Автономная некоммерческая организация образовательная организация дополнительного профессионального образования «Учебный Центр Люксофт»
(АНО ДПО «Учебный центр Люксофт»)

Направление: Проектирование и системный анализ в области разработки программного обеспечения и информационных систем

Утверждаю
Директор АНО ДПО «УЦ Люксофт»
Иванова Е. В.

(подпись)
«25» февраля 2019 г.

УЧЕБНАЯ ПРОГРАММА

Дисциплина **«Проектирование и системный анализ в области разработки программного обеспечения и информационных систем»**

ДЛЯ ОБУЧЕНИЯ В РАМКАХ ДОПОЛНИТЕЛЬНОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ (ПОВЫШЕНИЕ КВАЛИФИКАЦИИ)

Москва 2019 г.

1. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ, ЕЕ МЕСТО В УЧЕБНОМ ПРОЦЕССЕ

1.1 Цель преподавания дисциплины является

Сформировать у слушателей набор знаний и навыков для проектирования и системного анализа архитектур программного обеспечения.

1.2 Задачи изучения дисциплины

Задачами изучения дисциплины являются:

- Анализ архитектур программных систем
- Изучение лучших практик проектирования на основе апробированных во множестве проектов типовых решениях
- Применение шаблонов DDD для построения реализуемой модели домена
- Знакомство с техниками построения концептуальной модели приложения

1.3 Связь дисциплины с другими учебными дисциплинами

Изучение дисциплины предполагает наличие у слушателей специальных знаний и умений использовать UML, владение основами объектно-ориентированного проектирования. Желательно знать язык Java, основы архитектуры компьютеров, архитектуры сетей передачи данных и GOF паттернов.

2. СОДЕРЖАНИЕ УЧЕБНОЙ ПРОГРАММЫ

Категория слушателей программы: архитекторы, проектировщики, а также аналитики, руководители проектов, руководители разработки, ведущие разработчики.

К освоению учебной программы допускаются: лица, имеющие среднее профессиональное и (или) высшее образование.

Базовые компетенции, которыми должен владеть слушатель программы:

АРХИТЕКТОР

Должен знать: Владеть методами и инструментами анализа и проектирования, владеть интегрированными средами разработки, инструментарием управления проектом, системами контроля версий.
Проводить оценку осуществимости требований, вырабатывать требования к программному обеспечению, применять специализированные методологии для построения архитектуры программных систем и методы и технологии использования средств разработки для получения кода с заданной функциональностью и степенью качества.
Моделировать системы на базе готовой архитектуры. Разрабатывать архитектурный дизайн программного обеспечения и тестовые сценарии по спецификациям требований. Использовать методы и технологии верификации формальных спецификаций. Оценивать соответствие программного кода архитектуре компьютерной системы и качество построенной архитектуры системы (адекватность, полнота, непротиворечивость). Проектировать программное обеспечение с использование специализированных программных пакетов. Управлять персоналом.
Осуществлять объектно-ориентированное проектирование, взаимодействовать с представителями заказчика или специалистами в предметной области.

Требования к квалификации. Высшее профессиональное (техническое или инженерно-экономическое) образование и стаж работы по специальности не менее 2 лет.

ПРОЕКТИРОВЩИК

Должен знать: методы разработки, анализа и проектирования программного обеспечения, технологические и технико-эксплуатационные характеристики архитектур развертывания компонентов, методы по изготовлению, отладке и сдаче в эксплуатацию систем и средств автоматизации и управления, решать прикладные задачи с использованием современных информационнокоммуникационных технологий, моделирование и проектирование структуры данных и знаний, прикладные и информационные процессы.

Требования к квалификации. Высшее профессиональное (техническое или инженерно-экономическое) образование и стаж работы по специальности не менее 3 лет.

АНАЛИТИК

Должен знать: использовать терминологию, понятийный аппарат, базовые идеи, методы и процессы предметной области заказчика, работать с различными видами исходных данных о предметной области, разрабатывать документы в соответствии с требованиями государственных, отраслевых и корпоративных стандартов, работать с различными видами исходных данных о состоянии рынка информационных систем, владеть инструментарием обработки данных на персональном компьютере, анализировать требования заказчика по использованию информационных систем, применять формализованные языки и нотации для построения моделей процессов, данных, объектов, читать модели, описанные с помощью специализированных формализованных языков и нотаций, оформлять технические задания в соответствии с требованиями государственных, отраслевых и корпоративных стандартов.

Требования к квалификации. Высшее профессиональное (техническое или инженерно-экономическое) образование и стаж работы по специальности не менее 3 лет.

РУКОВОДИТЕЛЬ ПРОЕКТОВ

Должен знать: принципы обосновывания принимаемых проектных решений, осуществлять постановку и выполнять эксперименты по проверке их корректности и эффективности, готовить презентации, научно-технические отчеты по результатам выполненной работы, оформлять результаты исследований в виде статей и докладов на научно-технических конференциях, способы ставления и решения прикладные задачи с использованием современных нформационнокоммуникационных технологий, способен принимать участие в создании и управлении ИС на всех этапах жизненного цикла, способен применять системный подход и математические методы в формализации решения прикладных задач, понимание классических концепций и моделей менеджмента в управлении проектами, методовы управления процессами разработки требований, оценки рисков, приобретения, проектирования, конструирования, тестирования, эволюции и сопровождения, основы групповой динамики, психологии и профессионального поведения, специфичных для программной инженерии, методы контроля проекта и умение осуществлять контроль версий.

Требования к квалификации. Высшее профессиональное (техническое или инженерно-экономическое) образование и стаж работы по специальности не менее 3 лет.

РУКОВОДИТЕЛЬ РАЗРАБОТКИ

Должен знать: способы ставить и решать прикладные задачи с использованием современных информационно-коммуникационных технологий, способы принимать участие в создании и управлении ИС на всех этапах жизненного цикла, способы проводить оценку экономических затрат на проекты по информатизации и автоматизации решения прикладных задач, оценивать и выбирать современные операционные среды и информационно-коммуникационные технологии для информатизации и автоматизации решения прикладных задач и создания, способы применения методов анализа прикладной области на концептуальном, логическом, математическом и алгоритмическом уровнях, способен применять системный подход и математические методы в формализации решения прикладных задач.

