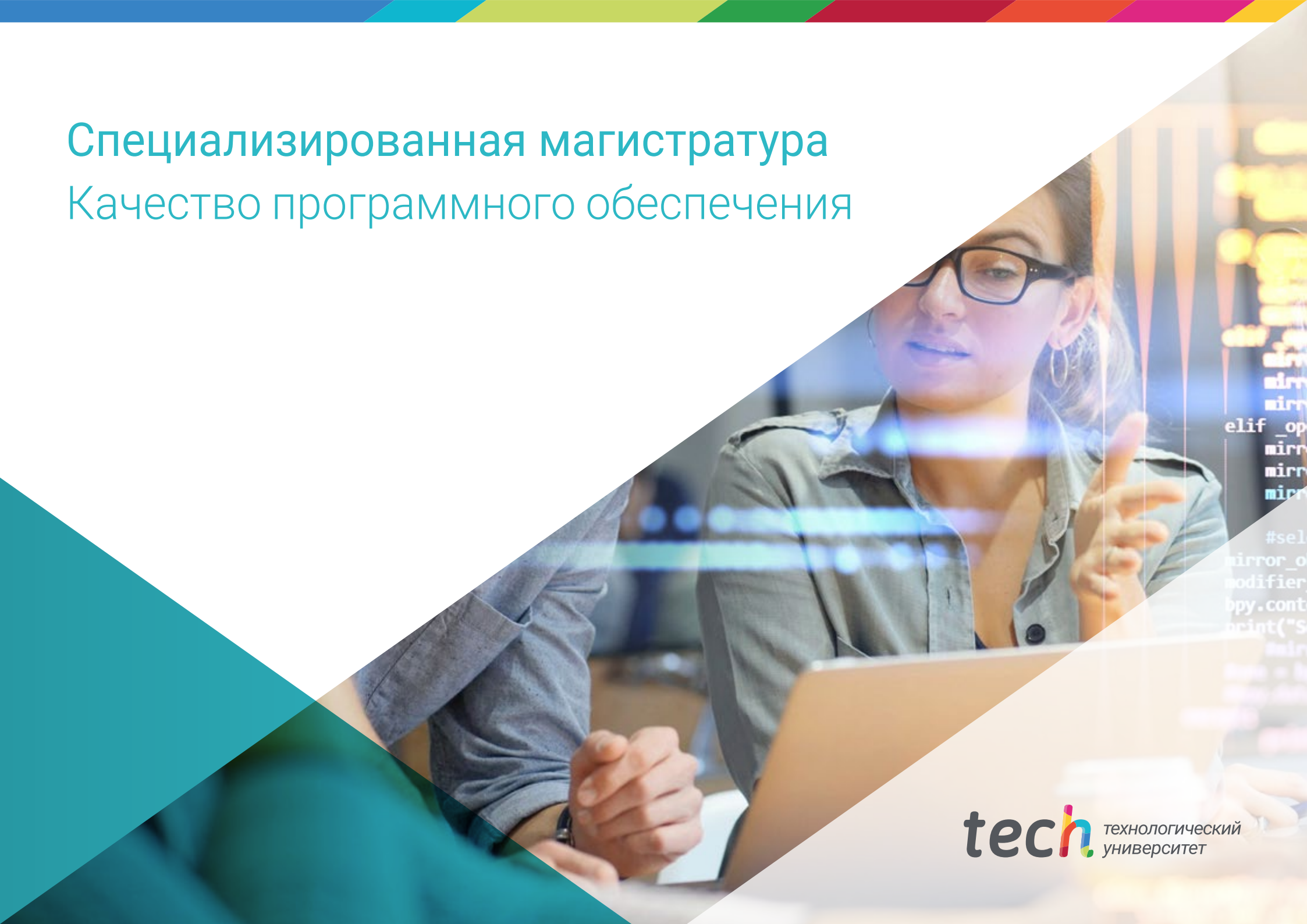
Специализированная магистратура
Качество программного обеспечения



tech технологический университет

tech технологический университет

Специализированная магистратура
Качество программного обеспечения

- Формат: **онлайн**
- Продолжительность: **12 месяцев**
- Учебное заведение: **TECH Технологический университет**
- Режим обучения: **16ч./неделя**
- Расписание: **по своему усмотрению**
- Экзамены: **онлайн**

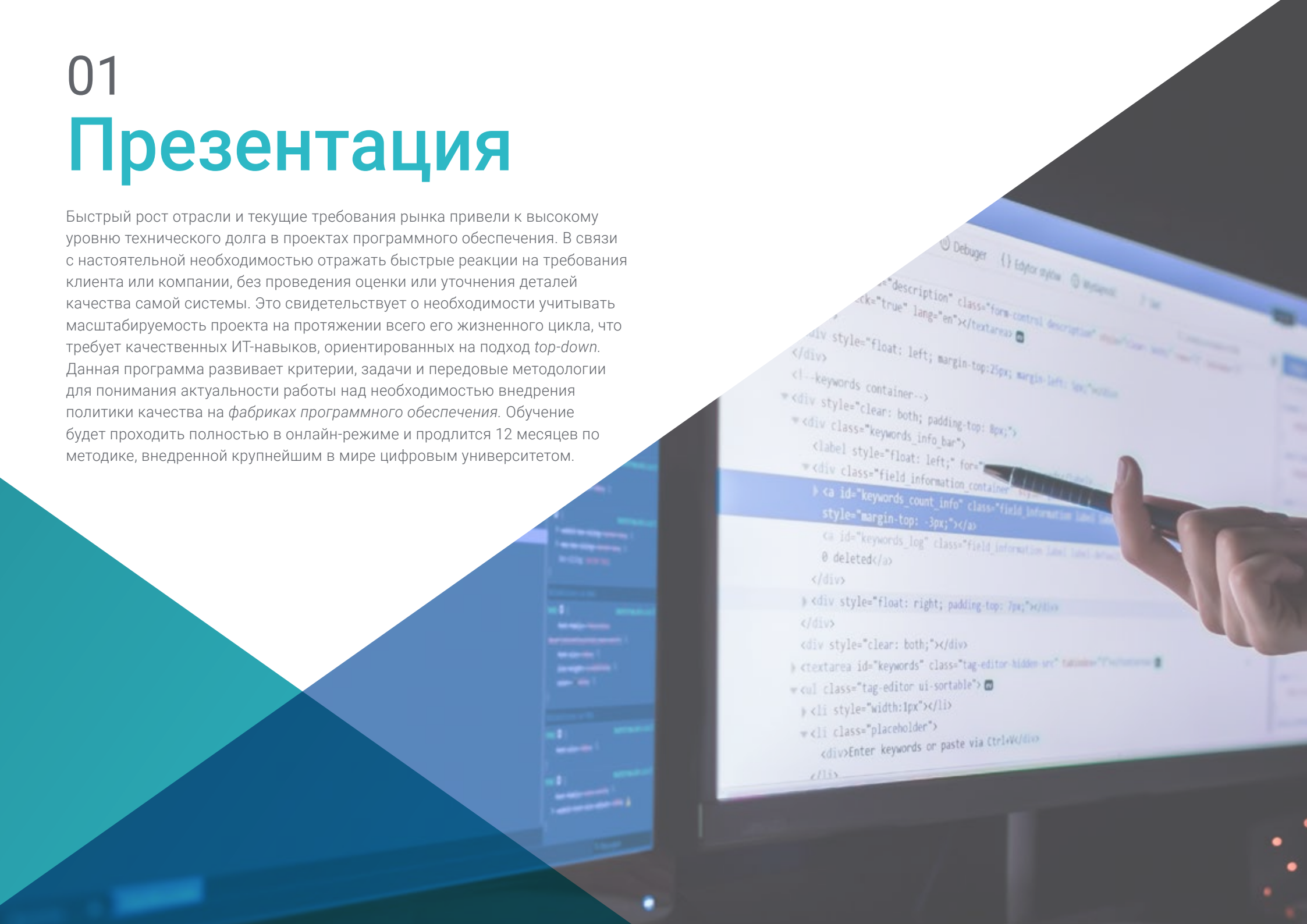
Веб-доступ: **www.techtitute.com/ru/information-technology/professional-master-degree/master-software-quality**

Оглавление

01

Презентация

Быстрый рост отрасли и текущие требования рынка привели к высокому уровню технического долга в проектах программного обеспечения. В связи с настоятельной необходимостью отражать быстрые реакции на требования клиента или компании, без проведения оценки или уточнения деталей качества самой системы. Это свидетельствует о необходимости учитывать масштабируемость проекта на протяжении всего его жизненного цикла, что требует качественных ИТ-навыков, ориентированных на подход *top-down*. Данная программа развивает критерии, задачи и передовые методологии для понимания актуальности работы над необходимостью внедрения политики качества на *фабриках программного обеспечения*. Обучение будет проходить полностью в онлайн-режиме и продлится 12 месяцев по методике, внедренной крупнейшим в мире цифровым университетом.



Специализируйтесь в области качества программного обеспечения с технической и управленческой точек зрения; пройдите специализацию за 12 месяцев и измените свою профессиональную среду к лучшему"

Концепция технического долга, применяемая в настоящее время большим количеством корпораций и администраций по отношению к своим поставщикам, отражает импровизированный способ разработки проектов. Генерирование новых неявных затрат, связанных с необходимостью переделывать проект путем принятия быстрого и простого решения в противовес тому, что должно быть масштабируемым подходом в процессе развития проекта.

Уже несколько лет проекты разрабатываются очень быстро, с целью их заключения с клиентом на основе цены и сроков; вместо применения качественного подхода. Такие решения теперь отнимают силы у многих поставщиков и клиентов.

Данная Специализированная магистратура позволит ИТ-специалистам анализировать критерии, лежащие в основе качества программного обеспечения на всех уровнях. Такие критерии, как стандартизация баз данных, развязка между компонентами информационной системы, масштабируемые архитектуры, метрики, документация, как функциональная, так и техническая. Помимо методологий в управлении и разработке проектов и других методов обеспечения качества, например, методов совместной работы, включая *парное программирование,* которое позволяет сохранять знания в компании, а не у людей.

Подавляющее большинство магистерских программ такого типа фокусируются на одной технологии, одном языке или одном инструменте. Настоящая программа уникальна тем, что позволяет профессионалу осознать важность качества программного обеспечения, сократить технический долг проектов с помощью качественного, а не экономического и временного подхода; она предоставляет учащемуся специальные знания, позволяющие обосновать бюджетирование проекта.

Для этого TECH Технологический университет собрал группу экспертов в этой области, которые передадут самые современные знания и опыт. Посредством современного виртуального кампуса с теоретическим и практическим содержанием, предоставленным в различных форматах. В вашем распоряжении будут 10 модулей, разделенных на различные темы и подтемы, которые позволят обучиться за 12 месяцев, используя методику *Relearning,* которая облегчает запоминание и обучение быстрым и эффективным способом.

Данная **Специализированная магистратура в области качества программного обеспечения** содержит самую полную и современную образовательную программу на рынке. Основными особенностями программы являются:

- Разработка практических кейсов, представленных экспертами в области разработки программного обеспечения
- Графическое, схематичное и исключительно практичное содержание курса предоставляет научную и практическую информацию по тем дисциплинам, которые необходимы для осуществления профессиональной деятельности
- Практические упражнения для самооценки, контроля и улучшения успеваемости
- Особое внимание уделяется инновационным методологиям
- Теоретические занятия, вопросы эксперту и самостоятельные работы
- Учебные материалы курса доступны с любого стационарного или мобильного устройства с выходом в интернет

Специализированная магистратура в области качества программного обеспечения анализирует критерии, лежащие в основе этой темы, на всех уровнях. Расширяйте свой профессиональный уровень. Поступайте сейчас"

Разработайте критерии, задачи и передовые методологии, чтобы понять актуальность работы, ориентированной на качество, и предоставить эффективные решения вашей компании или клиенту"

Программа, направленная на повышение осведомленности о важности качества программного обеспечения и необходимости внедрения политики качества на фабриках программного обеспечения.

Обучайтесь практическим и гибким способом. Мы сопровождаем вас в вашей повседневной жизни с помощью этого эксклюзивного 100% онлайн-обучения, предлагаемого TECH Технологическим университетом.

В преподавательский состав программы входят профессионалы отрасли, которые привносят в процесс обучения свой профессиональный опыт, а также признанные специалисты из ведущих сообществ и престижных университетов.

