 |
| 8 | УК 1.5. Грамотно, логично, аргументированно формирует собственные суждения, решения и оценки. | Список вопросов для устного экзамена |
| 9 | УК-2.1. Определяет круг задач в рамках поставленной цели. | Практики 1-4, 8 |
| 10 | УК-2.2. Предлагает способы решения поставленных задач. | Практики 1-4, 8 |
| 11 | УК-2.3. Оценивает соответствие способов решения цели проекта. | Практики 1, 4, 8 |
| 12 | УК-2.4. Планирует реализацию задач в зоне своей ответственности с учетом имеющихся ресурсов и ограничений, действующих правовых норм. | Практики 1-4, 8 |
| 13 | УК-2.5. Выполняет задачи в зоне своей ответственности в соответствии с запланированными результатами и точками контроля. | Практики 1-4, 8 |
| 14 | УКБ-1.1. Определяет круг задач в рамках поставленной цели. | Практики 1-4, 8 |
| 15 | УКБ-1.2. Предлагает способы решения поставленных задач. | Практики 1-4, 8 |
| 16 | УКБ-1.3. Оценивает соответствие способов решения цели проекта. | Практики 1, 4, 8 |
| 17 | УКБ-1.4. Планирует реализацию задач в зоне своей ответственности с учетом имеющихся ресурсов и ограничений, действующих правовых норм. | Практики 1-4, 8 |
| 18 | УКБ-1.5. Выполняет задачи в зоне своей ответственности в соответствии с запланированными результатами и точками контроля. | Практики 1-4, 8 |
| 19 | УКБ-1.6. Представляет результаты проекта. | Практики 1-9 |
| 20 | УКБ-1.7. Предлагает возможности использования результатов проекта и/или совершенствования. | Практики 1, 4, 8 |
| 21 | УКБ-3.1. Находит и использует различные источники информации. | Список вопросов для устного экзамена, практики 1-9 |
| 22 | УКБ-3.2. Точно определяет тип и форму необходимой информации. | Список вопросов для устного экзамена, практики 1-9 |
| 23 | УКБ-3.3. Получает информацию и сохраняет ее в удобном для работы формате. | Список вопросов для устного экзамена, практики 1-9 |
| 24 | УКБ-3.4. Проверяет достоверность собранной информации. | Список вопросов для устного экзамена, практики 1-9 |

**3.1.5 Методические материалы для оценки обучающимися содержания и качества учебного процесса**

Для оценки обучающимися содержания и качества учебного процесса применяется анкетирование в соответствии с методикой и графиком, утвержденными в установленном порядке.

**3.2. Кадровое обеспечение**

**3.2.1 Образование и (или) квалификация штатных преподавателей и иных лиц, допущенных к проведению учебных занятий**

К проведению лекционных занятий должны привлекаться преподаватели, имеющие диплом о высшем образовании по соответствующему направлению.

**3.2.2 Обеспечение учебно-вспомогательным и (или) иным персоналом**

Учебно-вспомогательный и инженерно-технический персонал должен иметь соответствующее образование и обладать навыками организации работы с пользовательскими программными продуктами в локальной сети компьютерного класса и в Интернете.

**3.3. Материально-техническое обеспечение**

**3.3.1 Характеристики аудиторий (помещений, мест) для проведения занятий**

Учебные аудитории для проведения учебных занятий, оснащенные стандартным оборудованием, используемым для обучения в СПбГУ в соответствии с требованиями материально-технического обеспечения.

**3.3.2 Характеристики аудиторного оборудования, в том числе неспециализированного компьютерного оборудования и программного обеспечения общего пользования**

Стандартное оборудование, используемое для обучения в СПбГУ.

MS Windows, MS Office, Mozilla FireFox, Google Chrome, Acrobat Reader DC, WinZip, Антивирус Касперского.

**3.3.3 Характеристики специализированного оборудования**

Не требуется.

**3.3.4 Характеристики специализированного программного обеспечения**

Не требуется.

**3.3.5 Перечень и объёмы требуемых расходных материалов**

Маркеры для доски, губка

**3.4. Информационное обеспечение**

**3.4.1 Список обязательной литературы**

1. Pooley, R. J. Wilcox, P., Applying UML : Advanced Applications, Oxford : Butterworth-Heinemann. 2004, 202 pp., <https://find.library.spbu.ru/vufind/Record/ocn173660459> (дата обращения: 24.06.2019г)
2. Bjøner, Dines. Software Engineering 3: Domains, Requirements, and Software Design, Berlin, Heidelberg : Springer Berlin Heidelberg, 2006, 768pp., URL: https://find.library.spbu.ru/vufind/Record/978-3- 540-33653- 2 (дата обращения: 15.01.2017г)

**3.4.2 Список дополнительной литературы**

1. Bernus, Peter. Mertins, Kai. Schmidt, Günter. Handbook on Architectures of Information Systems, Berlin, Heidelberg : Springer Berlin Heidelberg, 2006, 896pp., URL: https://find.library.spbu.ru/vufind/Record/978-3- 540-26661- 7 (дата обращения: 15.01.2017г)

**3.4.3 Перечень иных информационных источников**

* Сайт Научной библиотеки им . М. Горького СПбГУ: <http://www.library.spbu.ru/>
* Электронный каталог Научной библиотеки им. М. Горького СПбГУ:

<http://www.library.spbu.ru/cgi-bin/irbis64r/cgiirbis_64.exe?C21COM=F&I21DBN=IBIS&P21DBN=IBIS>

* Перечень электронных ресурсов, находящихся в доступе СПбГУ:

<http://cufts.library.spbu.ru/CRDB/SPBGU/>

* Перечень ЭБС, на платформах которых представлены российские учебники, находящиеся в доступе СПбГУ:

<http://cufts.library.spbu.ru/CRDB/SPBGU/browse?name=rures&resource_type=8>

**Раздел 4. Разработчики программы**

Литвинов Юрий Викторович, кандидат технических наук, доцент кафедры системного программирования, [y.litvinov@spbu.ru](mailto:y.litvinov@spbu.ru)

По материалам Брыксина Тимофея Александровича, кандидата технических наук