Требования к квалификации. Высшее профессиональное (техническое или инженерно-экономическое) образование и стаж работы по специальности не менее 3 лет.

ВЕДУЩИЙ РАЗРАБОТЧИК

Должен знать: ставить и решать прикладные задачи с использованием современных информационно-коммуникационных технологий, документировать процессы создания информационных систем на всех стадиях жизненного цикла, эксплуатировать и сопровождать информационные системы и сервисы, принимать участие во внедрении, адаптации и настройке прикладных, принимать участие в реализации профессиональных коммуникаций в рамках проектных групп, презентовать результаты проектов и обучать пользователей ИС, проводить оценку экономических затрат на проекты по информатизации и автоматизации, оценивать и выбирать современные операционные среды и информационно-коммуникационные технологии для информатизации и автоматизации, анализировать и выбирать методы и средства обеспечения информационной безопасности, анализировать рынок программно-технических средств, информационных продуктов и услуг для решения прикладных задач и создания информационных систем.

Требования к квалификации. Высшее профессиональное (техническое или инженерно-экономическое) образование и стаж работы по специальности не менее 3 лет.

Форма обучения: Очная и очно-заочная. Занятия проводятся в группах.

Срок обучения: дифференцированный от 16 академических часов до 296 академических часов, в зависимости от количества выбранных модулей.
Учебная программа состоит из 18 модулей.

3. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

В ходе изучения программы слушатель должен приобрести знания и навыки, необходимые для проектирования и системного анализа архитектур программного обеспечения, для работы в проектах с различной предметной областью и технологической базой.

После завершения обучения Слушатели смогут:

1. проектировать системы с использованием производительности приемов;
2. оптимизировать системы с повышенными требованиями к производительности;
3. проектировать интерфейс распределённого доступа к объектам
4. работать с требованиями и определять иерархию требований;
5. использовать атрибуты требований и связи между требованиями для оценки трудоемкости их реализации или изменения.

4. МОДУЛЬНЫЙ ПЛАН ПРОГРАММЫ

№	Наименование модулей	Объем в академ. часах		
		лекции	семинары	Всего часов
1	2	3	4	5
1	ARC-001 Основные практики архитектора ПО	10	14	24
2	ARC-003 Domain Driven Design	6	10	16
3	ARC-004 Шаблоны проектирования приложений масштаба предприятия	10	14	24
4	ARC-005 Аналитические шаблоны	10	14	24
5	ARC-008 Проектирование высокопроизводительных приложений	12	18	30
6	ARC-010 Введение в системную архитектуру ПО	2	2	4
7	ARC-013 Интеграция в корпоративных системах	10	14	24
8	REQ-001 Основы визуального моделирования с использованием UML 2.0	6	10	16
9	REQ-002 Мастерская по разработке и управлению требованиями. UML и Модель сценариев использования (Use Case Model)	6	10	16
10	REQ-003 Объектно-ориентированный анализ ИС. Концептуальное моделирование на UML для системных аналитиков	8	12	20
11	REQ-023 Принципы создания пользовательских и нтерфейсов	8	12	20
12	REQ-028 Подготовка технических писателей	6	10	16
13	REQ-047_PRG Школа проектирования графических интерфейсов	5	10	15
14	REQ-029 Техника написания сценариев использования	6	10	16
15	REQ-041 Исследование пользователей	2	2	4
16	REQ-042 Юзабилити-анализ и проектирование веб-интерфейсов	3	4	7
17	REQ-043 BABOK: Управление требованиями и коммуникации в IT-проекте	6	10	16
18	REQ-044 Бюджетное юзабилити	2	2	4
ИТОГО		**118**	**178**	**296**

5. УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН МОДУЛЕЙ ПРОГРАММЫ

Модуль 1. ARC-001 Основные практики архитектора ПО

№	Тема	Количество часов	Форма контроля
1	Контекст, основные понятия, обор дисциплины проектирования и анализа архитектур программных систем, роль архитектора	3	Выборочный опрос слушателей на лекционном и семинарском занятиях
2	Инициация проекта: определение заинтересованных сторон и бизнес целей	3	Выборочный опрос слушателей на лекционном и семинарском занятиях
3	Идентификация, формализация и приоритизация требований к качественным характеристикам системы	4	Выборочный опрос слушателей на лекционном и семинарском занятиях
4	Проектирование архитектуры	4	Проверка конспектов лекций слушателей
5	Документирование архитектуры	4	Проверка конспектов лекций слушателей
6	Архитектурный анализ	4	Подготовка анализа
7	Итоговая аттестация по модулю	2	Устный опрос слушателей
	Итого	24	

Модуль 2. ARC-003 Domain Driven Design

№	Тема	Количество часов	Форма контроля
1	Основные подходы в реализации бизнес-логики	2	Выборочный опрос слушателей на лекционном и семинарском занятиях
2	Моделирование предметной области	3	-
3	Шаблоны DDD	3	Проверка конспектов лекций слушателей
4	Рефакторинг модели. Дистилляция	3	Проверка конспектов лекций слушателей
5	DDD и архитектура корпоративных приложений	3	Проверка конспектов лекций слушателей
6	Итоговая аттестация по модулю	2	Устный опрос слушателей
	Итого	16	

Модуль 3. ARC-004 Шаблоны проектирования приложений масштаба предприятия

№	Тема	Количество часов	Форма контроля
1	Введение	1	-
2	Шаблоны предметной логики	2	Выборочный опрос слушателей на лекционном и семинарском занятиях
3	Архитектурные шаблоны доступа к данным	3	Проверка конспектов лекций слушателей
4	Веб-представление	4	Проверка конспектов лекций слушателей
5	Сложные шаблоны	4	Проверка конспектов лекций слушателей
6	Базовые шаблоны	4	Выборочный опрос слушателей на лекционном и семинарском занятиях
7	Заключение	4	Выборочный опрос слушателей на лекционном и семинарском занятиях
8	Итоговая аттестация по модулю	2	Устный опрос слушателей
	Итого	24	