Мультимедийное содержание программы, разработанное с использованием новейших образовательных технологий, позволит специалисту проходить обучение с учетом контекста и ситуации, т.е. в симулированной среде, обеспечивающей иммерсивный учебный процесс, запрограммированный на обучение в реальных ситуациях.

Структура этой программы основана на проблемно-ориентированном обучении, с помощью которого специалист должен попытаться решить различные ситуации из профессиональной практики, возникающие в течение учебного курса. Для этого практикующему будет помогать инновационная система интерактивных видеоматериалов, созданная признанными и опытными специалистами.

02

Цели

Специализированная магистратура в области качества программного обеспечения дает студентам четкое и специализированное представление о важности качества в процессах разработки программного обеспечения. А также вы узнаете о самых передовых инструментах для внедрения процессов DevOps и систем обеспечения качества. Одним словом, программа обеспечит широкие и специализированные теоретические и практические знания, чтобы студенты рассматривали разработку проектов с современной и эффективной точки зрения.

Вы сможете легко получить доступ ко всему содержимому, когда захотите. С компьютера или любимого устройства. Вы также можете скачать материалы, чтобы ознакомиться с ними заранее,"

Общие цели

- Разработать критерии, задачи и передовые методологии для понимания актуальности работы, ориентированной на качество
- Проанализировать ключевые факторы в качестве программного проекта
- Разработать соответствующие нормативные аспекты
- Применить DevOps и системные процессы для обеспечения качества
- Уменьшить технический долг проектов с помощью подхода, основанного на качестве, а не на экономике и ограниченных сроках
- Предоставить студенту специфические знания, позволяющие измерять и количественно оценивать качество программного проекта
- Защищать экономические предложения проектов на основе качества

Конкретные цели

Модуль 1. Качество программного обеспечения. Уровни развития TRL

- Четко и ясно формулировать элементы, составляющие качество программного обеспечения
- Применять модели и стандарты в соответствии с системой, продуктом и программным процессом
- Углубиться в стандарты качества ISO, применяемые как в целом, так и в отдельных частях
- Применять стандарты в соответствии с масштабами окружающей среды (местные, национальные, международные)
- Изучить уровни зрелости TRL и адаптировать их к различным частям программного проекта, с которым предстоит работать
- Приобрести способность к абстракции для применения одного или нескольких критериев элементов и уровней качества программного обеспечения
- Различать случаи применения стандартов и уровней зрелости в реальном смоделированном проекте

Модуль 2. Проектирование программного обеспечения. Функциональная и техническая документация

- Определить влияние управления проектом на качество
- Разработать различные фазы проекта
- Различать понятия качества, присущие функциональной и технической документации
- Проанализировать этап сбора требований, этап анализа, управление командой и этап построения
- Установить различные методологии управления программными проектами
- Сформировать критерии для принятия решения о том, какая методология является наиболее подходящей в зависимости от типа проекта

Модуль 3. *Тестирование* программного обеспечения. Автоматизация тестирования

- Установить различия между качеством продукции, качеством процесса и качеством использования
- Знать стандарты ISO/IEC 15504
- Определить детали CMMI
- Ознакомиться с ключами к непрерывной интеграции, репозиториями и последствиями, которые они оказывают на команду разработчиков программного обеспечения
- Определить значимость внедрения репозиториев для программных проектов. Узнать, как создавать их с помощью TFS
- Усвоить важность масштабируемости программного обеспечения при проектировании и разработке информационных систем

Модуль 4. Методологии управления проектами программного обеспечения. *Каскадная* модель vs agile-методологии

- Определить, из чего состоит методология каскадной модели
- Изучить методологию SCRUM
- Установить различия между каскадной моделью и SCRUM
- Указать различия между каскадной моделью и Scrum и как на это смотрит клиент
- Изучить Канбан-доску
- Использовать в одном и том же проекте каскадную модель и Scrum
- Создать гибридный проект

Модуль 5. TDD (*Test Driven Developement*). Разработка программного обеспечения через тестирование

- Узнать о практическом применении TDD и его возможностях, дальнейшем тестировании программного проекта
- Дополнять предложенные реальные симуляционные примеры в качестве непрерывного обучения этой концепции TDD
- Проанализировать на симуляционных примерах, в какой степени тесты могут быть успешными или неудачными, с конструктивной точки зрения
- Определить альтернативы TDD, проведя сравнительный анализ между ними

Модуль 6. DevOps. Управление качеством программного обеспечения

- Проанализировать недостатки традиционного процесса
- Оценить возможные решения и выбрать наиболее подходящее
- Понимать потребности бизнеса и их влияние на внедрение
- Оценить затраты на улучшения, которые необходимо осуществить
- Разработать эволюционирующий жизненный цикл программного обеспечения, адаптированный к реальным потребностям
- Предвидеть возможные ошибки и избегать их в процессе проектирования
- Обосновать использование различных моделей реализации

Модуль 7. DevOps и непрерывная интеграция. Передовые практические решения в разработке программного обеспечения

- Определить этапы цикла разработки и поставки программного обеспечения, адаптированные к конкретным случаям
- Разработать процесс поставки программного обеспечения с использованием непрерывной интеграции
- Создать и внедрить непрерывную интеграцию и развертывание на основе предыдущего проекта
- Установить автоматические контрольные точки качества в каждой поставке программного обеспечения
- Поддерживать автоматизированный и надежный процесс поставки программного обеспечения
- Адаптировать будущие потребности к процессу непрерывной интеграции и развертывания
- Проанализировать и предвидеть уязвимости безопасности во время и после процесса поставки программного обеспечения

Модуль 8. Проектирование баз данных (БД). Стандартизация и производительность. Качество программного обеспечения

- Оценить использование модели "сущность-связь" для предварительного проектирования базы данных
- Применить сущность, атрибут, ключ и т.д. для обеспечения наилучшей целостности данных
- Оценить зависимости, формы и правила нормализации баз данных
- Специализироваться на эксплуатации системы хранилища данных OLAP, разрабатывая и используя таблицы фактов и размерностей
- Определить ключевые моменты для производительности базы данных
- Выполнять предлагаемые примеры моделирования реального мира в качестве непрерывного обучения в области проектирования, нормализации и производительности баз данных
- Установить в имитационных примерах варианты, которые необходимо решить при создании базы данных с конструктивной точки зрения

Модуль 9. Проектирование масштабируемых архитектур. Архитектура в жизненном цикле программного обеспечения

- Разработать концепцию архитектуры программного обеспечения и ее характеристики
- Определить различные типы масштабируемости в архитектуре программного обеспечения
- Проанализировать различные уровни, которые могут возникать при масштабировании веб-сайтов
- Приобрести специализированные знания о концепции жизненного цикла программного обеспечения, его этапах и моделях

- Определить влияние архитектуры на жизненный цикл программного обеспечения с учетом ее преимуществ, ограничений и вспомогательных средств
- Выполнить предложенные реальные примеры моделирования в качестве непрерывного обучения об архитектуре и жизненном цикле программного обеспечения
- Оценить на примере моделирования, в какой степени проектирование архитектуры может быть целесообразным или ненужным

Модуль 10. Критерии качества ISO/IEC 9126. Метрики качества программного обеспечения

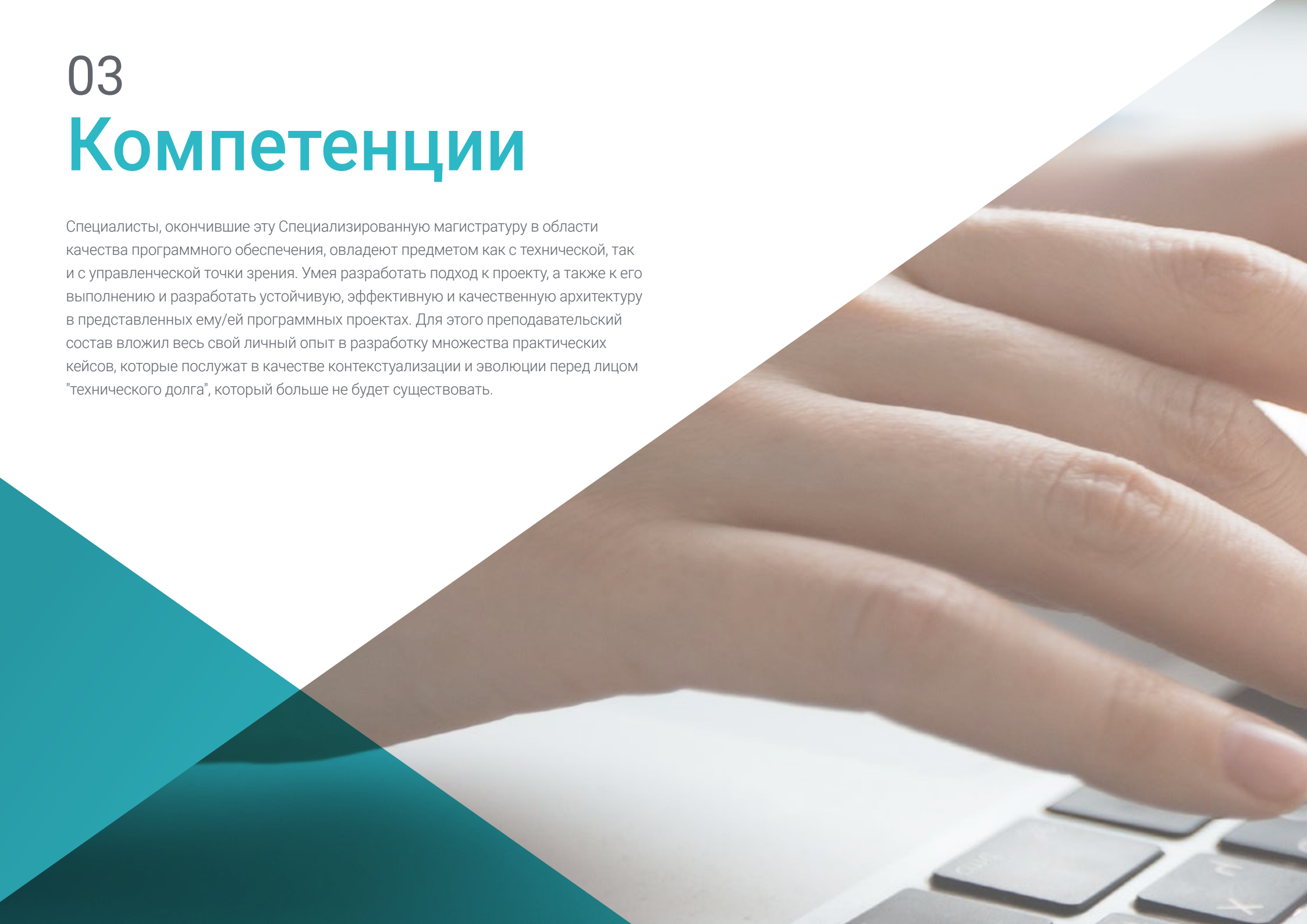
- Разработать концепцию критериев качества и соответствующих аспектов
- Изучить стандарт ISO/IEC 9126, основные аспекты и показатели
- Анализировать различные показатели для того, чтобы проект программного обеспечения соответствовал согласованным оценкам
- Изучить внутренние и внешние атрибуты, на которые следует обратить внимание при оценке качества проекта программного обеспечения
- Различать метрики в зависимости от типа программирования (структурированное, объектно-ориентированное, многоуровневое и т.д.)
- Выполнить реальные примеры моделирования в качестве непрерывного обучения измерению качества
- Посмотреть на примере моделирования, в какой степени это осуществимо или нет, т.е. с конструктивной точки зрения авторов

Выделите свой профессиональный профиль с помощью этой эксклюзивной программы. Получите диплом за 12 месяцев и практическим путем с методологией, которую может предложить вам только TECH Технологический университет"

03

Компетенции

Специалисты, окончившие эту Специализированную магистратуру в области качества программного обеспечения, овладеют предметом как с технической, так и с управленческой точки зрения. Умея разработать подход к проекту, а также к его выполнению и разработать устойчивую, эффективную и качественную архитектуру в представленных ему/ей программных проектах. Для этого преподавательский состав вложил весь свой личный опыт в разработку множества практических кейсов, которые послужат в качестве контекстуализации и эволюции перед лицом "технического долга", который больше не будет существовать.

