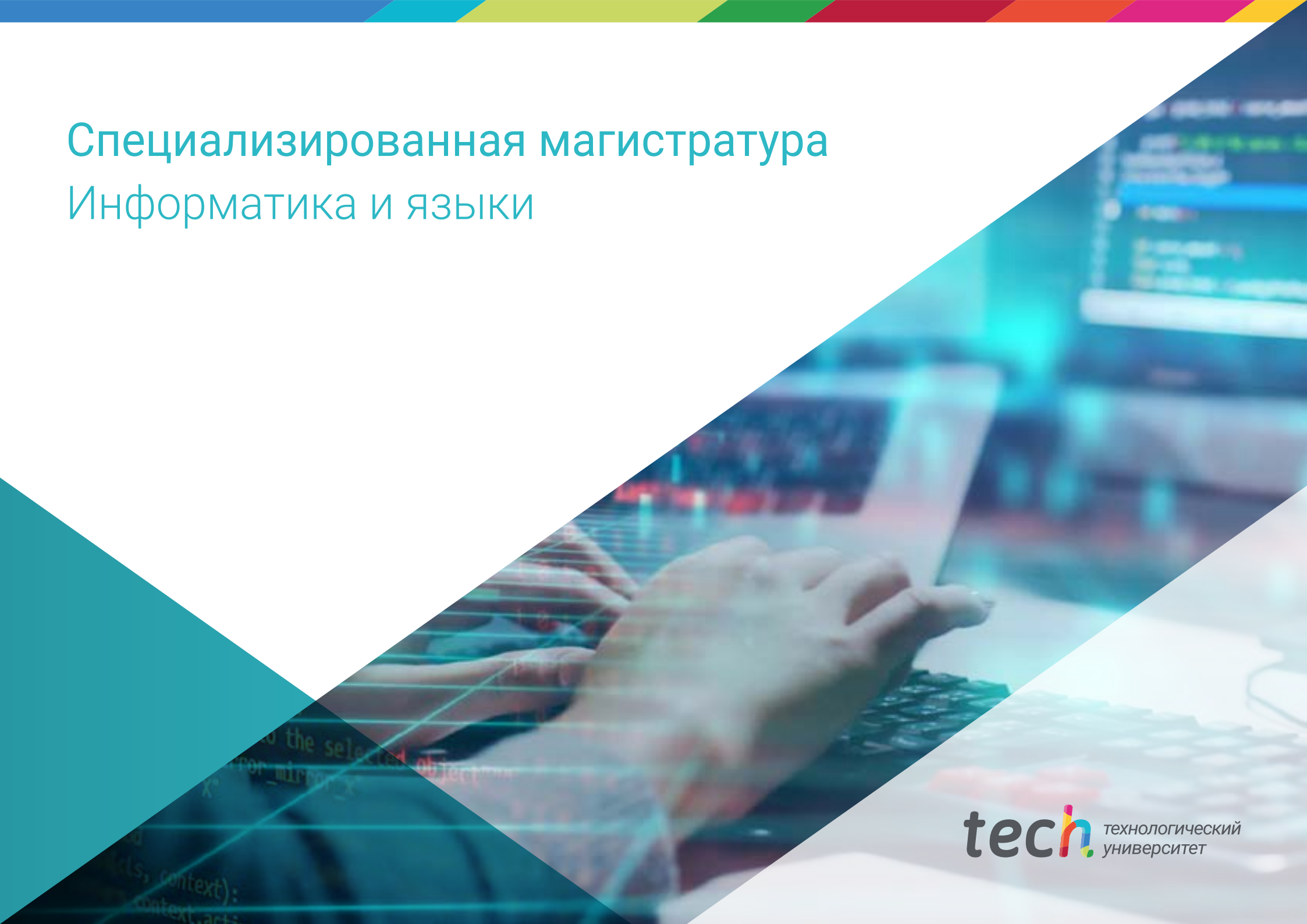
Специализированная магистратура

Информатика и языки



tech технологический университет

tech технологический университет

Специализированная магистратура

Информатика и языки

- Формат: **онлайн**
- Продолжительность: **12 месяцев**
- Учебное заведение: **TECH Технологический университет**
- Расписание: **по своему усмотрению**
- Экзамены: **онлайн**