Модуль 4. ARC-005 Аналитические шаблоны

№	Тема	Количество часов	Форма контроля
1	Объектно-ориентированный анализ	4	Проверка конспектов лекций слушателей
2	UML диаграммы аналитической модели	4	Проверка конспектов лекций слушателей
3	Базовые паттерны	5	Проверка конспектов лекций слушателей
4	Решение типовых задач	5	Проверка конспектов лекций слушателей
5	Техники перехода от концептуальной модели к дизайну	4	Выборочный опрос слушателей на лекционном и семинарском занятиях
6	Итоговая аттестация по модулю	2	Устный опрос слушателей
	Итого	24	

Модуль 5. ARC-008 Проектирование высокопроизводительных приложений

№	Тема	Количество часов	Форма контроля
1	Понятие высокопроизводительной системы	3	Выборочный опрос слушателей на лекционном и семинарском занятиях
2	Основные характеристики производительности систем	3	Выборочный опрос слушателей на лекционном и семинарском занятиях
3	Анализ требований для высокопроизводительных систем	3	Лабораторная работа
4	Проектирование высокопроизводительных систем	3	Лабораторная работа
5	Шаблоны для реализации высокопроизводительных систем	3	Лабораторная работа
6	Кодирование высокопроизводительных	4	Проверка конспектов лекций слушателей

	систем		
7	Тестирование высокопроизводительных систем	3	Проверка конспектов лекций слушателей
8	Оптимизация производительности для приложений	3	-
9	Методология SPE	3	-
10	Итоговая аттестация по модулю	2	Устный опрос слушателей
	Итого	30	

Модуль 6. ARC-010 Введение в системную архитектуру ПО

№	Тема	Количество часов	Форма контроля
1	Введение в системную архитектуру Задачи, решаемые разработкой архитектуры ПО Предприятие и системная архитектура ПО	1	Выборочный опрос слушателей на лекционном и семинарском занятиях
2	Аппаратная часть и архитектура ПО Программное обеспечение и системная архитектура Теоретические основы архитектуры ПО	1	Выборочный опрос слушателей на лекционном и семинарском занятиях
3	Системная архитектура ПО и практическая разработка ПО	1	Лабораторная работа
4	Итоговая аттестация по модулю	1	Устный опрос слушателей
	Итого	4	

Модуль 7. ARC-013 Интеграция в корпоративных системах

№	Тема	Количество часов	Форма контроля
1	Обзор Enterprise Integration: определение, задачи, обзор основных направлений и технологий (EAI, EII, ETL, SOA	2	Выборочный опрос слушателей на лекционном и семинарском занятиях
2	Интеграционные требования, способы сбора и описания, функциональные требования, нефункциональные требования	2	Выборочный опрос слушателей на лекционном и семинарском занятиях
3	Обзор инструментов и средств интеграции (основные функции; основные производители): Messaging, ESB, ETL, SOA appliances, Integration frameworks	2	Лабораторная работа
4	Интеграция на уровне данных: подходы и их ограничения (общая база данных, хранилище данных, витрины данных, федеративные базы данных, распределенные файловые системы, Event sourcing, архитектура Lambda)	2	Лабораторная работа
5	Системы обмена сообщениями и их интерфейсы (JMS, AMQP, IBM MQ, Apache MQ, Apache Kafka)	2	Лабораторная работа
6	Промышленные стандарты интеграции (Обзор OASIS, W3C, WS-I, SOAP, REST);	2	Проверка конспектов лекций слушателей
7	Шаблоны интеграции приложений (Enterprise Integration Patterns) и их реализация в Apache Camel, Mule ESB, IBM Integration Bus;	2	Проверка конспектов лекций слушателей
8	Подходы и шаблоны при реализации функциональных требований	2	-
9	Подходы при реализации нефункциональных требований	2	-
10	Способы построения сервисов и	4	

	интеграционных решений, лучшие практики		
11	Итоговая аттестация по модулю	2	Устный опрос слушателей
	Итого	24	

Модуль 8. REQ-001 Основы визуального моделирования с использованием UML 2.0

№	Тема	Количество часов	Форма контроля
1	Понятие модели и принципы визуального моделирования.	1	-
2	Основы языка UML.	1	-
3	Концепции объектно-ориентированного подхода.	2	Выборочный опрос слушателей на лекционном и семинарском занятиях
4	UML: Диаграмма классов.	2	Проверка конспектов лекций слушателей
5	UML: Диаграмма вариантов использования.	2	Проверка конспектов лекций слушателей
6	UML: Моделирование поведения.	2	Проверка конспектов лекций слушателей
7	UML: Прочие диаграммы.	2	Проверка конспектов лекций слушателей
8	Процесс построения модели.	2	Проверка конспектов лекций слушателей
9	Итоговая аттестация по модулю	2	Устный опрос слушателей
	Итого	16	

Модуль 9. REQ-002 Мастерская по разработке и управлению требованиями. UML и Модель сценариев использования (Use Case Model)

№	Тема	Количество часов	Форма контроля
1	Определение требований к системе.	1	-
2	Работа с заинтересованными лицами.	1	-
3	Анализ проблемы.	1	Проверка конспектов лекций слушателей
4	Документирование требований.	1	Выборочный опрос слушателей на лекционном и семинарском занятиях
5	Лучшие практики программной инженерии.	2	Проверка конспектов лекций слушателей
6	Иерархия моделей RUP.	2	Выборочный опрос слушателей на лекционном и семинарском занятиях
7	UML: Модель вариантов использования.	2	Проверка конспектов лекций слушателей
8	Структурирование модели прецедентов.	2	-
9	Переход от бизнес-модели к модели прецедентов.	2	Проверка конспектов лекций слушателей
10	Итоговая аттестация по модулю	2	Тестирование
	Итого	16	

Модуль 10. REQ-003 Объектно-ориентированный анализ ИС. Концептуальное моделирование на UML для системных аналитиков

№	Тема	Количество часов	Форма контроля
1	Объектно-ориентированный подход	3	Проверка конспектов лекций слушателей
2	Объектно-ориентированный анализ	3	-
3	Объектно-ориентированноепроектирование	3	Проверка конспектов лекций слушателей
4	Лучшие практики программной инженерии;	3	Выборочный опрос слушателей на лекционном и семинарском занятиях
5	RUP: иерархия моделей	3	Проверка конспектов лекций слушателей
6	RUP: дисциплина «Анализ и проектирование»	3	Выборочный опрос слушателей на лекционном и семинарском занятиях
7	Итоговая аттестация по модулю	2	Тестирование
	Итого	20	