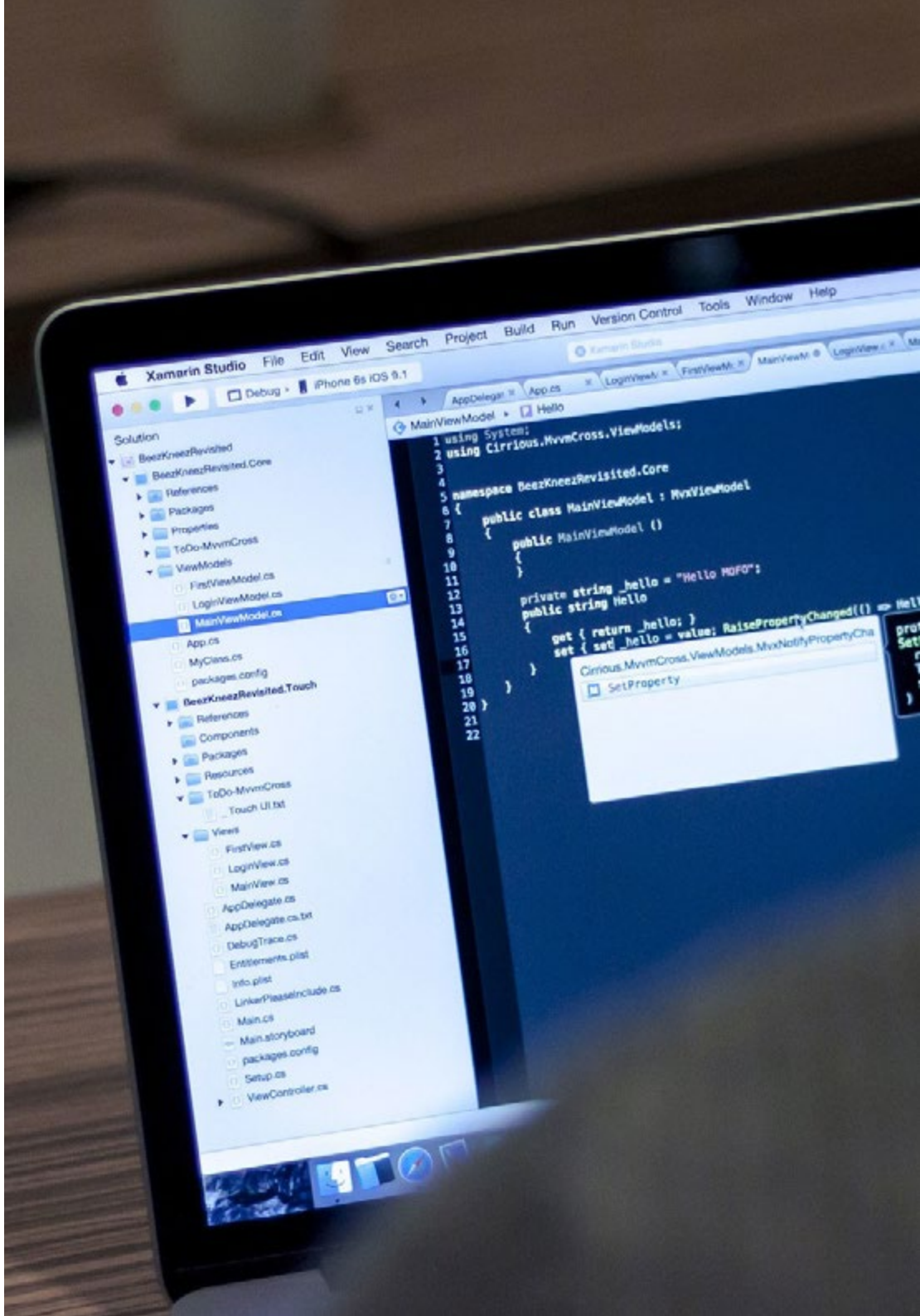


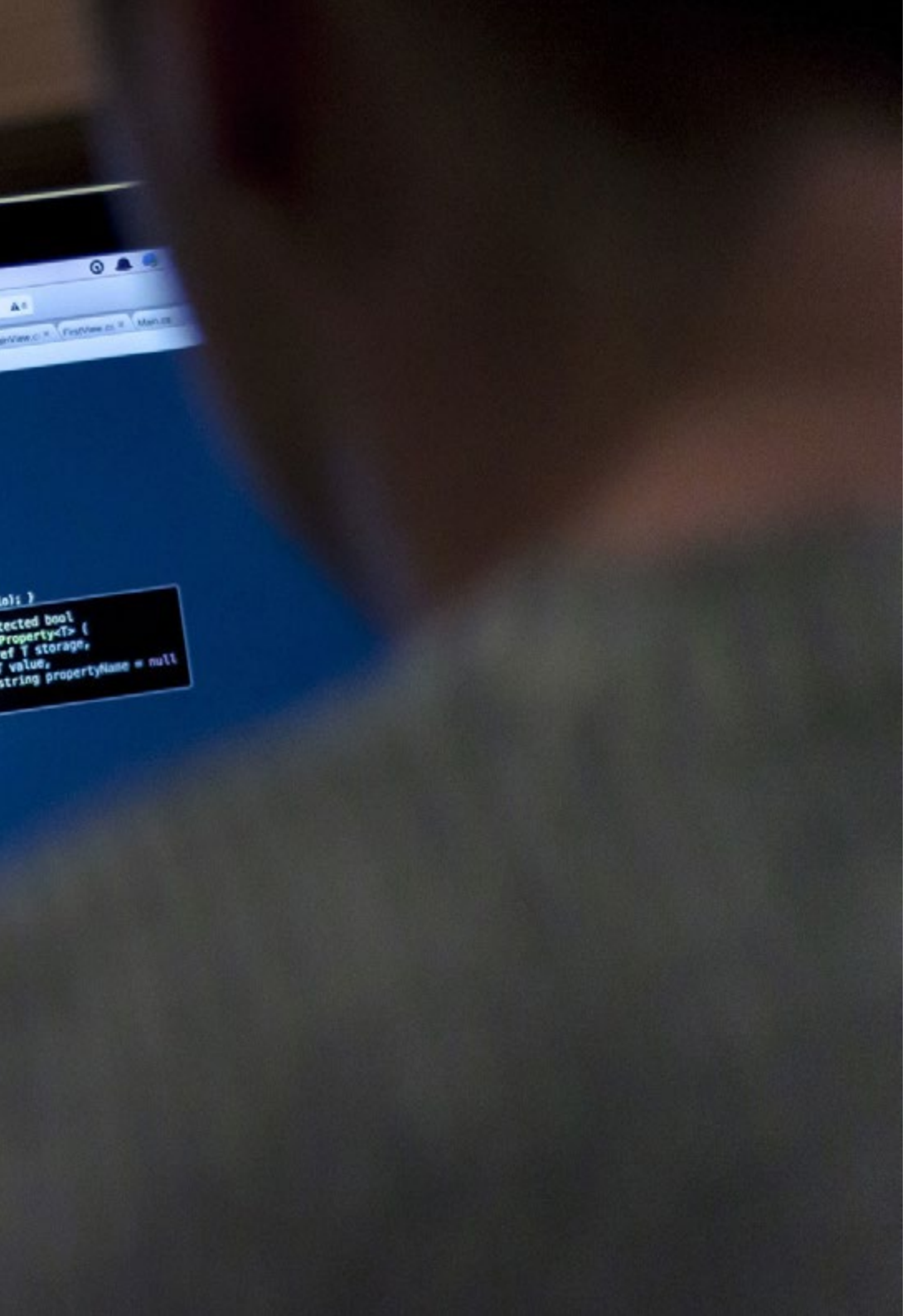
Профессионал в области информатики, ориентированный на качество — это растущий актив для консультационных компаний по программному обеспечению и крупных корпораций. Запишитесь сейчас на эту Специализированную магистратуру в области качества программного обеспечения"

Общие профессиональные навыки

- Уменьшить технический долг проектов с помощью подхода, основанного на качестве, а не на экономике и ограниченных сроках
- Измерять и количественно оценивать качество программного проекта
- Правильно выполнять TDD, для повышения стандартов качества программного обеспечения
- Составлять бюджеты проектов, ориентированных на качество
- Разрабатывать нормы, модели и стандарты качества
- Изучить различные оценки зрелости технологий
- Снизить риск и обеспечить обслуживание и контроль последующих версий
- Освоить фазы, на которые разбивается проект

Совершенствуйте ваши навыки и откройте для себя бесконечные возможности для профессионального роста, которые открываются с этим новым опытом"





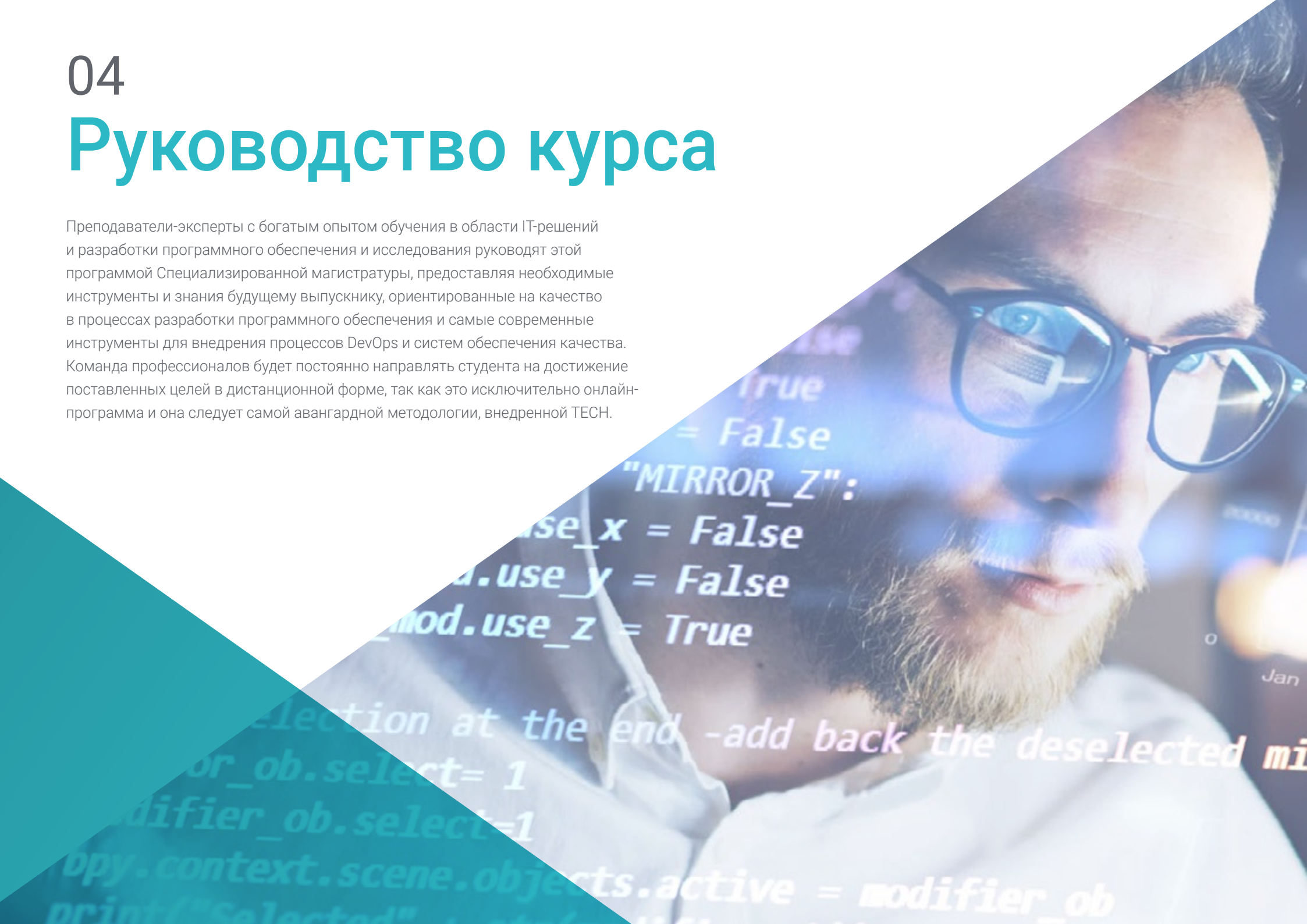
Профессиональные навыки

- Оценить программную систему с точки зрения степени прогресса в процессе реализации проекта
- Правильно и стратегически решать эти вопросы надежности, метрик и обеспечения в программных проектах
- Рассмотреть вопрос о принятии решения о методологии, которая будет использоваться в проекте
- Освоить основные нормативные аспекты для создания программного обеспечения
- Разработать *тестирование* в автоматическом режиме
- Установить адекватную коммуникацию с клиентом, понимая, как он/она воспринимает проект в соответствии с применяемой методологией
- Составить список требований к тестам
- Выполнить абстракцию, разделение на более унитарные тесты и устранить то, что не относится к хорошему выполнению тестов выполняемого программного проекта
- Обновлять список требований к тестам продуманно и корректно
- Адаптировать культуру DevOps к потребностям бизнеса
- Разработать новейшие методы и инструменты для непрерывной интеграции и развертывания
- Осуществлять рефакторинг и заниматься управлением и координацией данных

04

Руководство курса

Преподаватели-эксперты с богатым опытом обучения в области IT-решений и разработки программного обеспечения и исследования руководят этой программой Специализированной магистратуры, предоставляя необходимые инструменты и знания будущему выпускнику, ориентированные на качество в процессах разработки программного обеспечения и самые современные инструменты для внедрения процессов DevOps и систем обеспечения качества. Команда профессионалов будет постоянно направлять студента на достижение поставленных целей в дистанционной форме, так как это исключительно онлайн-программа и она следует самой авангардной методологии, внедренной TECH.