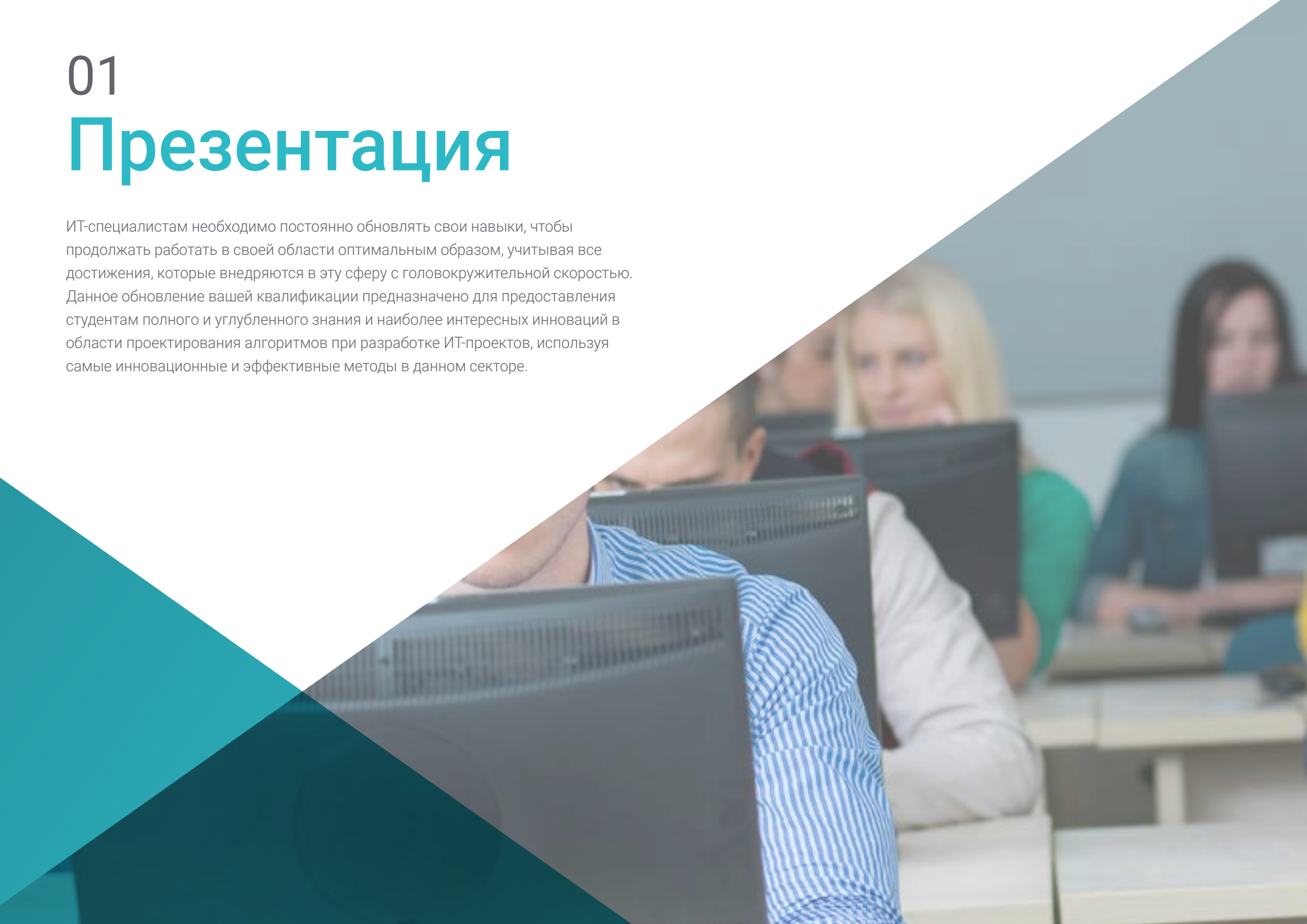
Веб-доступ: www.techtitute.com/ru/information-technology/professional-master-degree/master-computing-programming-languages

Оглавление

01

Презентация

ИТ-специалистам необходимо постоянно обновлять свои навыки, чтобы продолжать работать в своей области оптимальным образом, учитывая все достижения, которые внедряются в эту сферу с головокружительной скоростью. Данное обновление вашей квалификации предназначено для предоставления студентам полного и углубленного знания и наиболее интересных инноваций в области проектирования алгоритмов при разработке ИТ-проектов, используя самые инновационные и эффективные методы в данном секторе.



Приобретите фундаментальные знания о вычислительной технике и о том, как успешно применять их при разработке ИТ-проектов, в Специализированной магистратуре высокого качества"

Данная программа посвящена основам программирования и структуры данных, алгоритмике и сложности, а также продвинутому проектированию алгоритмов, продвинутому программированию или языковым процессорам и компьютерной графике, среди прочих аспектов, связанных с этой областью информатики.

Данная программа предоставляет студентам специальные инструменты и навыки для успешного развития профессиональной деятельности в широкой среде информатики и языков. Во время обучения профессионалы будут работать над такими ключевыми компетенциями, как знание реальности и повседневной практики в различных областях ИТ, а также развивать ответственность при контроле и надзоре за работой, а также конкретные навыки в этой области.

Более того, поскольку это 100% онлайн-программа, студент не обусловлен фиксированным расписанием или необходимостью переезда в другое физическое место, а может получить доступ к материалам в любое время суток, совмещая свою работу или личную жизнь с учебой.

Команда преподавателей этой программы по информатике и языкам провела тщательный отбор каждой из тем данной Специализированной магистратуры, чтобы предложить студенту наиболее полную возможность обучения и всегда связанную с последними данными.

Данная **Специализированная магистратура в области информатики и языки** содержит наиболее полную и современную программу на рынке. Основными особенностями обучения являются:

- Разработка тематических исследований, представленных экспертами в области информатики и языков
- Наглядное, схематичное и исключительно практичное содержание курса предоставляет научную и практическую информацию по тем дисциплинам, которые необходимы для профессиональной практики
- Практические упражнения для самопроверки, контроля и улучшения успеваемости
- Особое внимание уделяется инновационным методологиям в области информатики и языков
- Теоретические занятия, вопросы эксперту, дискуссионные форумы по спорным темам и самостоятельная работа
- Учебные материалы курса доступны с любого стационарного или мобильного устройства с выходом в интернет

Исключительная возможность в удобной и простой форме изучить математические и базовые процессы и знания, необходимые для качественного компьютерного программирования"