Модуль 11. REQ-028 Подготовка технических писателей

№	Тема	Количество часов	Форма контроля
1	Задачи технического писателя; Требования к документации; Этапы разработки документации. Планирование;	3	Проверка конспектов лекций слушателей
2	Комплектность документации; Виды, типы и структура документов; Разработка разделов документов	3	-
3	Описание функциональных возможностей; Структурно-обусловленные разделы; Оформление, стилевая дисциплина	4	Проверка конспектов лекций слушателей
4	Изложение. Организация текста; Верификация документации; Рекомендации по работе с MS Word;	4	Выборочный опрос слушателей на лекционном и семинарском занятиях
5	Итоговая аттестация по модулю	2	Тестирование
	Итого	16	

Модуль 12. REQ-023 Принципы создания пользовательских интерфейсов

№	Тема	Количество часов	Форма контроля
1	Принципы проектирования интерфейсов	5	Проверка конспектов лекций слушателей
2	проектирование интерфейсов для настольных систем	5	-
3	Разработка требований к графическому оконному интерфейсу (в формате лабораторной работы)	5	Проверка конспектов лекций слушателей
4	Итоговая аттестация по модулю	5	Тестирование
	Итого	20	

Модуль 13. REQ-047_PRG Школа проектирования графических интерфейсов

№	Тема	Количество часов	Форма контроля
1	REQ-041 Исследование пользователей	5	Проверка конспектов лекций слушателей
2	REQ-042 Юзабилити-анализ и проектирование веб-интерфейсов	3	-
3	REQ-044 Бюджетное юзабилити	3	Проверка конспектов лекций слушателей
7	Итоговая аттестация по модулю	3	Тестирование
	Итого	15	

Модуль 12. REQ-029 Техника написания сценариев использования

№	Тема	Количество часов	Форма контроля
1	Основные принципы и понятия управления требованиями	1	-
2	«Экторы» и сценарии использования	1	-
3	Модель сценариев использования и UML	2	Проверка конспектов лекций слушателей
4	Эскиз сценария использования	2	Проверка конспектов лекций слушателей
5	Детализация сценария использования	2	Выборочный опрос слушателей
6	Структуризация модели на UML	2	Выборочный опрос слушателей
7	Инструменты, Подсказки и эвристики	2	Выборочный опрос слушателей
8	Сценарии использования (Use case) и User story в Agile	2	Выборочный опрос слушателей
9	Итоговая аттестация по модулю	2	Устный опрос слушателей
	Итого	16	

Модуль 12. REQ-041 Исследование пользователей

№	Тема	Количество часов	Форма контроля
1	Desk Study	0,5	-
2	Interview	0,5	-
3	Observation	0,5	-
4	Context Enquiry	1	Проверка конспектов лекций слушателей
5	Diary Studies, etc.	1	Выборочный опрос слушателей
6	Итоговая аттестация по модулю	0,5	Устный опрос слушателей
	Итого	4	

Модуль 13. REQ-042 Юзабилити-анализ и проектирование веб-интерфейсов

№	Тема	Количество часов	Форма контроля
1	Что такое User Experience Design и Usability. Их важность для проекта	1	-
2	Оценка пользовательского интерфейса	3	Выборочный опрос слушателей
3	Проектирование дизайна интерфейсов	2	Проверка конспектов лекций слушателей
4	Итоговая аттестация по модулю	1	Устный опрос слушателей
	Итого	7	

Модуль 14. REQ-043 BABOK: Управление требованиями и коммуникации в IT-проекте

№	Тема	Количество часов	Форма контроля
1	BABOK® Guide. Организация и основные термины	1	-
2	Управление рамками решений и их требованиями	2	Проверка конспектов лекций слушателей
3	Управление прослеживаемостью требований	3	Проверка конспектов лекций слушателей
4	Управление требованиями для повторного использования	3	Выборочный опрос слушателей
5	Подготовка набора требований	3	Проверка конспектов лекций слушателей
6	Коммуникации при выявлении требований	3	Выборочный опрос слушателей на лекционном и семинарском занятиях
7	Итоговая аттестация по модулю	1	Устный опрос слушателей
	Итого	16	

Модуль 15. REQ-044 Бюджетное юзабилити

№	Тема	Количество часов	Форма контроля
1	Юзабилити, дизайн, управление проектами.	3,5	Выборочный опрос слушателей на лекционном и семинарском занятиях
2	Итоговая аттестация по модулю	0,5	Устный опрос слушателей
	Итого	4	

6. ФОРМА КОНТРОЛЯ

Контроль усвоения учебной программы проводится в различных формах:

1 Выборочный опрос на лекциях
2. Проверка конспектов лекций слушателей
3. Лабораторная работа
4. Опрос при проведении практических (семинарских) занятий

5. Тестирование

При успешном прохождении итогового контроля обучающемуся выдается сертификат о повышении квалификации установленного образца.

Вопросы для выходного тестирования

1. Система это:
   - ○ множество элементов реального мира, отобранных с определенной целью;
   - ○ множество связей между элементами реального мира;
   - ○ **все вышеперечисленное;**
   - ○ ничего из вышеперечисленного.

2. Модель системы это:
   - ○ UML-диаграмма;
   - ○ **упрощенное описание системы с акцентом на отдельных ее свойствах;**
   - ○ полное описание системы в виде текста или диаграмм;
   - ○ система математических уравнений, описывающих функционирование системы.

3. При моделировании системы следует описывать:
   - ☐ финансовые показатели системы;
   - ☐ **структуру системы;**
   - ☐ только алгоритмы работы программно-аппаратных модулей системы;
   - ☐ **поведение системы.**

4. Отбирая элементы реальной системы для включения их в модель, следует проверять соответствие этих элементов:
   - ☐ **цели моделирования;**
   - ☐ бюджету проекта;
   - ☐ **точке зрения модели;**
   - ☐ **рамкам проекта**.