Преподаватели-специалисты стремятся предоставить вам наилучшее содержание и сделать процесс обучения гибким и динамичным. Проясняя ваши вопросы и сопровождая вас на протяжении всего пути"

Руководство

Г-н Молина Молина, Херонимо

- QA-инженер и Архитектор ПО NASSAT - Спутниковая связь в движении
- Старший консультант в Hexa Ingenieros. Внедрение искусственного интеллекта (ML и CV)
- Эксперт по решениям на основе искусственного интеллекта в области Computer Vision, ML/DL и NLP. В настоящее время проводит исследования в области возможностей применения Transformers и Reinforcement Learning в личном исследовательском проекте
- Курс профессиональной подготовки в области создания и развития бизнеса. Bancaixa – FUNDEUN Аликанте
- Компьютерный инженер. Университет Аликанте
- Степень магистра в области искусственного интеллекта. Католический университет Авила
- Executive MBA. Европейский форум Бизнес Кампус

Преподаватели

Г-н Пи Морель, Ориоль

- Владелец продукта хостинга и почты. CDMON
- Функциональный аналитик и инженер-программист в различных организациях, таких как Fihoca, Atmira, CapGemini.
- Преподаватель различных курсов, таких как BPM в CapGemini, ORACLE Forms CapGemini, Business Processes Atmira.
- Степень бакалавра в области компьютерной инженерии в Автономном университете Мадрида
- Степень магистра в области искусственного интеллекта
- Степень магистра в области управления и администрирования бизнеса. MBA
- Степень магистра в области управления информационными системами, Опыт преподавания
- Последипломное образование, Курс последипломного образования в области паттернов проектирования Открытый университет Каталонии

Г-н Тенреро Моран, Маркос

- Инженер по DevOps – Allot Communications
- Application Lifecycle Management & DevOps – Meta4 Испания. Cegid
- Инженер по автоматизации QA – Meta4 Испания. Cegid
- Степень в области компьютерной инженерии Университета короля Хуана Карлоса
- Разработка профессиональных приложений для Android– Университет Галилео (Гватемала)
- Разработка облачных сервисов (nodeJs, JavaScript, HTML5) - UPM
- Непрерывная интеграция с Jenkins – Meta4. Cegid

Д-р Перальта Мартин-Паломино, Артуро

- CEO и CTO Prometeus Global Solutions
- CTO в Korporate Technologies
- CTO в AI Shephers GmbH
- Доктор в области компьютерной инженерии Университета Кастильи-ла-Манчи
- Доктор в области экономики, бизнеса и финансов Университета Камило Хосе Села. Награда за выдающуюся докторскую степень
- Доктор психологии Университета Кастилии-ла-Манчи
- Степень магистра в области передовых информационных технологий Университета Кастильи-ла-Манчи
- Магистр MBA+E (магистр в области делового администрирования и организационной инженерии) Университета Кастилья-ла-Манча
- Доцент, преподающий в Университете Кастильи-ла-Манчи программы бакалавриата и магистратуры по компьютерной инженерии
- Преподаватель магистратуры в области больших данных и науки о данных в Международном университете Валенсии
- Преподаватель магистратуры «Индустрия 4.0» и магистратуры «Промышленный дизайн и разработка продуктов»
- Член исследовательской группы SMILe Университета Кастилии-ла-Манчи
- Веб-разработка с использованием Angular-CLI (4), Ionic и nodeJS. Meta4 - Университет короля Хуана Карлоса

Г-жа Мартинес Серрато, Йесика

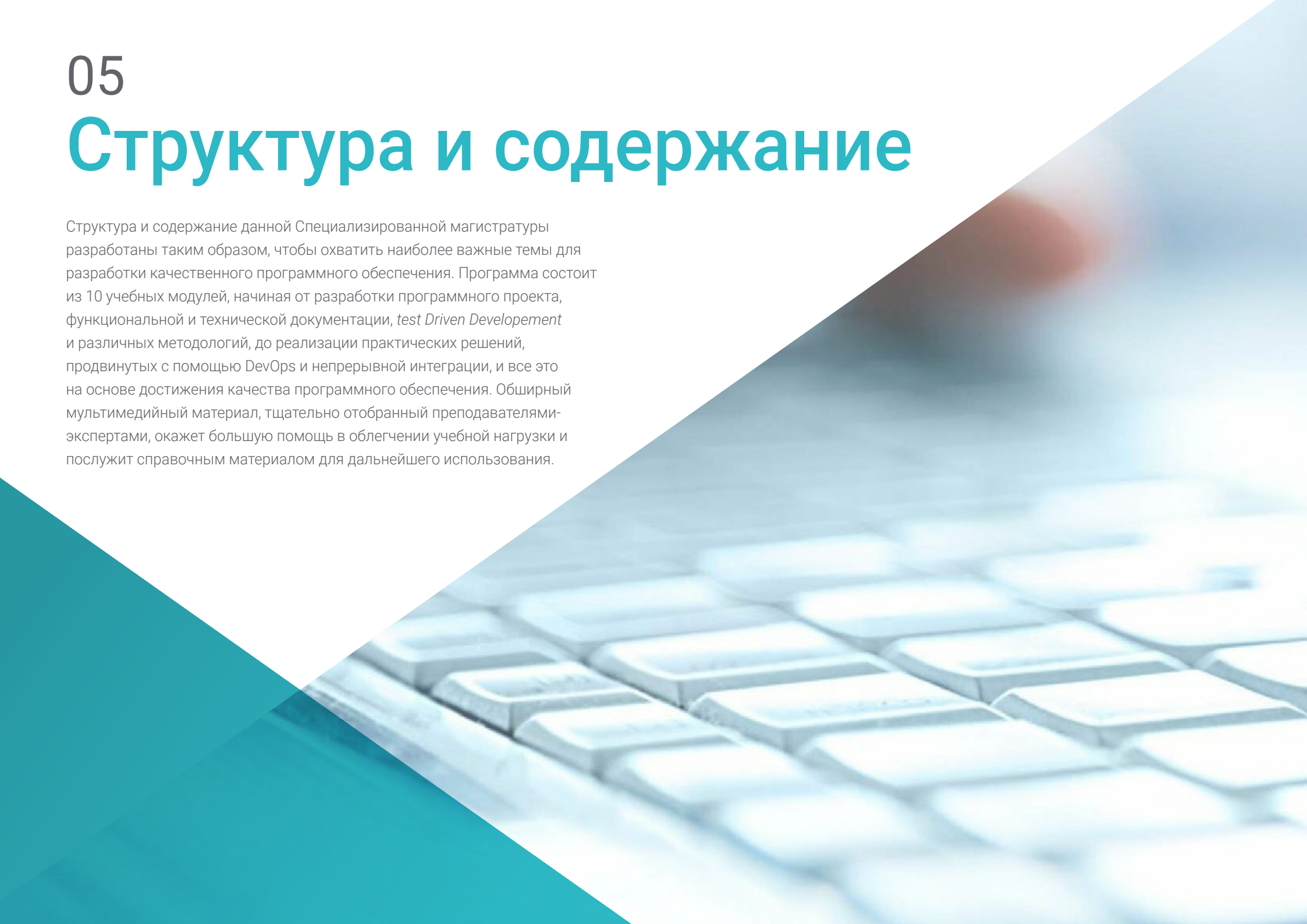
- Технический специалист в области электронных средств безопасности в компании Securitas Seguridad España
- Аналитик бизнес-аналитики в Ricopia Technologies (Алькала-де-Энарес). Степень бакалавра в области инженерии электронных коммуникаций в Высшей политехнической школе, Университет Алькала
- Ответственная за обучение новых сотрудников программному обеспечению для управления продажами (CRM, ERP, INTRANET), продуктам и процедурам в компании Ricopia Technologies (Алькала-де-Энарес)
- Ответственная за обучение новых стипендиатов, принятых в компьютерные классы Университета Алькала
- Руководитель проекта в области интеграции ключевых клиентов в компании Correos y Telégrafos (Мадрид)
- Специалист в области IT - ответственная за компьютерные классы OTEC, Университет Алькала (Алькала-де-Энарес)
- Преподаватель компьютерных классов в Ассоциации ASALUMA (Алькала-де-Энарес).
- Стипендия на обучение по специальности «ИТ « в OTEC, Университет Алькала (Алькала-де-Энарес)

Наша команда преподавателей передаст вам все свои знания, чтобы вы ознакомились с самыми актуальными данными в этой сфере”

05

Структура и содержание

Структура и содержание данной Специализированной магистратуры разработаны таким образом, чтобы охватить наиболее важные темы для разработки качественного программного обеспечения. Программа состоит из 10 учебных модулей, начиная от разработки программного проекта, функциональной и технической документации, *test Driven Developement* и различных методологий, до реализации практических решений, продвинутых с помощью DevOps и непрерывной интеграции, и все это на основе достижения качества программного обеспечения. Обширный мультимедийный материал, тщательно отобранный преподавателями-экспертами, окажет большую помощь в облегчении учебной нагрузки и послужит справочным материалом для дальнейшего использования.



Практические кейсы, основанные на реальности, послужат закреплением и контекстуализацией всей теории, изученной в ходе программы"

Модуль 1. Качество программного обеспечения. Уровни развития TRL

1.1. Элементы, влияющие на качество программного обеспечения (I). Технический долг
- 1.1.1. Технический долг. Причины и последствия
- 1.1.2. Качество программного обеспечения. Общие принципы
- 1.1.3. Беспринципное и принципиальное качество программного обеспечения
  - 1.1.3.1. Последствия
  - 1.1.3.2. Необходимость применения принципов качества в программном обеспечении
- 1.1.4. Качество программного обеспечения. Типология
- 1.1.5. Качественное программное обеспечение. Специфические особенности

1.2. Элементы, влияющие на качество программного обеспечения (II). Сопутствующие расходы
- 1.2.1. Качество программного обеспечения. Влияющие элементы
- 1.2.2. Качество программного обеспечения. Заблуждения
- 1.2.3. Качество программного обеспечения. Сопутствующие расходы

1.3. Модели качества программного обеспечения (I). Управление знаниями
- 1.3.1. Общие модели качества
  - 1.3.1.1. Всеобщее управление качеством
  - 1.3.1.2. Европейская модель делового совершенства (EFQM)
  - 1.3.1.3. Концепция шести сигм
- 1.3.2. Модели управления знаниями
  - 1.3.2.1. Модель Dyba
  - 1.3.2.2. Модель SEKS
- 1.3.3. Опыт работы в парадигме Factory и QIP
- 1.3.4. Качество используемых моделей (25010)

1.4. Модели качества программного обеспечения (III). Качество данных, процессов и моделей SEI
- 1.4.1. Модель качества данных
- 1.4.2. Моделирование процессов программного обеспечения
- 1.4.3. Спецификация метамодели программной и системной инженерии процессов (SPEM)
- 1.4.4. Модели SEI
  - 1.4.4.1. CMMI
  - 1.4.4.2. SCAMPI
  - 1.4.4.3. IDEAL