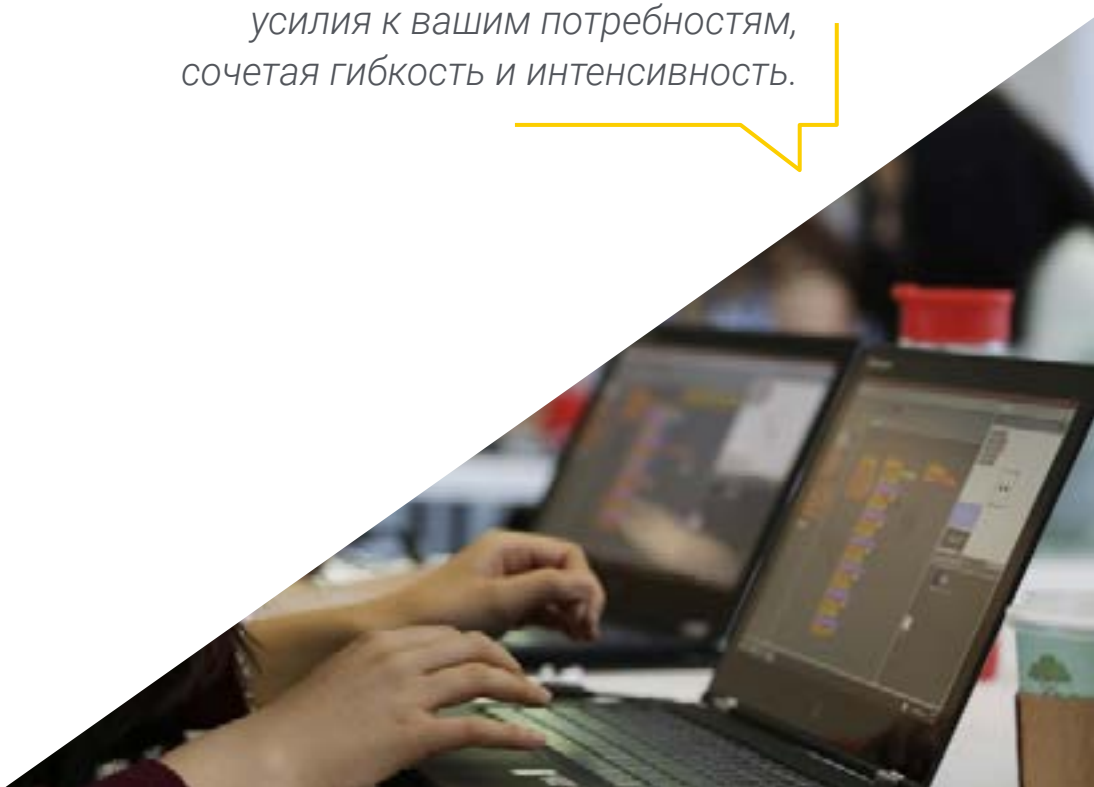
Специализированная магистратура, которая основывает свою эффективность на наиболее высоко ценимых образовательных технологиях на рынке, с аудиовизуальными и учебными системами, которые позволят вам учиться быстрее и комфортнее"

Мультимедийное содержание, разработанное с использованием новейших образовательных технологий, позволит специалисту проходить обучение с учетом ситуации и контекста, т.е. в такой среде, которая обеспечит погружение в 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 года. В этом специалисту будет помогать инновационная интерактивная видеосистема, разработанная известными и опытными специалистами в области информатики и языков.

У вас будет широкий и понятный дидактический материал, включающий все актуальные темы, представляющие интерес для специалиста, желающего продвинуться в области информатики и языков.

Обучение с высоким образовательным эффектом, которое позволит вам адаптировать усилия к вашим потребностям, сочетая гибкость и интенсивность.



02

Цели

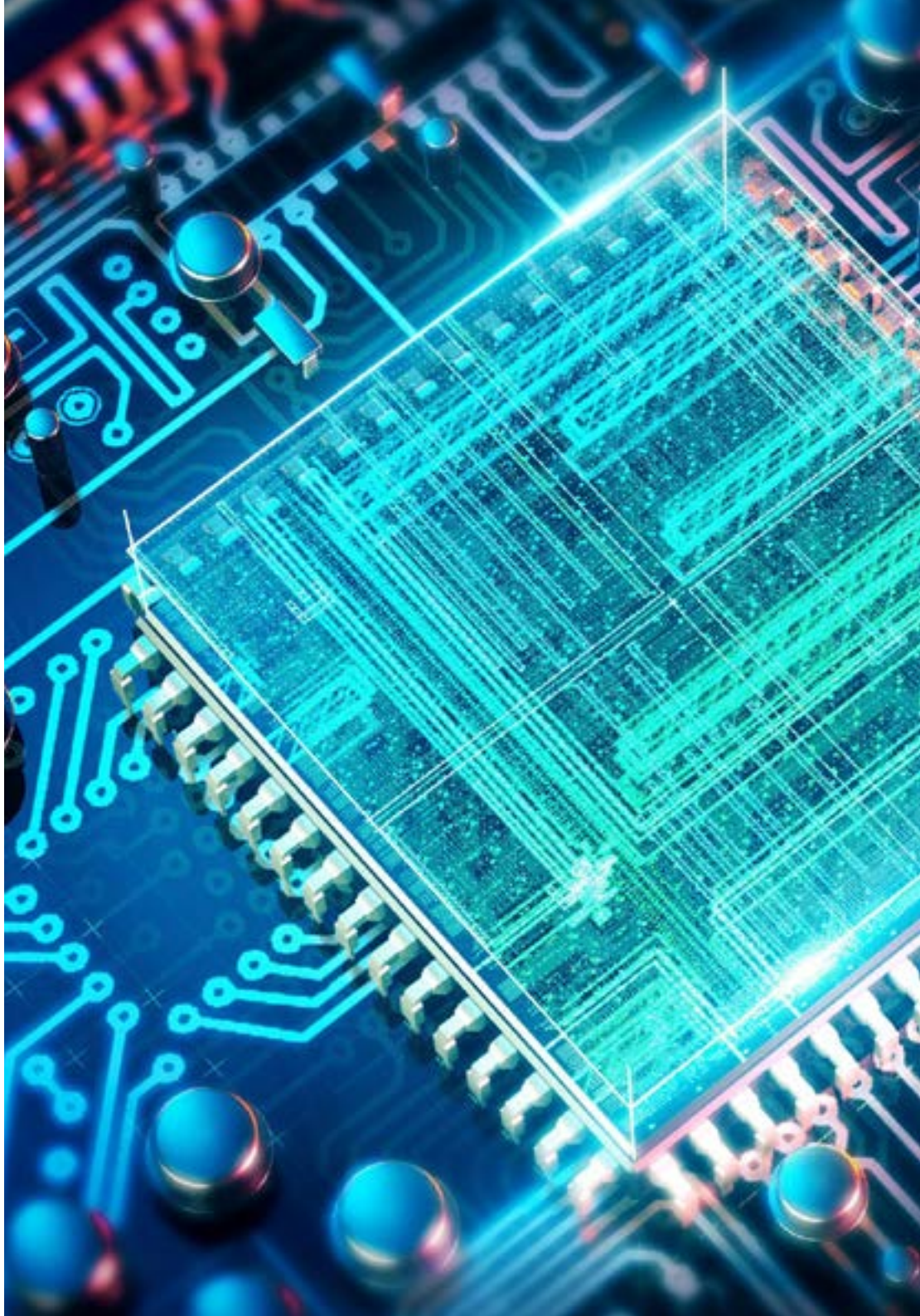
Программа "Информатика и языки" была создана специально для профессионалов, которые стремятся быстро и качественно продвинуться в этой области, организуя ее на основе реалистичных и высокоценных целей, которые позволят им перейти на новый уровень работы в этой сфере.