5. От того, как построена модель, зависит:
   - ○ размер файла, содержащего модель;
   - ○ **понимание проблем, существующих в системе, и путей их решения**;
   - ○ можно ли показывать модель заказчику;
   - ○ сложность реализации программного обеспечения.

6. Правильно построенная модель должна содержать:
   - ○ только верхнеуровневые (абстрактные) элементы;
   - ○ только детальные описания конкретных элементов;
   - ○ **описания с разной степенью детальности, дополняющие друг друга**;
   - ○ только графические элементы без текстовых комментариев.

7. Сложная система может быть описана:
   - ○ **только совокупностью взаимодополняющих моделей**;
   - ○ даже одной моделью – просто уметь надо;
   - ○ графической моделью с подробными текстовыми комментариями;
   - ○ просто графической моделью, т.к. тексты слишком сложны для понимания.

8. Моделирование – это:
   - ○ основная работа аналитика;
   - ○ основная цель этапа анализа при разработке программного обеспечения;
   - ○ основной способ коммуникации с заказчиком;
   - ○ **способ осмысления и анализа проблем реальной системы**.

9. Семантика – это:
   - ○ набор графических обозначений, применяемых в модели;
   - ○ **набор фундаментальных элементов моделирования, обладающих определенным смыслом;**
   - ○ правила выделения элементов из среды и правила их использования в модели;
   - ○ этот термин вообще не имеет отношения к моделированию.

10. Методология – это:
    - ○ набор графических обозначений, применяемых в модели;
    - ○ набор фундаментальных элементов моделирования, обладающих определенным смыслом;
    - ○ **правила выделения элементов из среды и правила их использования в модели;**
    - ○ этот термин вообще не имеет отношения к моделированию.

11. Нотация – это:
    - ○ **набор графических обозначений, применяемых в модели;**
    - ○ набор фундаментальных элементов моделирования, обладающих определенным смыслом;
    - ○ правила выделения элементов из среды и правила их использования в модели;
    - ○ этот термин вообще не имеет отношения к моделированию.

12. UML-диаграммы могут быть использованы на этапе:
    - ☐ **сбора и анализа требований;**
    - ☐ **проектирования и разработки архитектуры системы;**
    - ☐ **тестирования;**
    - ☐ **разворачивания готового ПО на территории заказчика.**

13. UML-диаграммы описывают:
    - ☐ только структуру системы;
    - ☐ только поведение системы;
    - ☐ только архитектуру системы;
    - ☐ **все необходимые аспекты системы.**

14. С помощью UML артефакты программной системы можно:
    - ☐ **визуализировать;**
    - ☐ **специфицировать;**
    - ☐ **конструировать;**
    - ☐ **документировать.**

15. Применение UML принесет наибольшую пользу, если процесс разработки:
    - ☐ **управляется вариантами использования;**
    - ☐ **базируется на архитектуре системы;**
    - ☐ **итеративен и инкрементален;**
    - ☐ не относится к методам экстремального программирования.

16. Согласно стандарту UML расширение семантики языка:
    - ○ запрещено, т.к. отклонение от стандарта сделает модель нечитабельной;
    - ○ **позволено с помощью стереотипов, ограничений и поименованных значений;**
    - ○ позволено, но только с помощью добавления новых графических символов;
    - ○ позволено, но только с помощью текстовых комментариев.

17. Стереотип – это:
    - ○ стандартное (стереотипное) использование определенных ключевых слов;
    - ○ **текстовое или графическое расширение стандартной семантики UML;**
    - ○ шаблонный способ мышления аналитика или другого участника проекта;
    - ○ графический символ, которым моделировщик может расширить нотацию UML.

18. Ограничение – это:
    - ○ признание того, что время проекта и его бюджет не бесконечны;
    - ○ текстовое описание условий, при которых система не будет работоспособной;
    - ○ [логическое условие], которое должно быть выполнено, чтобы связанный с ним переход можно было запустить;
    - ○ **{логическое условие}, ограничивающее семантику выбранного элемента модели;**

19. Сторожевое условие – это:
    - ○ контрольная дата или минимальная сумма остатка бюджета, по достижении которых менеджер проекта фиксирует наступление проектного риска;
    - ○ текстовое описание условий, при которых система не будет работоспособной;
    - ○ **[логическое условие], которое должно быть выполнено, чтобы связанный с ним переход можно было запустить;**
    - ○ {логическое условие}, ограничивающее семантику выбранного элемента модели;

20. Поименованное значение – это:
    - ○ значение атрибута класса;
    - ○ часть описания реального объекта, содержащая метку (tag) и присвоенное ей значение;
    - ○ **часть описания элемента модели, содержащая метку (tag) и присвоенное ей значение;**
    - ○ произвольный фрагмент модели, на который можно ссылаться, используя присвоенное ему имя;

21. Объектно-ориентированный подход:
    - ○ **основан на естественной склонности человеческого сознания к выделению объектов реального мира и их классификации;**
    - ○ является математической абстракцией, удобной для высокоуровневого описания сложных систем;
    - ○ используется разработчиками, применяющими объектно-ориентированные языки программирования;
    - ○ очередной модный термин, скрывающий давно известные истины.

22. Абстракция – это:
   - ○ философская концепция, упрощающая понимание сложных описаний;
   - ○ разбиение сложной модели на части, более простые для анализа и/или реализации;
   - ○ сокрытие внутренней структуры и логики работы объекта с целью защиты от внешнего вмешательства или неправильного использования;
   - ○ **выделение существенных характеристик реального объекта, отличающих его от других объектов.**

23. Инкапсуляция – это:
   - ○ философская концепция, упрощающая понимание сложных описаний;
   - ○ разбиение сложной модели на части, более простые для анализа и/или реализации;
   - ○ **сокрытие внутренней структуры и логики работы объекта с целью защиты от внешнего вмешательства или неправильного использования;**
   - ○ выделение существенных характеристик реального объекта, отличающих его от других объектов.

24. Модульность – это:
   - ○ философская концепция, упрощающая понимание сложных описаний;
   - ○ **разбиение сложной модели на более простые части, удобные для анализа и/или реализации;**
   - ○ сокрытие внутренней структуры и логики работы объекта с целью защиты от внешнего вмешательства или неправильного использования;
   - ○ выделение существенных характеристик реального объекта, отличающих его от других объектов.