1.5. Стандарты качества программного обеспечения ISO (I). Анализ стандартов
- 1.5.1. Стандарт ISO 9000
  - 1.5.1.1. Стандарт ISO 9000
  - 1.5.1.2. Семейство стандартов качества ISO (9000)
- 1.5.2. Другие стандарты ISO, связанные с качеством
- 1.5.3. Стандарты моделирования качества (ISO 2501)
- 1.5.4. Стандарты измерения качества (ISO 2502n)

1.6. Стандарты качества программного обеспечения ISO (II). Требования и оценка
- 1.6.1. Стандарты по требованиям к качеству (2503n)
- 1.6.2. Стандарты по оценке качества (2504n)
- 1.6.3. ISO/IEC 24744:2007

1.7. Уровни развития TRL (I). Уровни с 1 по 4
- 1.7.1. Уровни TRL
- 1.7.2. Уровень 1: основные принципы
- 1.7.3. Уровень 2: концепция и/или применение
- 1.7.4. Уровень 3: критическая аналитическая функция
- 1.7.5. Уровень 4: проверка основных компонентов в лабораторных условиях

1.8. Уровни развития TRL (II). Уровни с 5 по 9
- 1.8.1. Уровень 5: проверка основных компонентов в реальных условиях
- 1.8.2. Уровень 6: модель системы/подсистемы
- 1.8.3. Уровень 7: демонстрация в условиях эксплуатации
- 1.8.4. Уровень 8: полная и сертифицированная система
- 1.8.5. Уровень 9: успех в реальных условиях

1.9. Уровни развития TRL. Использование
- 1.9.1. Пример компании с лабораторной средой
- 1.9.2. Пример компании, занимающейся НИОКР
- 1.9.3. Пример промышленной компании, занимающейся НИОКР
- 1.9.4. Пример совместной лабораторно-инженерной компании

1.10. Качество программного обеспечения. Ключевые детали
- 1.10.1. Методологические детали
- 1.10.2. Технические детали
- 1.10.3. Детали управления программными проектами
  - 1.10.3.1. Качество ИТ-систем
  - 1.10.3.2. Качество программного продукта
  - 1.10.3.3. Качество программного процесса

Модуль 2. Проектирование программного обеспечения. Функциональная и техническая документация

2.1. Управление проектами
- 2.1.1. Управление проектами в области качества программного обеспечения
- 2.1.2. Управление проектами. Преимущества
- 2.1.3. Управление проектами. Типология

2.2. Методика управления проектами
- 2.2.1. Методика управления проектами
- 2.2.2. Проектные методики. Типология
- 2.2.3. Методики управления проектами. Область применения

2.3. Фаза выявления требований
- 2.3.1. Определение требований к проекту
- 2.3.2. Управление проектными совещаниями
- 2.3.3. Предоставляемая документация

2.4. Модель
- 2.4.1. Начальный этап
- 2.4.2. Этап анализа
- 2.4.3. Этап конструкции
- 2.4.4. Этап тестирования
- 2.4.5. Предоставление

2.5. Используемая модель данных
- 2.5.1. Определение новой модели данных
- 2.5.2. Определение плана миграции данных
- 2.5.3. Набор данных

2.6. Влияние на другие проекты
- 2.6.1. Влияние проекта. Примеры
- 2.6.2. Риски в проекте
- 2.6.3. Управление рисками

2.7. "*Must*" проекта
- 2.7.1. *Must* проекта
- 2.7.2. Идентификация *Must* проекта
- 2.7.3. Определение точек реализации проекта

2.8. Команда для строительства проекта
- 2.8.1. Роли, которые необходимо выполнять в соответствии с проектом
- 2.8.2. Контакт с отделом кадров для подбора персонала
- 2.8.3. Результаты и график проекта

2.9. Технические аспекты программного проекта
- 2.9.1. Архитектор проекта. Технические аспекты
- 2.9.2. Технические лидеры
- 2.9.3. Проектирование программного проекта
- 2.9.4. Оценка качества кода, Sonar

2.10. Результаты проекта
- 2.10.1. Функциональный анализ
- 2.10.2. Модель данных
- 2.10.3. Диаграмма состояний
- 2.10.4. Техническая документация

Модуль 3. *Тестирование программного обеспечения* Автоматизация тестирования

3.1. Модели качества программного обеспечения
- 3.1.1. Качество продукта
- 3.1.2. Качество процесса
- 3.1.3. Качество использования

3.2. Качество процесса
- 3.2.1. Качество процесса
- 3.2.2. Модели зрелости

3.2.3. Стандарт ISO 15504
3.2.3.1. Цели
3.2.3.2. Контекст
3.2.3.3. Этапы
3.3. Стандарт ISO/IEC 15504
3.3.1. Категории процессов
3.3.2. Процесс разработки. Пример
3.3.3. Фрагмент программы
3.3.4. Этапы
3.4. CMMI (*Capability Maturity Model Integration*)
3.4.1. CMMI. Интеграция модели зрелости возможностей
3.4.2. Модели и области. Типология
3.4.3. Области процесса
3.4.4. Уровни мощности
3.4.5. Управление процессами
3.4.6. Управление проектами
3.5. Управление изменениями и репозиториями
3.5.1. Управление изменениями в программном обеспечении
3.5.1.1. Элемент конфигурации. Непрерывная интеграция
3.5.1.2. Линии
3.5.1.3. Блок-схемы
3.5.1.4. *Бранчи*
3.5.2. Репозиторий
3.5.2.1. Контроль версий
3.5.2.2. Работа в команде и использование репозитория
3.5.2.3. Непрерывная интеграция в репозитории
3.6. *Team Foundation Server* (TFS)
3.6.1. Установка и настройка
3.6.2. Создание командного проекта
3.6.3. Добавление содержимого в систему управления источниками
3.6.4. *TFS в облаке*
3.7. *Тестирование*
3.7.1. Мотивы для тестирования
3.7.2. Проверочное тестирование
3.7.3. Бета-тестирование
3.7.4. Реализация и обслуживание
3.8. Нагрузочное тестирование
3.8.1. *Нагрузочное тестирование*
3.8.2. Тестирование с помощью *LoadView*
3.8.3. Тестирование с помощью K6 *Cloud*
3.8.4. Тестирование с помощью *Loader*
3.9. Испытания устройства, на стресс и выносливость
3.9.1. Мотивы для проведения модульного тестирования
3.9.2. Инструменты для *модульного тестирования*
3.9.3. Мотивы для проведения тестирования на стресс
3.9.4. *Стресс-тестирование*
3.9.5. Мотивы для проведения тестов на устойчивость
3.9.6. Тестирование с помощью *LoadRunner*
3.10. Масштабируемость. Масштабируемое проектирование программного обеспечения
3.10.1. Масштабируемость и архитектура программного обеспечения
3.10.2. Независимость между слоями
3.10.3. Связь между слоями. Архитектурные паттерны

Модуль 4. Методологии управления проектами программного обеспечения. *Каскадная* модель vs agile-методологии

4.1. Методология *каскадной модели*
4.1.1. Методология *каскадной модели*
4.1.2. Методология *каскадной модели*. Влияние на качество программного обеспечения
4.1.3. Методология *каскадной модели*. Примеры

4.2. Методологии *Agile*
- 4.2.1. Методологии *Agile*
- 4.2.2. Методологии *Agile*. Влияние на качество программного обеспечения
- 4.2.3. Методологии *Agile*. Примеры

4.3. Методология Scrum
- 4.3.1. Методология Scrum
- 4.3.2. Манифест Scrum
- 4.3.3. Применения Scrum

4.4. Kanban-доска
- 4.4.1. Kanban-метод
- 4.4.2. Kanban-доска
- 4.4.3. Kanban-доска. Примеры применения

4.5. Управление проектами в *каскадной модели*
- 4.5.1. Фазы проекта
- 4.5.2. Видение в проекте *каскадной модели*
- 4.5.3. Результаты, которые необходимо рассмотреть

4.6. Управление проектами в Scrum
- 4.6.1. Фазы проекта Scrum
- 4.6.2. Видение проекта Scrum
- 4.6.3. Результаты, которые необходимо рассмотреть

4.7. *Каскадная модель* vs. Scrum сравнение
- 4.7.1. Пилотный проект
- 4.7.2. Проект с применением *каскадной модели*. Пример
- 4.7.3. Проект с применением Scrum. Пример

4.8. Видение клиента
- 4.8.1. Документы в *каскадной модели*
- 4.8.2. Документы в Scrum
- 4.8.3. Сравнение

4.9. Структура в Kanban
- 4.9.1. Истории пользователей
- 4.9.2. *Backlog*
- 4.9.3. Анализ Kanban

4.10. Гибридные проекты
- 4.10.1. Конструкция проекта
- 4.10.2. Управление проектами
- 4.10.3. Результаты, которые необходимо рассмотреть

Модуль 5. TDD (*Test Driven Developement*). Разработка программного обеспечеия через тестирование

5.1. TDD. *Разработка программного обеспечения через тестирование*
- 5.1.1. TDD. *Разработка программного обеспечения через тестирование*
- 5.1.2. TDD. Влияние TDD на качество
- 5.1.3. Проектирование и разработка на основе тестирования. Примеры

5.2. Цикл TDD
- 5.2.1. Выбор требования
- 5.2.2. Тестирование. Типологии
  - 5.2.2.1. Модульные тесты
  - 5.2.2.2. Интеграционное тестирование
  - 5.2.2.3. *Сквозное тестирование* (End To End)
- 5.2.3. Проверка тестирования. Ошибки
- 5.2.4. Создание внедрения
- 5.2.5. Выполнение автоматизированных тестов
- 5.2.6. Устранение дублирования
- 5.2.7. Обновление списка требований
- 5.2.8. Повторение цикла TDD
- 5.2.9. Цикл TDD. Теоретический и практический пример

5.3. Стратегии внедрения TDD
- 5.3.1. Ошибочное внедрение
- 5.3.2. Треугольное внедрение
- 5.3.3. Очевидное внедрение

5.4. TDD. Применение. Преимущества и недостатки
- 5.4.1. Преимущества использования
- 5.4.2. Ограничения на использование
- 5.4.3. Баланс качества при реализации

5.5. TDD. Передовая практика
- 5.5.1. Правила TDD
- 5.5.2. Правило 1: проводить предыдущий неудачный тест, прежде чем приступать к кодированию в производстве
- 5.5.3. Правило 2: не писать более одного модульного теста
- 5.5.4. Правило 3: не писать больше кода, чем требуется
- 5.5.5. Ошибки и антипаттерны, которых следует избегать в TDD

5.6. Моделирование реального проекта для использования TDD (I)
- 5.6.1. Обзор проекта (Компания A)
- 5.6.2. Применение TDD
- 5.6.3. Предлагаемые тесты
- 5.6.4. Тесты. *Обратная связь*

5.7. Моделирование реального проекта для использования TDD (II)
- 5.7.1. Обзор проекта (Компания B)
- 5.7.2. Применение TDD
- 5.7.3. Предлагаемые тесты
- 5.7.4. Тесты. *Обратная связь*

5.8. Моделирование реального проекта для использования TDD (III)
- 5.8.1. Обзор проекта (Компания C)
- 5.8.2. Применение TDD
- 5.8.3. Предлагаемые упражнения
- 5.8.4. Тесты. *Обратная связь*

5.9. Альтернативы TDD. *Разработка программного обеспечения через тестирование*
- 5.9.1. TCR (*Test Commit Revert*)
- 5.9.2. BDD (*Behavior Driven Development*)
- 5.9.3. ATDD (*Acceptance Test Driven Development*)
- 5.9.4. TDD. Теоретическое сравнение

5.10. TDD TCR, BDD y ATDD. Практическое сравнение
- 5.10.1. Определение проблемы
- 5.10.2. Решение с помощью TCR
- 5.10.3. Решение с помощью BDD
- 5.10.4. Решение с помощью ATDD

Модуль 6. DevOps Управление качеством программного обеспечения

6.1. DevOps. Управление качеством программного обеспечения
- 6.1.1. DevOps
- 6.1.2. DevOps и качество программного обеспечения
- 6.1.3. DevOps. Преимущества культуры DevOps

6.2. DevOps. Отношения с Agile
- 6.2.1. Ускоренная поставка
- 6.2.2. Качество
- 6.2.3. Снижение затрат

6.3. Запуск DevOps
- 6.3.1. Определение проблемы
- 6.3.2. Внедрение в компании
- 6.3.3. Метрики внедрения