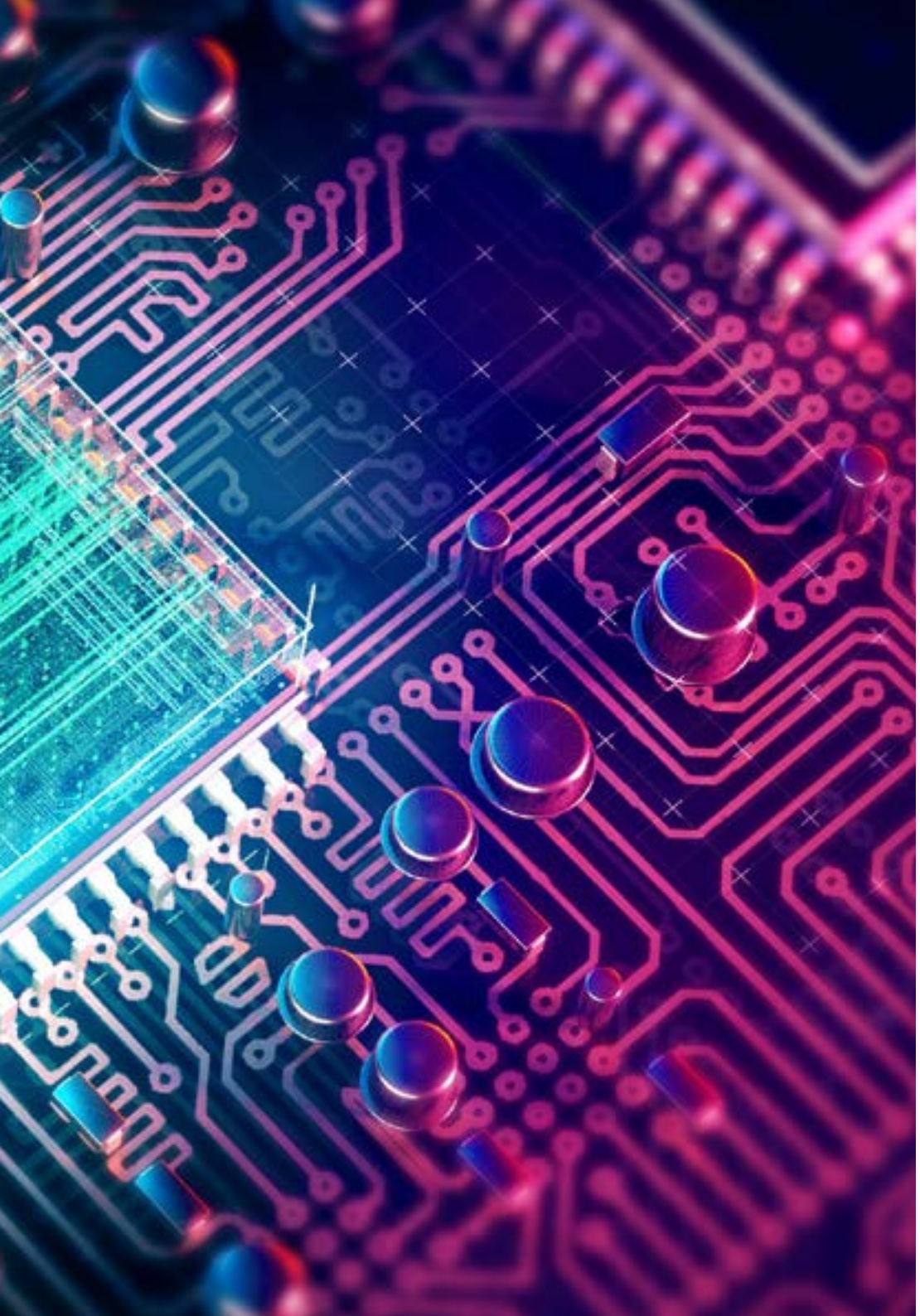
Наша цель - предоставить специалистам в области информатики высококачественное обновление знаний, которое позволит им компетентно работать в области информатики и языков"

Общая цель

- Обучать научно и технологически, а также готовить к профессиональной практике в области информатики и языков, все это с помощью всеобъемлющей и разносторонней подготовки, адаптированной к новым технологиям и инновациям в этой области

Воспользуйтесь этой возможностью и сделайте решающий шаг, чтобы быть в курсе последних достижений в области информатики и языков"





Конкретные цели

Модуль 1. Основы программирования

- Понимать базовую структуру компьютера, программное обеспечение и языки программирования общего назначения
- Научиться разрабатывать и интерпретировать алгоритмы, которые являются необходимой основой для разработки программного обеспечения
- Понимать основные элементы компьютерной программы, такие как различные типы данных, операторы, выражения, утверждения, операторы ввода-вывода и управления
- Понимание различных структур данных, доступных в языках программирования общего назначения, как статических, так и динамических, и приобретение необходимых знаний по работе с файлами
- Понять различные методы тестирования программного обеспечения и важность создания хорошей документации вместе с хорошим исходным кодом
- Изучить основы языка программирования C++, одного из самых распространенных языков программирования в мире

Модуль 2. Структура данных

- Изучить основы программирования на C++, включая классы, переменные, условные выражения и объекты
- Понимать абстрактные типы данных, линейные типы структур данных, простые и сложные иерархические структуры данных и их реализацию на C++
- Понимать функционирование продвинутых структур данных, отличных от обычных
- Понимать теорию и практику, связанные с использованием приоритетных насыпей и приоритетных очередей

- Узнать, как работают таблицы Hash в качестве абстрактных типов данных и функций
- Понять теорию графов, а также продвинутые алгоритмы и концепции графов