25. Объект – это:
   - ○ абстракция любой реальной сущности из выбранной предметной области;
   - ○ **абстракция реальной или воображаемой сущности с четко выраженными границами, индивидуальностью, состоянием и поведением;**
   - ○ философская концепция, помогающая описывать сложные системы реального мира;
   - ○ элемент реализации программного обеспечения.

26. В качестве объекта может выступать:
   - ☐ **предмет;**
   - ☐ **устройство;**
   - ☐ **событие;**
   - ☐ **понятие.**

27. Класс – это:
   - ○ элемент исходного кода программного обеспечения;
   - ○ тип значения переменной или константы;
   - ○ степень точности, с которой модель отражает реальную систему;
   - ○ **абстракция совокупности объектов, имеющих сходные атрибуты, поведение и отношения с другими объектами.**

28. Класс и Объект:
   - ☐ являются синонимами;
   - ☐ **объект является экземпляром класса;**
   - ☐ класс является коллекцией однотипных объектов;
   - ☐ **класс является описанием шаблона, на основании которого создаются однотипные объекты.**

29. Наследование – это:
   - ○ правила оформления моделей, обеспечивающее совместимость разных версий для каждой диаграммы;
   - ○ **возможность описать новый класс на основе уже существующего;**
   - ○ процесс создания экземпляров класса;
   - ○ возможность в различных классах иметь различную реализацию для функций, имеющих одинаковые названия.

30. Полиморфизм – это:
   - ○ правила оформления моделей, поддерживающие преемственность разных версий для каждой диаграммы;
   - ○ возможность описать новый класс на основе уже существующего;
   - ○ процесс создания экземпляров класса;
   - ○ **возможность в различных классах иметь различную реализацию для функций, имеющих одинаковые названия.**

31. Диаграмма классов предназначена для построения моделей:
   - ○ только концептуального уровня;
   - ○ только уровня спецификации требований;
   - ○ только уровня проектирования и реализации;
   - ○ **все вышеперечисленное.**

32. Атрибут класса – это:
   - ○ **описание свойства, общего для всех объектов данного класса;**
   - ○ характеристика элемента модели, определяющая его важность для системы;
   - ○ услуга, которую класс может выполнить в рамках реализации своего поведения;

○ поименованное значение для элемента модели.

33. Метод класса – это:
○ описание свойства, общего для всех объектов данного класса;
○ характеристика элемента модели, определяющая его важность для системы;
○ **услуга, которую класс может выполнить в рамках реализации своего поведения;**
○ поименованное значение для элемента модели.

34. Видимость (атрибута или метода) – это:
○ права доступа разных членов проектной команды к разным элементам модели;
○ актуальность версии диаграммы классов – невидимые диаграммы не включаются в модель;
○ режим отображения связей между классами, определяющий видны ли пользователю связи, проходящие позади изображений классов;
○ **признак того, имеют ли другие классы возможность обращаться к значению атрибута или вызывать метод;**

35. Символ «–» перед именем атрибута означает, что атрибут:
○ Открытый (public), т.е. виден для любого другого класса;
○ защищенный (protected), т.е. виден только для потомков данного класса;
○ **закрытый (private), т.е. не виден для внешних классов и может использоваться только классом, его содержащим;**
○ пакетный (package), т.е. виден для любого класса из данного пакета.

36. Символ «+» перед именем атрибута означает, что атрибут:
○ **открытый (public), т.е. виден для любого другого класса;**
○ защищенный (protected), т.е. виден только для потомков данного класса;
○ закрытый (private), т.е. не виден для внешних классов и может использоваться только классом, его содержащим;
○ пакетный (package), т.е. виден для любого класса из данного пакета.

37. Символ «~» перед именем атрибута означает, что атрибут:
○ открытый (public), т.е. виден для любого другого класса;
○ защищенный (protected), т.е. виден только для потомков данного класса;
○ закрытый (private), т.е. не виден для внешних классов и может использоваться только классом, его содержащим;
○ **пакетный (package), т.е. виден для любого класса из данного пакета.**

38. Символ «#» перед именем атрибута означает, что атрибут:
○ открытый (public), т.е. атрибут виден для любого другого класса;
○ **защищенный (protected), т.е. виден только для потомков данного класса;**
○ закрытый (private), т.е. не виден для внешних классов и может использоваться только классом, его содержащим;
○ пакетный (package), т.е. виден для любого класса из данного пакета.

39. Ассоциация – это:
○ сходство спецификаций двух классов (подобные наборы атрибутов и методов);
○ **отношение между классами, которое описывает связи между экземплярами этих классов;**
○ сходство перечня атрибутов двух классов;
○ сходство поведения двух классов;

40. Полюс ассоциации содержит информацию для экземпляра класса, который:
○ **соприкасается с противоположным полюсом ассоциации;**
○ соприкасается с данным полюсом ассоциации;
○ расположен выше или левее по ходу ассоциации;
○ расположен ниже или правее по ходу ассоциации.

41. Ассоциация и Атрибут:
○ дополняют друг друга;
○ ассоциация для класса является более важной характеристикой, чем атрибут;
○ **атрибут – это вырожденная ассоциация, на другом конце которой неявно находится класс, отсутствующий в модели;**
○ и то, и другое является необязательным при описании класса.

42. Ассоциативный класс:
○ класс, имеющий не менее трех ассоциаций с другими классами;
○ **ассоциация, имеющая собственные свойства;**
○ класс, подобный связанному с ним классу (по составу атрибутов и/или методов);
○ класс, реализующий ассоциативный поиск в хранилище данных.

43. Агрегация – это отношение вида «часть-целое», при котором:
- ☐ часть не может существовать вне целого;
- ☐ **целое является контейнером, содержащим части;**
- ☐ **часть может входить в состав нескольких других составных объектов;**
- ☐ время жизни части не может превышать времени жизни целого.

44. Композиция – это отношение вида «часть-целое», при котором:
- ☐ **часть не может существовать вне целого;**
- ☐ **целое является контейнером, содержащим части;**
- ☐ часть может входить в состав нескольких других составных объектов;
- ☐ **время жизни части не может превышать времени жизни целого.**

45. Наконечник на линии, обозначающей связи Агрегации и Композиции, ставится:
- ○ со стороны части;
- ○ **со стороны целого;**
- ○ с обеих сторон;
- ○ не ставится – целое обозначается специальным знаком на изображении класса.