6.4. Цикл доставки программного обеспечения
- 6.4.1. Методы проектирования
- 6.4.2. Соглашения
- 6.4.3. Дорожная карта проекта

6.5. Разработка кода без ошибок
- 6.5.1. Удобный в обслуживании код
- 6.5.2. Схемы развития
- 6.5.3. *Тестирование* кода
- 6.5.4. Разработка программного обеспечения на уровне кода. Передовая практика

6.6. Автоматизация
- 6.6.1. Автоматизация. Виды тестирования
- 6.6.2. Стоимость автоматизации и технического обслуживания
- 6.6.3. Автоматизация. Смягчение ошибок

6.7. Развертывания
- 6.7.1. Оценка цели
- 6.7.2. Разработка автоматического и адаптированного процесса
- 6.7.3. Обратная связь и оперативность

6.8. Управление инцидентами
- 6.8.1. Готовность к инцидентам
- 6.8.2. Анализ и разрешение инцидентов
- 6.8.3. Как избежать ошибок в будущем

6.9. Автоматизация развертывания
- 6.9.1. Подготовка к автоматическому развертыванию
- 6.9.2. Автоматическая оценка состояния процессов
- 6.9.3. Метрики и возможность отката

6.10. Передовая практика. Эволюция DevOps
- 6.10.1. Руководство по использованию DevOps
- 6.10.2. DevOps. Методология для команды
- 6.10.3. Избегание ниш

Модуль 7. DevOps и непрерывная интеграция. Передовые практические решения в разработке программного обеспечения

7.1. Поток поставки программного обеспечения
- 7.1.1. Идентификация действующих лиц и артефактов
- 7.1.2. Проектирование потока поставки программного обеспечения
- 7.1.3. Поток поставки программного обеспечения, межэтапные требования

7.2. Автоматизация процессов
- 7.2.1. Непрерывная интеграция
- 7.2.2. Непрерывное развертывание
- 7.2.3. Конфигурирование сред и управление секретами

7.3. Декларативные конвейеры
- 7.3.1. Различия между традиционными, кодоподобными и декларативными конвейерами
- 7.3.2. Декларативные конвейеры
- 7.3.3. Декларативный конвейер в Jenkins
- 7.3.4. Сравнение поставщиков услуг непрерывной интеграции

7.4. Качественные ворота и богатая обратная связь
- 7.4.1. Качественные ворота
- 7.4.2. Стандарты качества с качественными воротами. Техническое обслуживание
- 7.4.3. Бизнес-требования в запросах на интеграцию

7.5. Управление артефактами
- 7.5.1. Артефакты и жизненный цикл
- 7.5.2. Системы хранения и управления артефактами
- 7.5.3. Безопасность в управлении артефактами

7.6. Непрерывное развертывание
- 7.6.1. Непрерывное развертывание в виде контейнеров
- 7.6.2. Непрерывное развертывание с помощью PaaS
- 7.6.3. Непрерывное развертывание мобильных приложений

7.7. Улучшение времени выполнения конвейера: статический анализ и *Git Hooks*
- 7.7.1. Статический анализ
- 7.7.2. Правила стиля кода
- 7.7.3. *Git Hooks* и *модульные тесты*
- 7.7.4. Влияние инфраструктуры

7.8. Уязвимости контейнеров
- 7.8.1. Уязвимости контейнеров
- 7.8.2. Сканирование изображений
- 7.8.3. Периодические отчеты и оповещения

Модуль 8. Проектирование баз данных (БД). Стандартизация и производительность. Качество программного обеспечения

8.1. Проектирование баз данных
- 8.1.1. Базы данных. Типология
- 8.1.2. Используемые в настоящее время базы данных
  - 8.1.2.1. Реляционные
  - 8.1.2.2. Ключ-значение
  - 8.1.2.3. На основе сети
- 8.1.3. Качество данных

8.2. Проектирование модели сущность - связь (I)
- 8.2.1. Модель сущность - связь. Качество и документация
- 8.2.2. Сущности
  - 8.2.2.1. Сильная сущность
  - 8.2.2.2. Слабая сущность
- 8.2.3. Атрибуты
- 8.2.4. Набор отношений
  - 8.2.4.1. 1 к 1
  - 8.2.4.2. 1 ко многим
  - 8.2.4.3. Многие к 1
  - 8.2.4.4. Многие ко многим
- 8.2.5. Ключевые моменты
  - 8.2.5.1. Первичный ключ
  - 8.2.5.2. Внешний ключ
  - 8.2.5.3. Первичный ключ слабой сущности
- 8.2.6. Ограничения
- 8.2.7. Кардинальность
- 8.2.8. Наследственность
- 8.2.9. Агрегация

8.3. Модель сущность - связь (II). Инструменты
- 8.3.1. Модель сущность-связь. Инструменты
- 8.3.2. Модель сущность-связь. Наглядный пример
- 8.3.3. Реальная модель сущность-связь
  - 8.3.3.1. Визуальный пример
  - 8.3.3.2. Образец в представлении таблицы

8.4. Стандартизация базы данных (БД) (I). Соображения, касающиеся качества программного обеспечения
- 8.4.1. Стандартизация БД и качество
- 8.4.2. Зависимости
  - 8.4.2.1. Функциональная зависимость
  - 8.4.2.2. Свойства функциональной зависимости
  - 8.4.2.3. Предполагаемые свойства
- 8.4.3. Ключевые моменты

8.5. Стандартизация базы данных (БД) (II). Нормальная форма и правила Codd
- 8.5.1. Нормальные формы
  - 8.5.1.1. Первая нормальная форма (1НФ)
  - 8.5.1.2. Вторая нормальная форма (2НФ)
  - 8.5.1.3. Третья нормальная форма (3НФ)
  - 8.5.1.4. Нормальная форма Бойса — Кодда (НФ)
  - 8.5.1.5. Четвертая нормальная форма (4НФ)
  - 8.5.1.6. Пятая нормальная форма (5НФ)
- 8.5.2. Правила Кодда
  - 8.5.2.1. Правило 1: Информационное правило
  - 8.5.2.2. Правило 2: Гарантированный доступ к данным
  - 8.5.2.3. Правило 3: Систематическая поддержка отсутствующих значений
  - 8.5.2.4. Правило 4: Описание базы данных
  - 8.5.2.5. Правило 5: Полнота подмножества языка
  - 8.5.2.6. Правило 6: Изменение представлений
  - 8.5.2.7. Правило 7. Вставка и обновление
  - 8.5.2.8. Правило 8: Физическая независимость данных
  - 8.5.2.9. Правило 9: Логическая независимость данных
  - 8.5.2.10. Правило 10: Независимость контроля целостности
    - 8.5.2.10.1. Правила целостности
  - 8.5.2.11. Правило 11: Независимость от расположения
  - 8.5.2.12. Правило 12: Согласование языковых уровней
- 8.5.3. Наглядный пример

8.6. Хранилище данных/OLAP-система
- 8.6.1. Хранилище данных
- 8.6.2. Таблица фактов
- 8.6.3. Таблица размеров
- 8.6.4. Создание системы OLAP. Инструменты

8.7. Производительность базы данных (БД)
- 8.7.1. Оптимизация индекса
- 8.7.2. Оптимизация запросов
- 8.7.3. Разбиение таблиц

8.8. Моделирование реального проекта для проектирования БД (I)
- 8.8.1. Обзор проекта (Компания A)
- 8.8.2. Применение проектирования баз данных
- 8.8.3. Предлагаемые тесты
- 8.8.4. Предлагаемые тесты. *Обратная связь*

8.9. Моделирование реального проекта для проектирования БД (II)
- 8.9.1. Обзор проекта (Компания B)
- 8.9.2. Применение проектирования баз данных
- 8.9.3. Предлагаемые тесты
- 8.9.4. Предлагаемые тесты. *Обратная связь*

8.10. Актуальность оптимизации БД для качества программного обеспечения
- 8.10.1. Оптимизация конструкции
- 8.10.2. Оптимизация кода запросов
- 8.10.3. Оптимизация кода хранимой процедуры
- 8.10.4. Влияние *триггеров* на качество программного обеспечения. Рекомендации по применению

Модуль 9. Проектирование масштабируемых архитектур. Архитектура в жизненном цикле программного обеспечения

9.1. Проектирование масштабируемых архитектур (I)
- 9.1.1. Масштабируемые архитектуры
- 9.1.2. Принципы масштабируемой архитектуры
  - 9.1.2.1. Надежность
  - 9.1.2.2. Масштабируемость
  - 9.1.2.3. Поддержка
- 9.1.3. Типы масштабируемости
  - 9.1.3.1. Вертикальная
  - 9.1.3.2. Горизонтальная
  - 9.1.3.3. Комбинированная

9.2. Предметно-ориентированное проектирование DDD (*Domain-Driven Design*)
- 9.2.1. Модель DDD. Ориентация на домен
- 9.2.2. Слои, распределение ответственности и шаблоны проектирования
- 9.2.3. Разделение как основа для качества

9.3. Проектирование масштабируемых архитектур (II). Преимущества, ограничения и стратегии проектирования
- 9.3.1. Масштабируемая архитектура. Преимущества
- 9.3.2. Масштабируемая архитектура. Ограничения
- 9.3.3. Стратегии разработки масштабируемых архитектур (описательная таблица)

9.4. Жизненный цикл программного обеспечения (I). Этапы
- 9.4.1. Жизненный цикл программного обеспечения
  - 9.4.1.1. Этап планирования
  - 9.4.1.2. Этап анализа
  - 9.4.1.3. Этап проектирования
  - 9.4.1.4. Этап реализации
  - 9.4.1.5. Этап тестирования
  - 9.4.1.6. Этап установки/развертывания
  - 9.4.1.7. Этап использования и обслуживания

9.5. Модели жизненного цикла программного обеспечения
- 9.5.1. Каскадная модель
- 9.5.2. Повторяющаяся модель
- 9.5.3. Спиральная модель
- 9.5.4. Модель *Большого взрыва*

9.6. Жизненный цикл программного обеспечения (II). Автоматизация
- 9.6.1. Жизненный цикл разработки программного обеспечения. Решение
  - 9.6.1.1. Непрерывная интеграция и разработка (CI/CD)
  - 9.6.1.2. Agile-методологии
  - 9.6.1.3. DevOps / производственные операции
- 9.6.2. Будущие тенденции
- 9.6.3. Практические примеры

9.7. Архитектура программного обеспечения в жизненном цикле программного обеспечения
- 9.7.1. Преимущества
- 9.7.2. Ограничения
- 9.7.3. Инструменты

9.8. Моделирование реального проекта для проектирования архитектуры программного обеспечения (I)
- 9.8.1. Обзор проекта (Компания A)
- 9.8.2. Применение проектирования архитектуры программного обеспечения
- 9.8.3. Предлагаемые тесты
- 9.8.4. Предлагаемые тесты. *Обратная связь*

9.9. Моделирование реального проекта для проектирования архитектуры программного обеспечения (II)
- 9.9.1. Обзор проекта (Компания B)
- 9.9.2. Применение проектирования архитектуры программного обеспечения
- 9.9.3. Предлагаемые тесты
- 9.9.4. Предлагаемые тесты. *Обратная связь*

9.10. Моделирование реального проекта для проектирования архитектуры программного обеспечения (III)
- 9.10.1. Обзор проекта (Компания C)
- 9.10.2. Применение проектирования архитектуры программного обеспечения
- 9.10.3. Предлагаемые тесты
- 9.10.4. Предлагаемые тесты. *Обратная связь*

Модуль 10. Критерии качества ISO, IEC 9126. Метрики качества программного обеспечения

10.1. Критерии качества. Стандарт ISO, IEC 9126
- 10.1.1. Критерии качества
- 10.1.2. Качество программного обеспечения. Обоснование. Стандарт ISO, IEC 9126
- 10.1.3. Измерение качества программного обеспечения как ключевой показатель

10.2. Критерии качества программного обеспечения. Характеристики
- 10.2.1. Надежность
- 10.2.2. Функциональность
- 10.2.3. Эффективность
- 10.2.4. Юзабилити
- 10.2.5. Управляемость
- 10.2.6. Портативность
- 10.2.7. Безопасность