Модуль 3. Алгоритм и сложность

- Изучить основные стратегии проектирования алгоритмов, а также различные методы и меры для вычисления алгоритмов
- Знать основные алгоритмы сортировки, используемые при разработке программного обеспечения
- Понять, как различные алгоритмы работают с деревьями, *Heaps* и графами
- Понять, как работают *жадные* алгоритмы, их стратегию и примеры их использования в основных известных проблемах
- Знать также применение *жадных* алгоритмов на графах
- Изучить основные стратегии поиска минимального пути, с приближением существенных проблем данной области и алгоритмов их решения
- Понять технику *Backtracking* и ее основные применения, а также альтернативные техники

Модуль 4. Продвинутое проектирование алгоритмов

- Углубитесь в передовое проектирование алгоритмов, анализируя рекурсивные алгоритмы и алгоритмы "разделяй и властвуй", а также выполняя амортизированный анализ
- Понять концепции динамического программирования и алгоритмы для задач NP
- Понимать, как работает комбинаторная оптимизация, а также различные алгоритмы рандомизации и параллельные алгоритмы
- Знать и понимать, как работают различные методы локального поиска и с кандидатами
- Изучить механизмы формальной проверки программ и итеративной проверки программ, включая логику первого порядка и формальную систему Хоара
- Изучить работу некоторых основных численных методов, таких как метод бисекции, метод Ньютона-Рафсона и метод секущих

Модуль 5. Расширенное программирование

- Углубить знания по программированию, особенно в отношении объектно-ориентированного программирования, и различных типов отношений между существующими классами
- Знать различные шаблоны проектирования для решения объектно-ориентированных задач
- Изучить событийно-управляемое программирование и разработку пользовательского интерфейса с помощью Qt
- Приобрести основные знания о параллельном программировании, процессах и потоках
- Научиться управлять использованием потоков и синхронизации, а также решать общие проблемы в рамках параллельного программирования
- Понять важность документации и тестирования при разработке программного обеспечения

Модуль 6. Теоретическая информатика

- Понять основные теоретические математические концепции, лежащие в основе информатики, такие как пропозициональная логика, теория множеств, исчисляемые и неисчисляемые множества
- Понять концепции формальных языков и грамматик, а также машин Тьюринга в их различных вариантах
- Узнать о различных типах неопределимых и неподдающихся решению проблем, включая различные варианты этих проблем и подходы к их решению

- Понять функционирование различных видов языков, основанных на рандомизации, и других видов классов и грамматик
- Узнать о других передовых вычислительных системах, таких как мембранные вычисления, ДНК-вычисления и квантовые вычисления

Модуль 7. Теория автоматов и формальных языков

- Понять теорию автоматов и формальных языков, изучить понятия алфавитов, строк и языков, а также научиться проводить формальные демонстрации
- Углубить понимание различных типов конечных автоматов, как детерминированных, так и недетерминированных
- Изучить основные и расширенные понятия, связанные с регулярными языками и регулярными выражениями, а также применение леммы накачки и закрытие регулярных языков
- Понимать контекстно-независимые грамматики, а также работу стековых автоматов
- Углубить нормальные формы, лемму накачки контекстно-независимых грамматик и свойства контекстно-независимых языков

Модуль 8. Языковые процессоры

- Ввести понятия, связанные с процессом компиляции и различными видами анализа: лексическим, синтаксическим и семантическим
- Знать, как работает лексический анализатор, его применение и устранение ошибок
- Углубить знания в области синтаксического анализа, как нисходящего, так и восходящего, но с особым акцентом на различные типы нисходящих синтаксических анализаторов
- Понять, как работают семантические синтаксические анализаторы, традиция синтаксиса, таблица символов и различные типы
- Изучить различные механизмы генерации кода, как в среде выполнения, так и для генерации промежуточного кода
- Заложить основы оптимизации кода, включая переупорядочивание выражений и оптимизацию циклов

Модуль 9. Компьютерная графика и визуализация

- Ввести основные понятия компьютерной графики и компьютерной визуализации, такие как теория цвета и ее модели, а также свойства света
- Понимать функционирование примитивов вывода и их алгоритмы, как для рисования линий, так и для рисования окружностей и заливок
- Углубленно изучить различные 2D и 3D преобразования и их системы координат, а также компьютерную визуализацию
- Научиться делать 3D-проекции и разрезы, а также удалять скрытые поверхности
- Изучить теорию, связанную с интерполяцией и параметрическими кривыми, а также кривыми Безье и B-сплайнами

Модуль 10. Биоинспирированные вычисления

- Ввести понятие биоинспирированных вычислений, а также понять функционирование различных типов алгоритмов социальной адаптации и генетических алгоритмов
- Углубить изучение различных моделей эволюционных вычислений, зная их стратегии, программирование, алгоритмы и модели, основанные на оценке распределений
- Понять основные стратегии исследования и освоения пространства для генетических алгоритмов
- Понять функционирование эволюционного программирования в применении к проблемам обучения и многоцелевым задачам
- Изучить основные понятия, связанные с нейронными сетями, и понять, как они работают в реальных ситуациях, применяемых в таких различных областях, как медицинские исследования, экономика и компьютерное зрение