46. Связь, обозначающая наследование, направлена:
- ○ **в сторону родителя;**
- ○ в сторону наследника;
- ○ направление не обозначается;
- ○ родитель указывается с помощью специального обозначения на изображении класса.

47. Диаграмма вариантов использования:
- ○ содержит полный перечень требований к системе;
- ○ описывает сценарии использования, выполняемые действующими лицами с применением системы;
- ○ **определяет систему в виде «черного ящика», предоставляющего интерфейс для внешних по отношению к системе действующих лиц;**
- ○ описывает входы и выходы системы, а также – правила, по которым входная информация преобразуется в выходную.

48. Вариант использования (use case) – это:
- ○ должностные обязанности пользователя, работающего с системой;
- ○ **описание того, как действующее лицо пытается достичь некой цели, используя систему;**
- ○ вид лицензии, по которой используется система: передача в собственность, сдача в аренду, предоставление услуги;
- ○ описанное в контракте пожелание заказчика.

49. В виде Актора (actor) на диаграмме вариантов использования могут быть изображены:
- ☐ организация-заказчик;
- ☐ сама система;
- ☐ **пользователи системы;**
- ☐ **другие (внешние) системы.**

50. Диаграмма последовательности описывает поведение системы в виде:
- ○ потоков сообщений между объектами;
- ○ **последовательности передачи сообщений между объектами;**
- ○ упорядоченного выполнения отдельных действий, соединённых между собой потоками управления;
- ○ логики перехода конечного автомата из одного состояния в другое под воздействием внешних событий (сообщений).

51. Диаграмма состояний описывает поведение системы в виде:
- ○ потоков сообщений между объектами;
- ○ последовательности передачи сообщений между объектами;
- ○ упорядоченного выполнения отдельных действий, соединённых между собой потоками управления;
- ○ **логики перехода конечного автомата из одного состояния в другое под воздействием внешних событий (сообщений).**

52. Диаграмма коммуникации описывает поведение системы в виде:
- ○ **потоков сообщений между объектами;**
- ○ последовательности передачи сообщений между объектами;
- ○ упорядоченного выполнения отдельных действий, соединённых между собой потоками управления;
- ○ логики перехода конечного автомата из одного состояния в другое под воздействием внешних событий (сообщений).

53. Диаграмма деятельности описывает поведение системы в виде:
- ○ потоков сообщений между объектами;
- ○ последовательности передачи сообщений между объектами;
- ○ **упорядоченного выполнения отдельных действий, соединённых между собой потоками управления;**

○ логики перехода конечного автомата из одного состояния в другое под воздействием внешних событий (сообщений).

54. На диаграмме последовательности используются следующие виды сообщений:
- ☐ **асинхронное (передатчик не ждет реакции от получателя сообщения и продолжает свою деятельность);**
- ☐ **синхронное (вызов метода или передача потока управления);**
- ☐ **возврат из вызова метода (завершение синхронного сообщения);**
- ☐ материальный поток (поток документов, денежных средств и др.).

55. На диаграмме последовательности линия жизни обозначает:
- ○ путь, по которому передаются сообщения между объектами;
- ○ **промежуток времени, в течение которого данный объект существует;**
- ○ вертикальную область, соответствующую одному из исполнителей в моделируемом процессе;
- ○ горизонтальную шкалу времени, на которой отмечаются моменты наступления событий.

56. На диаграмме последовательности фокус управления обозначает:
- ○ видимость и доступность данного объекта пользователю;
- ○ центральный элемент модели, управляющий ходом выполнения процесса;
- ○ **период времени, в течение которого объект выполняет некоторое действие, находясь в активном состоянии;**
- ○ класс, в котором сконцентрирована логика работы данной диаграммы.

57. Компонент – это:
- ○ файл с исходным кодом и связанная с ним часть модели системы;
- ○ **физически существующая часть системы, которая обеспечивает реализацию части поведения системы;**
- ○ аппаратная часть программно-аппаратного комплекса системы;
- ○ написанная другим разработчиком часть программы.

58. Декомпозиция – это:
- ○ то же самое, что и декомпиляция;
- ○ способ обратного инжиниринга (получение моделей из исходных кодов);
- ○ **разбиение модели на уровни таким образом, что структура элемента более высокого уровня раскрывается на диаграмме более низкого уровня;**
- ○ способ многоуровневой нумерации диаграмм модели в зависимости от степени абстрактности диаграммы.

59. Компоненты должны:
- ☐ содержать не более 6 классов;
- ☐ **инкапсулировать свое содержимое;**
- ☐ иметь понятные для заказчика названия;
- ☐ **иметь четко определенный интерфейс;**

60. На диаграмме деятельности количество потоков управления, которые выходят из элемента условного разветвления, может быть:
- ○ **любым (равным числу условий для данного шага диаграммы);**
- ○ только два (true/false);
- ○ не более трех (по числу свободных сторон элемента);
- ○ не более 7±2 (чтобы диаграмма не потеряла читабельность).

1. Задача «Архитектурный анализ». Какая роль ее выполняет?
2. Задача «Анализ сценариев использование». Какая роль ее выполняет?
3. Задача «Идентификация механизмов дизайна». Какая роль ее выполняет?
4. Задача «Проектирование сценариев использование». Какая роль ее выполняет?
5. Задача «Проектирование подсистем». Какая роль ее выполняет?
6. Задача «Проектирование классов». Какая роль ее выполняет?
7. Задача «Архитектурный анализ». Какая цель этой задачи?
8. В какую задачу входит «Определение ключевых абстракций ИС»?
9. Как представляются в модели ключевые абстракции ИС»?
10. В какую задачу входит «Определение механизмов анализа»?
11. В какую задачу входит «Определение механизмов реализации»?
12. В какую задачу входит «Определение классов реализации сценария использования»?
13. В какую задачу входит «Определение подсистем и их интерфейсов»?
14. Задача «Анализ сценариев использования». Какая цель этой задачи?
15. Задача «Анализ сценариев использования». Какие виды классов определяются?
16. Какие операции (функции) назначаются на классы <<Boundary>>?
17. Какие операции (функции) назначаются на классы <<Control>>
18. Какие операции (функции) назначаются на классы <<Entity>>
19. В какой задаче назначаются «ответственности» (responsibility) на классы?
20. В какой задаче не используются диаграммы последовательности(Sequence)?
21. В какой задаче используются “диаграммы машины состояний”(State Machine)?
22. В какой задаче используются диаграммы деятельности (activity)?