10.3. Стандарт ISO, IEC 9126(I) Презентация
- 10.3.1. Описание нарушений Стандарт ISO, IEC 9126
- 10.3.2. Функциональность
- 10.3.3. Надежность
- 10.3.4. Юзабилити
- 10.3.5. Управляемость
- 10.3.6. Портативность
- 10.3.7. Качество использования
- 10.3.8. Метрики качества программного обеспечения
- 10.3.9. Метрические данные качества в ISO 9126

10.4. Стандарт ISO, IEC 9126 (II). Модели МакКолла и Боэма
- 10.4.1. Модель МакКолла: факторы качества
- 10.4.2. Модель Боэма
- 10.4.3. Промежуточный уровень. Характеристики

10.5. Метрики качества программного обеспечения (I). Элементы
- 10.5.1. Измерения
- 10.5.2. Метрические данные
- 10.5.3. Показатели
  - 10.5.3.1. Типы показателей
- 10.5.4. Размеры и модели
- 10.5.5. Сфера применения метрики программного обеспечения
- 10.5.6. Классификация метрик программного обеспечения

10.6. Измерение качества программного обеспечения (II). Практика измерений
- 10.6.1. Сбор метрических данных
- 10.6.2. Измерение внутренних атрибутов продукта
- 10.6.3. Измерение внешних атрибутов продукта
- 10.6.4. Измерение ресурсов
- 10.6.5. Метрики для объектно-ориентированных систем

10.7. Разработка единого показателя качества программного обеспечения
- 10.7.1. Отдельный показатель как глобальный показатель
- 10.7.2. Разработка, обоснование и применение индикаторов
- 10.7.3. Примеры применения. Необходимость знать детали

10.8. Моделирование реального проекта для измерения качества (I)
- 10.8.1. Обзор проекта (Компания A)
- 10.8.2. Применение измерения качества
- 10.8.3. Предлагаемые тесты
- 10.8.4. Предлагаемые тесты. *Обратная связь*

10.9. Моделирование реального проекта для измерения качества (II)
- 10.9.1. Обзор проекта (Компания B)
- 10.9.2. Применение измерения качества
- 10.9.3. Предлагаемые тесты
- 10.9.4. Предлагаемые тесты. *Обратная связь*

10.10. Моделирование реального проекта для измерения качества (III)
- 10.10.1. Обзор проекта (Компания C)
- 10.10.2. Применение измерения качества
- 10.10.3. Предлагаемые тесты
- 10.10.4. Предлагаемые тесты. *Обратная связь*

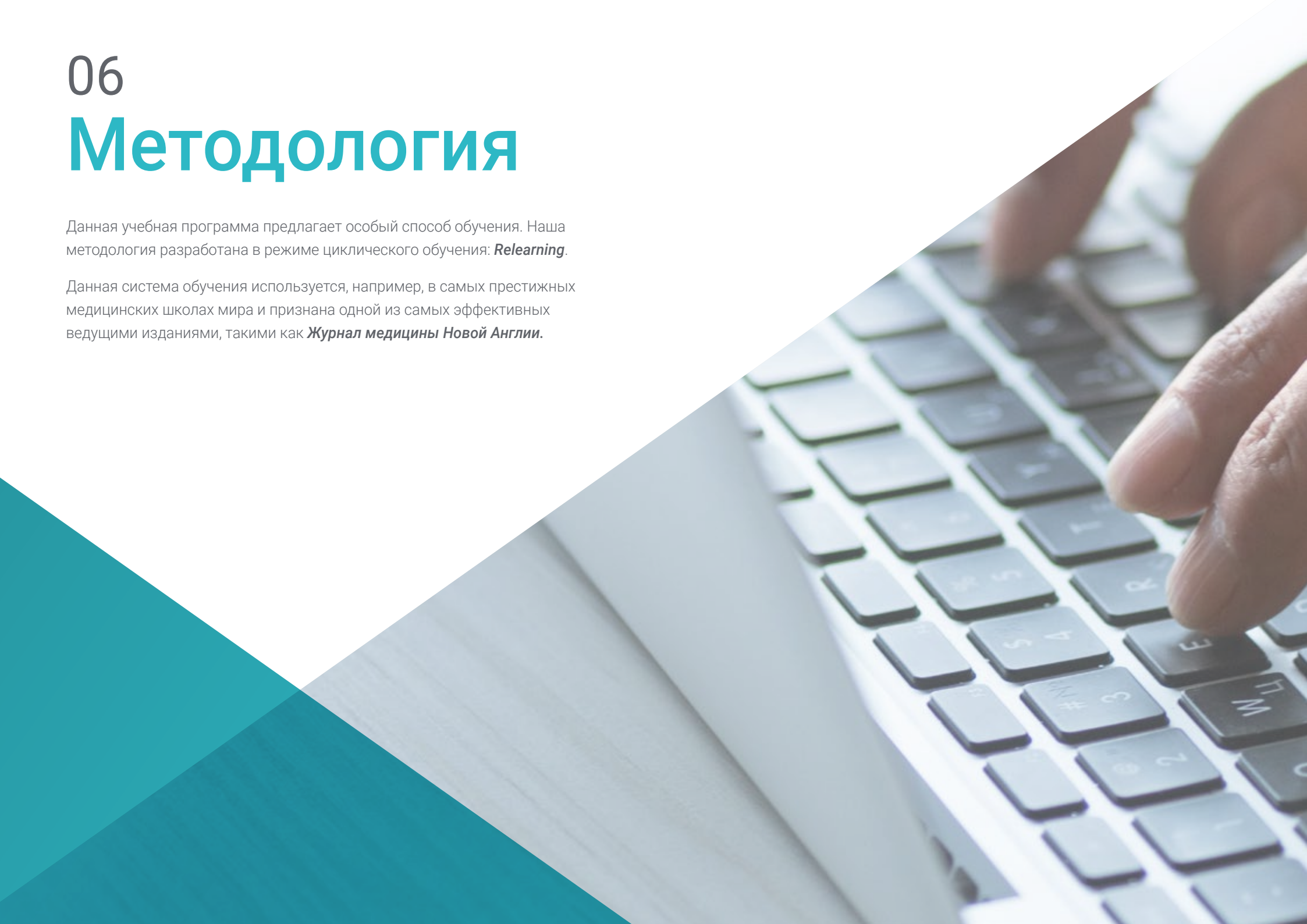
Вы получите доступ к уникальному и специализированному содержанию. Выбранному преподавателями-экспертами для прохождения программы, которая сделает ваш профессиональный профиль более значительным"

06

Методология

Данная учебная программа предлагает особый способ обучения. Наша методология разработана в режиме циклического обучения: ***Relearning***.

Данная система обучения используется, например, в самых престижных медицинских школах мира и признана одной из самых эффективных ведущими изданиями, такими как ***Журнал медицины Новой Англии.***



Откройте для себя методику Relearning, которая отвергает традиционное линейное обучение, чтобы показать вам циклические системы обучения: способ, который доказал свою огромную эффективность, особенно в предметах, требующих запоминания"

Исследование кейсов для контекстуализации всего содержания

Наша программа предлагает революционный метод развития навыков и знаний. Наша цель - укрепить компетенции в условиях меняющейся среды, конкуренции и высоких требований.

С TECH вы сможете познакомиться со способом обучения, который опровергает основы традиционных методов образования в университетах по всему миру”

Вы получите доступ к системе обучения, основанной на повторении, с естественным и прогрессивным обучением по всему учебному плану.

В ходе совместной деятельности и рассмотрения реальных кейсов студент научится разрешать сложные ситуации в реальной бизнес-среде.

Инновационный и отличный от других метод обучения

Эта программа TECH - интенсивная программа обучения, созданная с нуля, которая предлагает самые сложные задачи и решения в этой области на международном уровне. Благодаря этой методологии ускоряется личностный и профессиональный рост, делая решающий шаг на пути к успеху. Метод кейсов, составляющий основу данного содержания, обеспечивает следование самым современным экономическим, социальным и профессиональным реалиям.

> *Наша программа готовит вас к решению новых задач в условиях неопределенности и достижению успеха в карьере"*

Кейс-метод является наиболее широко используемой системой обучения лучшими преподавателями в мире. Разработанный в 1912 году для того, чтобы студенты-юристы могли изучать право не только на основе теоретического содержания, метод кейсов заключается в том, что им представляются реальные сложные ситуации для принятия обоснованных решений и ценностных суждений о том, как их разрешить. В 1924 году он был установлен в качестве стандартного метода обучения в Гарвардском университете.

Что должен делать профессионал в определенной ситуации? Именно с этим вопросом мы сталкиваемся при использовании кейс-метода - метода обучения, ориентированного на действие. На протяжении всей курса студенты будут сталкиваться с многочисленными реальными случаями из жизни. Им придется интегрировать все свои знания, исследовать, аргументировать и защищать свои идеи и решения.

Методология *Relearning*

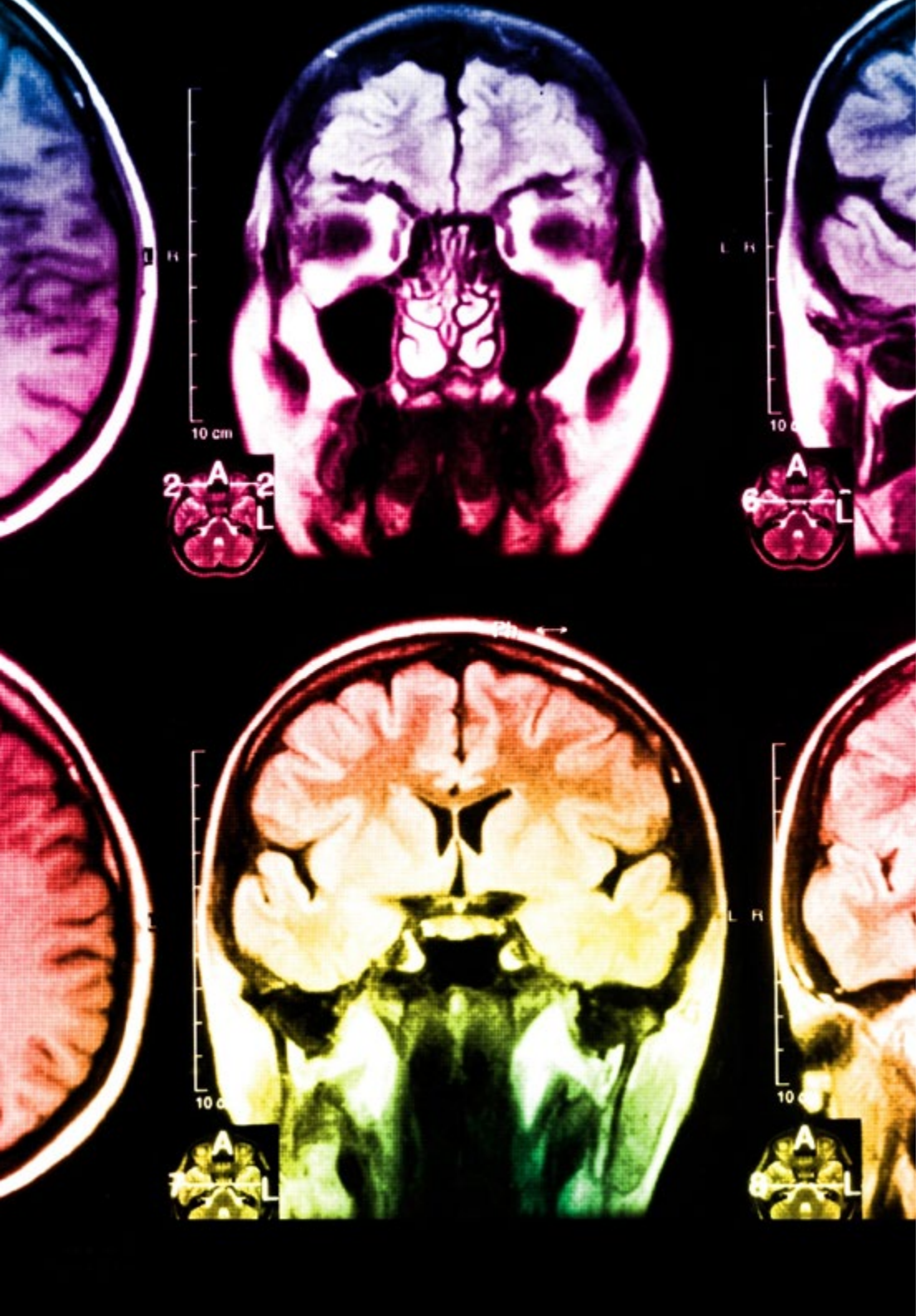
TECH эффективно объединяет метод кейсов с системой 100% онлайн-обучения, основанной на повторении, которая сочетает различные дидактические элементы в каждом уроке.