03

Компетенции

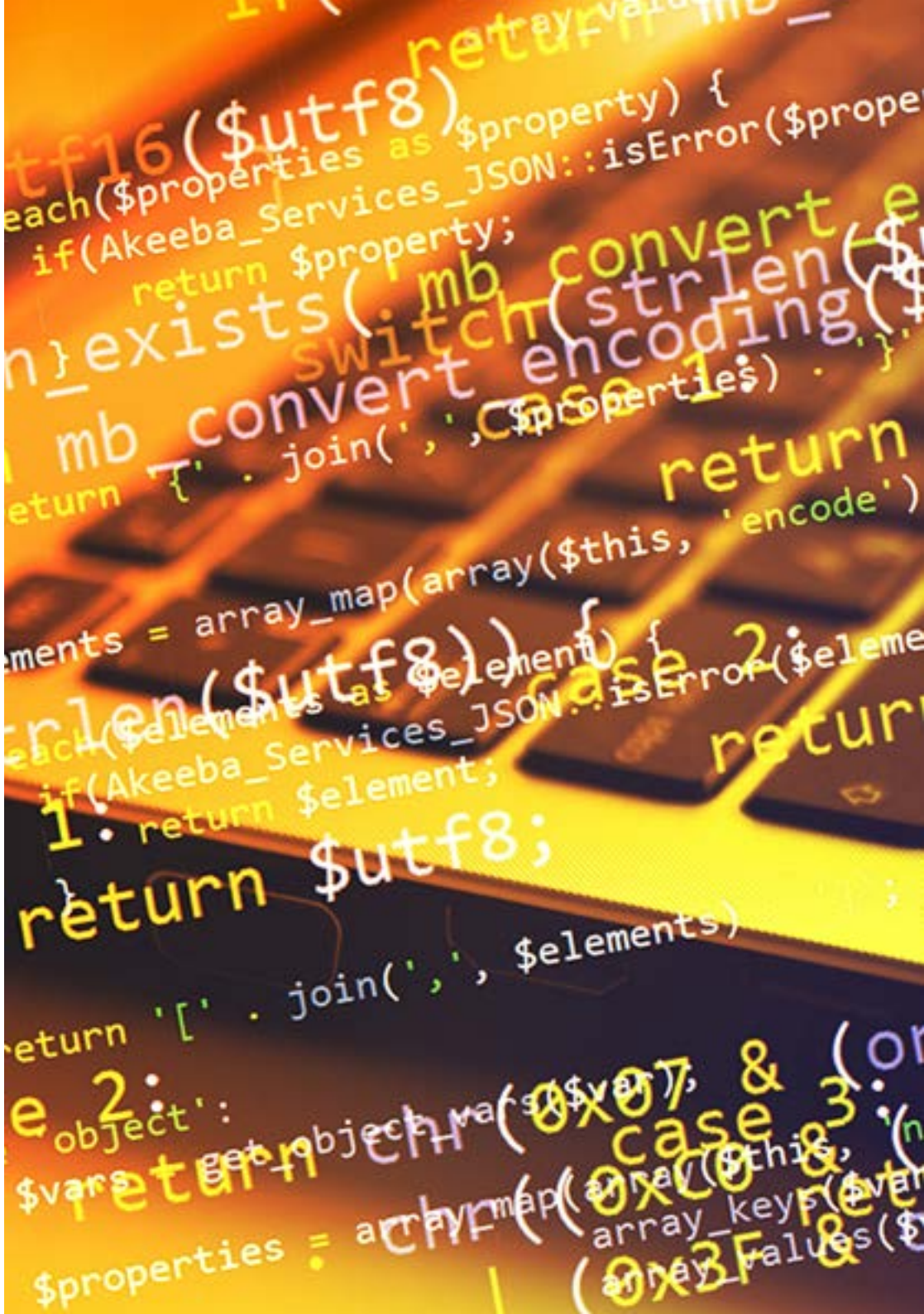
После сдачи экзаменов по программе "Информатика и языки" специалист приобретет необходимые компетенции для знания фундаментальных принципов вычислений с умением работать с языками программирования и данными.

Получите способность осуществлять новые компьютерные разработки, работая на основе понимания и контроля различных языков и алгоритмов и их практического применения"

Общий профессиональный навык

- Правильно выполнять задания, связанные с компьютерами и компьютерным языком

Совершенствуйте свои навыки для участия в различных технологических проектах"