23. В какой задаче используются компонентная диаграммы деятельности (Component)?
24. Чем отношение между классами «Агрегация» отличается от отношения «Композиция»?
25. Чем отношение между классами «Ассоциация» отличается от отношения «Композиция»?
26. Чем отношение между классами «Ассоциация» отличается от отношения «Агрегация»?
27. Чем отношение между классами «Обобщение» отличается от отношения «Агрегация»?
28. Чем отношение между классами «Зависимость» отличается от отношения «Композиция»?
29. Чем отношение между классами «Зависимость» отличается от отношения «Агрегация»?
30. Ромбик на конце линии используется при изображении такого отношения как:
31. Ромбик на конце линии используется при изображении такого отношения как:
32. Стрелка может отсутствовать на конце линии при изображении такого отношения как:
33. Стрелка конце линии отношения «Зависимость» указывает на:
34. Стрелка конце линии отношения «Реализация» указывает на:
35. Стрелка конце линии отношения «Обобщение» указывает на:
36. Стрелка конце линии отношения «Ассоциация» указывает на:
37. Стрелка на конце линии отношения «Агрегация» означает что это:
38. Стрелка на конце линии отношения «Композиция» означает что это:
39. Наименования на конце линии отношения «Композиция» это:
40. Наименования на конце линии отношения «Реализация» это:
41. Наименования на конце линии отношения «Обобщение» это:
42. Наименования на конце линии отношения «Зависимость» это:
43. Наименования на конце линии отношения «Развертывание» это:
44. Наименования на конце линии отношения «Включение» это:
45. Символ «*» на конце линии отношения «Включение» это:
46. Символ «*» на конце линии отношения «Ассоциация» это:
47. Символ «*» на конце линии отношения «Развертывание» это:
48. Символ «*» на конце линии отношения «Зависимость» это:
49. Символ «*» на конце линии отношения «Обобщение» это:
50. Символ «*» на конце линии отношения «Реализация» это:
51. Символ «*» на конце линии отношения «Агрегация» это:
52. Символ «*» на конце линии отношения «Композиция» это:
53. Символ «&» на конце линии отношения «Композиция» это:
54. Символ «+» на конце линии отношения «Композиция» это:
55. Символ «&» на конце линии отношения «Ассоциация» это:
56. Символ «+» на конце линии отношения «Ассоциация» это:
57. Символы «2..5» на конце линии отношения «Ассоциация» это:
58. Символ «2..5» на конце линии отношения «Агрегация» это:
59. Символ «2..5» на конце линии отношения «Композиция» это:
60. Символ «2..5» на конце линии отношения «Зависимость» это:
61. Символ «2..5» на конце линии отношения «Обобщение» это:
62. Символ «2..5» на конце линии отношения «Реализация» это:
63. Что означает символ «Н» в кружке на элементе Диаграммы машины состояний?
64. Что означает символ «Н» в кружке на элементе Диаграммы Активности?
65. Что означает символ «Н» в кружке на элементе Диаграммы Классов?
66. Модель системы это:
68. Сложная система может быть описана:
69. Нотация – это:
70. Согласно стандарту UML расширение семантики языка:
71. Стереотип – это:
72. Ограничение – это:
73. Сторожевое условие – это:
75. Абстракция – это:
76. Инкапсуляция – это:
77.Объект – это:
78. Класс – это:
79. Обобщение – это:
80. Полиморфизм – это:
81. Атрибут класса – это:
82. Операция класса – это:
83. Видимость (атрибута или метода) – это:
84. Символ «–» перед именем атрибута означает, что атрибут:
85. Ассоциация – это:
86. Ассоциативный класс:
87. Ромбик на линии, обозначающей связи Агрегации и Композиции, ставится:
88. Диаграмма последовательности описывает поведение системы в виде:
89. Диаграмма состояний описывает поведение системы в виде:
90. Диаграмма коммуникации описывает поведение системы в виде:

7. ТЕХНИЧЕСКИЕ СРЕДСТВА, ПРИМЕНЯЕМЫЕ ДЛЯ ОБУЧЕНИЯ И КОНТРОЛЯ

Наименование	Форма использования	Ауд.	Кол-во
Персональный компьютер на рабочем месте слушателя	Выполнение практических и лабораторных заданий	1.09	12
Персональный компьютер на рабочем месте преподавателя	Демонстрация рабочего материала	1.09	1
Проектор	Презентация/демонстрация рабочего материала группе	1.09	1

8. УЧЕБНО-МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО ПРОГРАММЕ

1. Буч Г., Рамбо Д., Джекобсон А. Язык UML Руководство пользователя. – ДМК Пресс, 2006
2. Соммервилл Иан. Инженерия программного обеспечения = Software Engineering. – 6-е изд. – М.: «Вильямс», 2002
3. Крэг Ларман. Применение UML и шаблонов проектирования. 3-е издание. М.: Издательский дом «Вильямс», 2013
4. Леффингуэлл Д., Уидриг Д. Принципы работы с требованиями к программному обеспечению. Унифицированный подход.: М.: Вильямс, 2002.
5. Крачтен Ф. Введение в Rational Unified Process. М.: Вильямс, 2002.
6. Соммервилл И. Инженерия программного обеспечения. 6-е издание. Пер. с англ.: - М.: Вильямс, 2002.
7. «BABOK® Guide» - http://www.iiba.org/babok-guide/babok-guide-online.aspx
8. Леффингуэлл Д., Уидриг Д. Принципы работы с требованиями к программному обеспечению. Унифицированный подход.:М.: Вильямс, 2002.
9. Вигерс К. Разработка требований к программному обеспечению. / /Пер, с англ. — М.: Издат.-торговый дом «Русская Редакция», 2004. —576с

Учебная программа разработана: *Юнусов Максим Николаевич*