Мы улучшаем метод кейсов с помощью лучшего метода 100% онлайн-обучения: *Relearning*.

В 2019 году мы достигли лучших результатов обучения среди всех онлайн-университетов в мире.

В TECH вы будете учиться по передовой методике, разработанной для подготовки руководителей будущего. Этот метод, играющий ведущую роль в мировой педагогике, называется *Relearning*.

Наш университет - единственный вуз, имеющий лицензию на использование этого успешного метода. В 2019 году нам удалось повысить общий уровень удовлетворенности наших студентов (качество преподавания, качество материалов, структура курса, цели...) по отношению к показателям лучшего онлайн-университета.



В нашей программе обучение не является линейным процессом, а происходит по спирали (мы учимся, разучиваемся, забываем и заново учимся). Поэтому мы дополняем каждый из этих элементов по концентрическому принципу. Благодаря этой методике более 650 000 выпускников университетов добились беспрецедентного успеха в таких разных областях, как биохимия, генетика, хирургия, международное право, управленческие навыки, спортивная наука, философия, право, инженерное дело, журналистика, история, финансовые рынки и инструменты. Наша методология преподавания разработана в среде с высокими требованиями к уровню подготовки, с университетским контингентом студентов с высоким социально-экономическим уровнем и средним возрастом 43,5 года.

Методика Relearning позволит вам учиться с меньшими усилиями и большей эффективностью, все больше вовлекая вас в процесс обучения, развивая критическое мышление, отстаивая аргументы и противопоставляя мнения, что непосредственно приведет к успеху.

Согласно последним научным данным в области нейронауки, мы не только знаем, как организовать информацию, идеи, образы и воспоминания, но и знаем, что место и контекст, в котором мы что-то узнали, имеют фундаментальное значение для нашей способности запомнить это и сохранить в гиппокампе, чтобы удержать в долгосрочной памяти.

Таким образом, в рамках так называемого нейрокогнитивного контекстно-зависимого электронного обучения, различные элементы нашей программы связаны с контекстом, в котором участник развивает свою профессиональную практику.

В рамках этой программы вы получаете доступ к лучшим учебным материалам, подготовленным специально для вас:

Учебный материал

Все дидактические материалы создаются преподавателями специально для студентов этого курса, чтобы они были действительно четко сформулированными и полезными.

Затем вся информация переводится в аудиовизуальный формат, создавая дистанционный рабочий метод TECH. Все это осуществляется с применением новейших технологий, обеспечивающих высокое качество каждого из представленных материалов.

Мастер-классы

Существуют научные данные о пользе экспертного наблюдения третьей стороны.

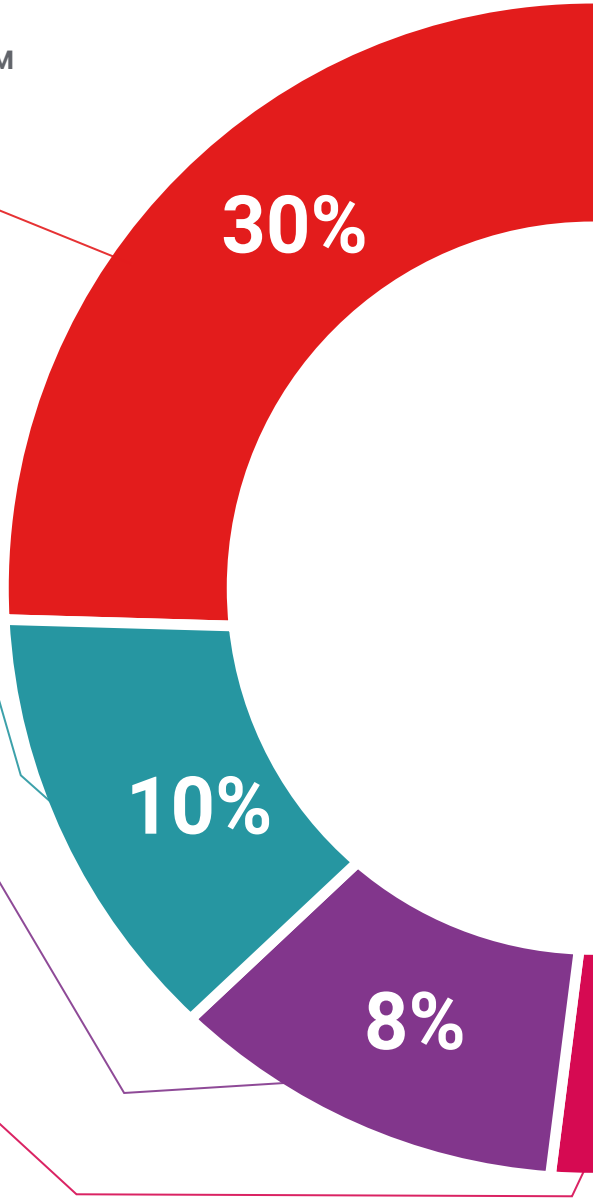
Так называемый метод обучения у эксперта укрепляет знания и память, а также формирует уверенность в наших будущих сложных решениях.

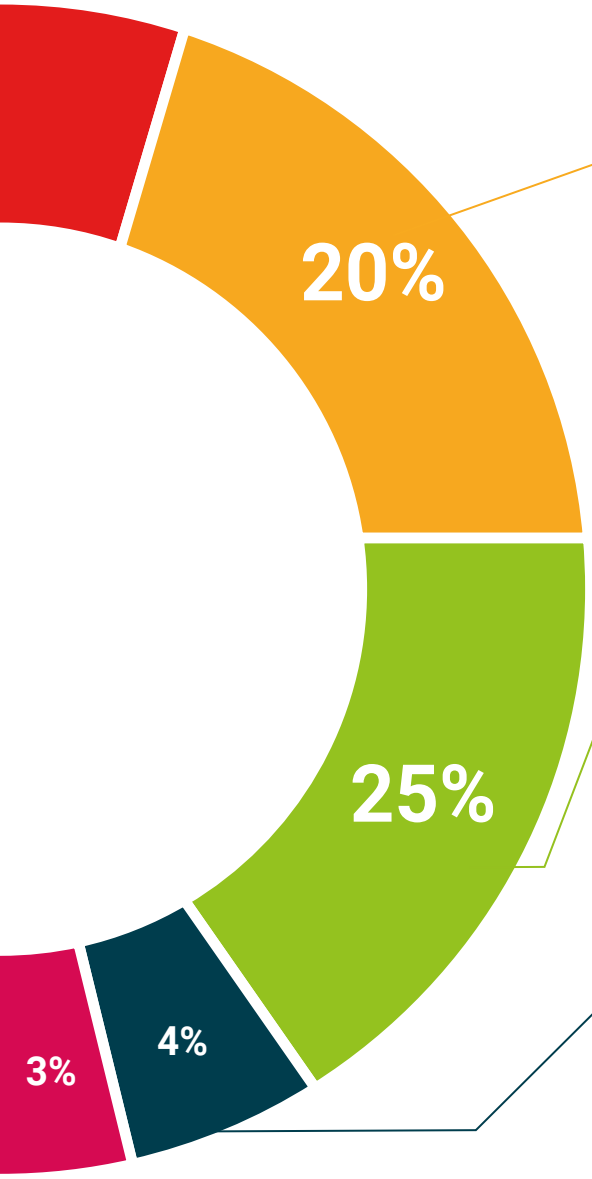
Практика навыков и компетенций

Студенты будут осуществлять деятельность по развитию конкретных компетенций и навыков в каждой предметной области. Практика и динамика приобретения и развития навыков и способностей, необходимых специалисту в рамках глобализации, в которой мы живем.

Дополнительная литература

Новейшие статьи, консенсусные документы и международные руководства включены в список литературы курса. В виртуальной библиотеке TECH студент будет иметь доступ ко всем материалам, необходимым для завершения обучения.





Метод кейсов

Метод дополнится подборкой лучших кейсов, выбранных специально для этой квалификации. Кейсы представляются, анализируются и преподаются лучшими специалистами на международной арене.

Интерактивные конспекты

Мы представляем содержание в привлекательной и динамичной мультимедийной форме, которая включает аудио, видео, изображения, диаграммы и концептуальные карты для закрепления знаний.

Эта уникальная обучающая система для представления мультимедийного содержания была отмечена компанией Microsoft как "Европейская история успеха".

Тестирование и повторное тестирование

На протяжении всей программы мы периодически оцениваем и переоцениваем ваши знания с помощью оценочных и самооценочных упражнений: так вы сможете убедиться, что достигаете поставленных целей.

07

Квалификация

Специализированная магистратура в области Качество программного обеспечения гарантирует, помимо самого строгого и современного обучения, получение диплома об окончании Специализированной магистратуры, выдаваемого TECH Технологическим университетом.

Успешно пройдите эту программу и получите университетский диплом без хлопот, связанных с поездками и оформлением документов"

Данная **Специализированная магистратура в области качества программного обеспечения** содержит самую полную и современную программу на рынке.

После прохождения аттестации студент получит по почте* с подтверждением получения соответствующий диплом **Специализированной магистратуры,** выданный **ТЕСН Технологическим университетом.**

Диплом, выданный **ТЕСН Технологическим университетом,** подтверждает квалификацию, полученную в Специализированной магистратуре, и соответствует требованиям, обычно предъявляемым биржами труда, конкурсными экзаменами и комитетами по оценке карьеры.

Диплом: **Специализированная магистратура в области качества программного обеспечения**

Количество учебных часов: **1500 часов**


tech технологический университет

Настоящий

ДИПЛОМ

свидетельствует о том, что

Г-н/Г-жа _____, с документом, удостоверяющий личность_____

Освоил(а) и успешно прошел(ла) аккредитацию программы

СПЕЦИАЛИЗИРОВАННАЯ МАГИСТРАТУРА

в области

Качество программного обеспечения

Данный диплом специализированной программы, присуждаемый Университетом, соответствует 1500 учебным часам, с датой начала дд/мм/гггг и датой окончания дд/мм/гггг.

ТЕСН является частным высшим учебным заведением, признанным Министерством народного образования Мексики с 28 июня 2018 года.

17 июня 2020 г.

Д-р Tere Guevara Navarro
Ректор

unique TECH code: AFWOR23S techtitute.com/certificates


Специализированная магистратура в области Качество программного обеспечения

Общее распределение учебного плана

Тип предмета	Часы
Обязательный (Обязат.)	1.500
Необязательный (Необязат.)	0
Внешние стажировки (ВС)	0
Магистерская диссертация (МД)	0
Всего	1.500

Общее распределение учебного плана

Курс	Предмет	Часы	Характер
1°	Качество программного обеспечения. Уровни развития TRL	150	Обязат
1°	Проектирование программного обеспечения. Функциональная и техническая документация	150	Обязат
1°	Тестирование программного обеспечения. Автоматизация тестирования	150	Обязат
1°	Методологии управления проектами программного обеспечения. Каскадная модель vs agile-методологии	150	Обязат
1°	TDD (Test Driven Developement). Разработка программного обеспечеия через тестирование	150	Обязат
1°	DevOps Управление качеством программного обеспечения	150	Обязат
1°	DevOps и непрерывная интеграция. Передовые практические решения в разработке программного обеспечения	150	Обязат
1°	Проектирование баз данных (БД). Стандартизация и производительность. Качество программного обеспечения	150	Обязат
1°	Проектирование масштабируемых архитектур. Архитектура в жизненном цикле программного обеспечения	150	Обязат
1°	Критерии качества ISO, IEC 9126. Метрики качества программного обеспечения	150	Обязат

Д-р Tere Guevara Navarro
Ректор

tech технологический университет

*Гаагский апостиль. В случае, если студент потребует, чтобы на его диплом в бумажном формате был проставлен Гаагский апостиль, TECH EDUCATION предпримет необходимые шаги для его получения за дополнительную плату.

tech технологический университет

Специализированная магистратура

Качество программного обеспечения

- Формат: **онлайн**
- Продолжительность: **12 месяцев**
- Учебное заведение: **TECH Технологический университет**
- Режим обучения: **16ч./неделя**
- Расписание: **по своему усмотрению**
- Экзамены: **онлайн**

Специализированная магистратура
Качество программного обеспечения

tech технологический университет