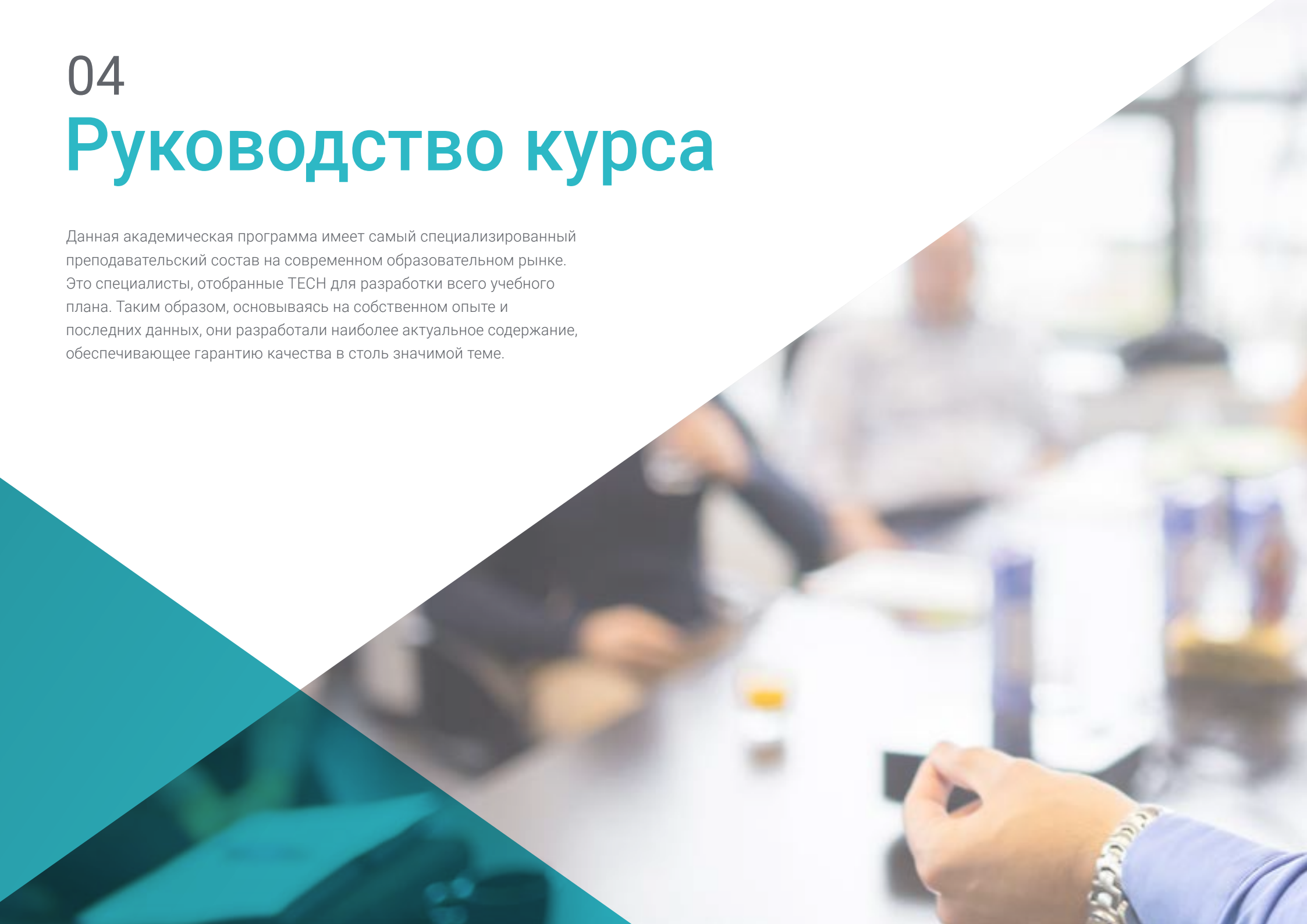
Профессиональные навыки

- Проектировать алгоритмы для разработки компьютерных программ и применять язык программирования
- Понимать и использовать структуру компьютерных данных
- Использовать алгоритмы, необходимые для решения компьютерных задач
- Глубоко изучить разработку передовых алгоритмов и методов поиска
- Выполнять задачи по компьютерному программированию
- Понимать и применять теорию, лежащую в основе компьютерных наук, например, математику
- Знать теорию автоматов и применять компьютерный язык
- Знать теоретические основы языков программирования и связанные с ними лексические, синтаксические и семантические методы обработки
- Понимать основные понятия математики и вычислительной сложности, чтобы применять их для решения вычислительных задач
- Знать и применять фундаментальные принципы вычислительной техники для осуществления новых компьютерных разработок

04

Руководство курса

Данная академическая программа имеет самый специализированный преподавательский состав на современном образовательном рынке. Это специалисты, отобранные TECH для разработки всего учебного плана. Таким образом, основываясь на собственном опыте и последних данных, они разработали наиболее актуальное содержание, обеспечивающее гарантию качества в столь значимой теме.



TECH предлагает вам самый специализированный преподавательский состав в области обучения. Поступайте прямо сейчас и наслаждайтесь качеством, которого вы заслуживаете”

Приглашенный международный руководитель

Доктор Джереми Гиббонс считается международным авторитетом благодаря своему вкладу в область методологии программирования и ее применения в программной инженерии. Более двух десятилетий этот специалист, работающий на факультете компьютерных наук Оксфордского университета, руководит различными проектами, наиболее ощутимые результаты которых применяются учеными-компьютерщиками в разных частях света.

Его работа охватывает такие области, как общее программирование, формальные методы, вычислительная биология, биоинформатика и разработка алгоритмов на языке Haskell. Последний был активно разработан совместно с его наставником, доктором Ричардом Бердом.

В качестве директора исследовательской группы алгебры программирования Гиббонс руководил разработками в области функциональных языков программирования и теории паттернов в программировании. В то же время его инновации нашли применение в сфере здравоохранения, о чем свидетельствует его сотрудничество с CancerGrid и Datatype-Generic Programming. Эти и другие инициативы отражают его интерес к решению практических проблем в области исследований рака и клинической информатики.

Гиббонс также добился значительных успехов в качестве главного редактора научных публикаций в журналах The Journal of Functional Programming и The Programming Journal: The Art, Science, and Engineering of Programming. Выполняя эти обязанности, он вел активную работу по распространению знаний. Кроме того, он возглавлял несколько учебных кафедр, связанных с такими известными учебными заведениями, как Оксфордский университет Брукс и Оклендский университет (Новая Зеландия).

Он также является членом рабочей группы 2.1 по алгоритмическим языкам и вычислениям Международной федерации по обработке информации (IFIP). В этой организации он обеспечивает сопровождение языков программирования ALGOL 60 и ALGOL 68.

Д-р. Гиббонс, Джереми

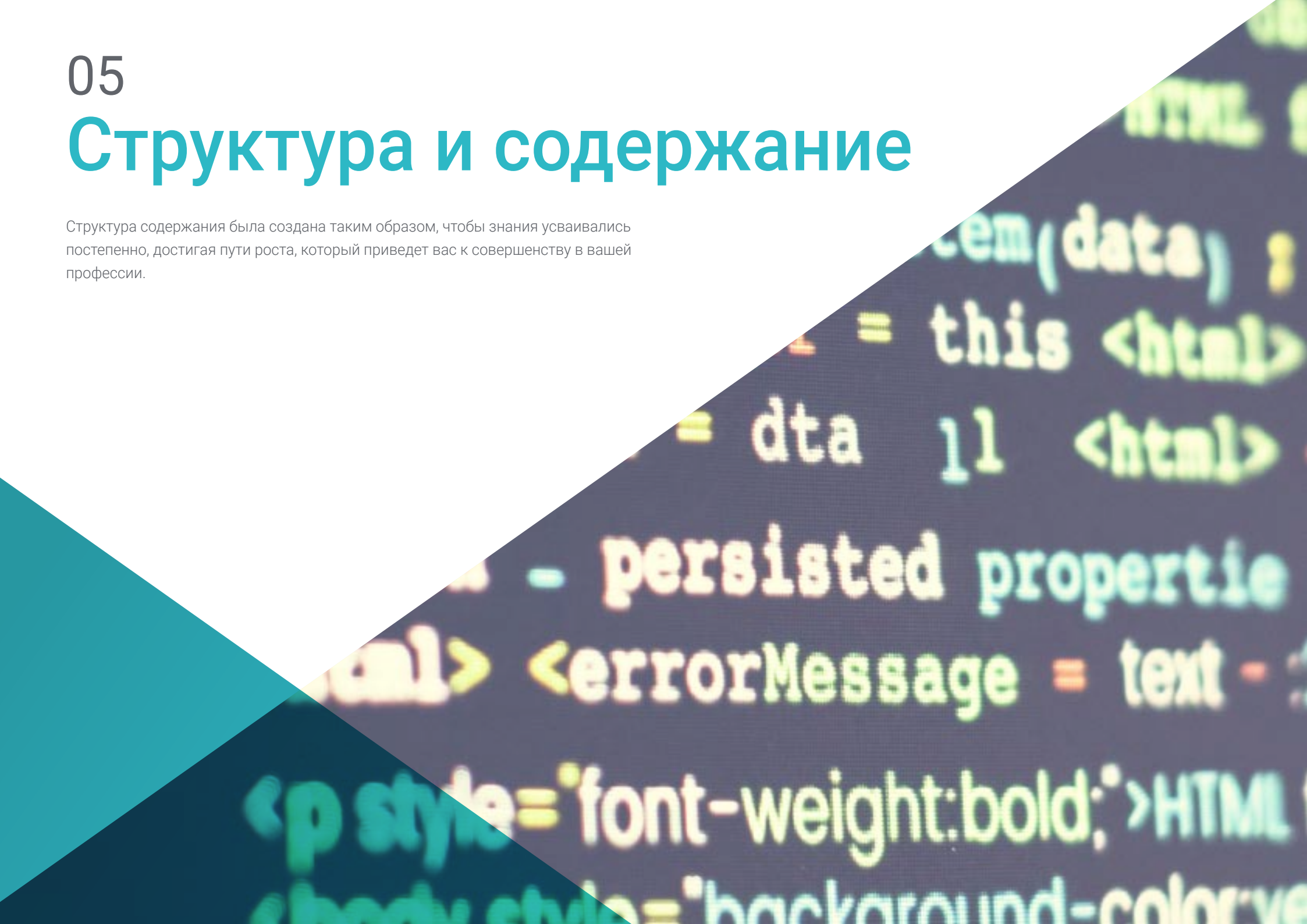
- Руководитель программы по разработке программного обеспечения, Оксфордский университет, Великобритания
- Заместитель руководителя Лаборатории информатики и факультета компьютерных наук Оксфордского университета
- Профессор Келлогского колледжа, Оксфордского университета Брукс и Оклендского университета в Новой Зеландии
- Руководитель исследовательской группы алгебры программирования
- Главный редактор журналов The Art, Science, and Engineering of Programming и Journal of Functional Programming
- Докторская степень в области компьютерных наук Оксфордского университета
- Степень бакалавра компьютерных наук Эдинбургского университета
  Член:
  Рабочая группа 2.1 Международной федерации по обработке информации (IFIP) по алгоритмическим языкам и вычислениям (WG2.1)

Благодаря TECH вы сможете учиться у лучших мировых профессионалов”

05

Структура и содержание

Структура содержания была создана таким образом, чтобы знания усваивались постепенно, достигая пути роста, который приведет вас к совершенству в вашей профессии.



Все области, которые вам необходимо освоить для безопасной и успешной работы в области вычислений и языков, собраны в учебном плане высшего качества"

Модуль 1. Основы программирования

1.1. Введение в программирование
- 1.1.1. Базовая структура компьютера
- 1.1.2. Программное обеспечение
- 1.1.3. Языки программирования
- 1.1.4. Жизненный цикл программного приложения

1.2. Проектирование алгоритмов
- 1.2.1. Решение проблем
- 1.2.2. Описательные техники
- 1.2.3. Элементы и структура алгоритма

1.3. Составные части программы
- 1.3.1. Происхождение и характеристики языка C++
- 1.3.2. Среда разработки
- 1.3.3. Концепция программы
- 1.3.4. Основные типы данных
- 1.3.5. Операторы
- 1.3.6. Выражения
- 1.3.7. Утверждения
- 1.3.8. Ввод и вывод данных

1. 4. Контрольные предложения
- 1.4.1. Утверждения
- 1.4.2. Бифуркации
- 1.4.3. Циклы

1.5. Абстракция и модульность: функции
- 1.5.1. Модульный дизайн
- 1.5.2. Понятие функции и пользы
- 1.5.3. Определение функции
- 1.5.4. Поток выполнения при вызове функции
- 1.5.5. Прототип функции
- 1.5.6. Возврат результатов
- 1.5.7. Вызов функции: параметры
- 1.5.8. Передача параметров по ссылке и по значению
- 1.5.9. Идентификация области

1.6. Статические структуры данных
- 1.6.1. *Arrays*
- 1.6.2. Матрицы. Полиэдры
- 1.6.3. Поиск и сортировка
- 1.6.4. Цепочки. Функции ввода/вывода для строк
- 1.6.5. Структуры. Союзы
- 1.6.6. Новые виды данных

1.7. Динамические структуры данных: указатели
- 1.7.1. Понятие. Определение указателя
- 1.7.2. Операторы и операции с указателями
- 1.7.3. *Arrays* указателей
- 1.7.4. Указатели и *arrays*
- 1.7.5. Указатели на строки
- 1.7.6. Указатели на структуры
- 1.7.7. Множественная косвенность
- 1.7.8. Указатели на функции
- 1.7.9. Передача функций, структур и *arrays* в качестве параметров функции

1.8. Файлы
- 1.8.1. Основные понятия
- 1.8.2. Операции с файлами
- 1.8.3. Типы файлов
- 1.8.4. Организация архивов
- 1.8.5. Введение в файлы C++
- 1.8.6. Управление файлами

1.9. Рекурсивность
- 1.9.1. Определение рекурсии
- 1.9.2. Виды рекурсии
- 1.9.3. Преимущества и недостатки
- 1.9.4. Соображения
- 1.9.5. Итеративная рекурсивная конверсия
- 1.9.6. Стек рекурсии

1.10. Доказательства и документация
- 1.10.1. Тестирование программ
- 1.10.2. Тестирование методом «белого ящика»
- 1.10.3. Тестирование методом «черного ящика»
- 1.10.4. Инструменты для тестирования
- 1.10.5. Программная документация

Модуль 2. Структура данных

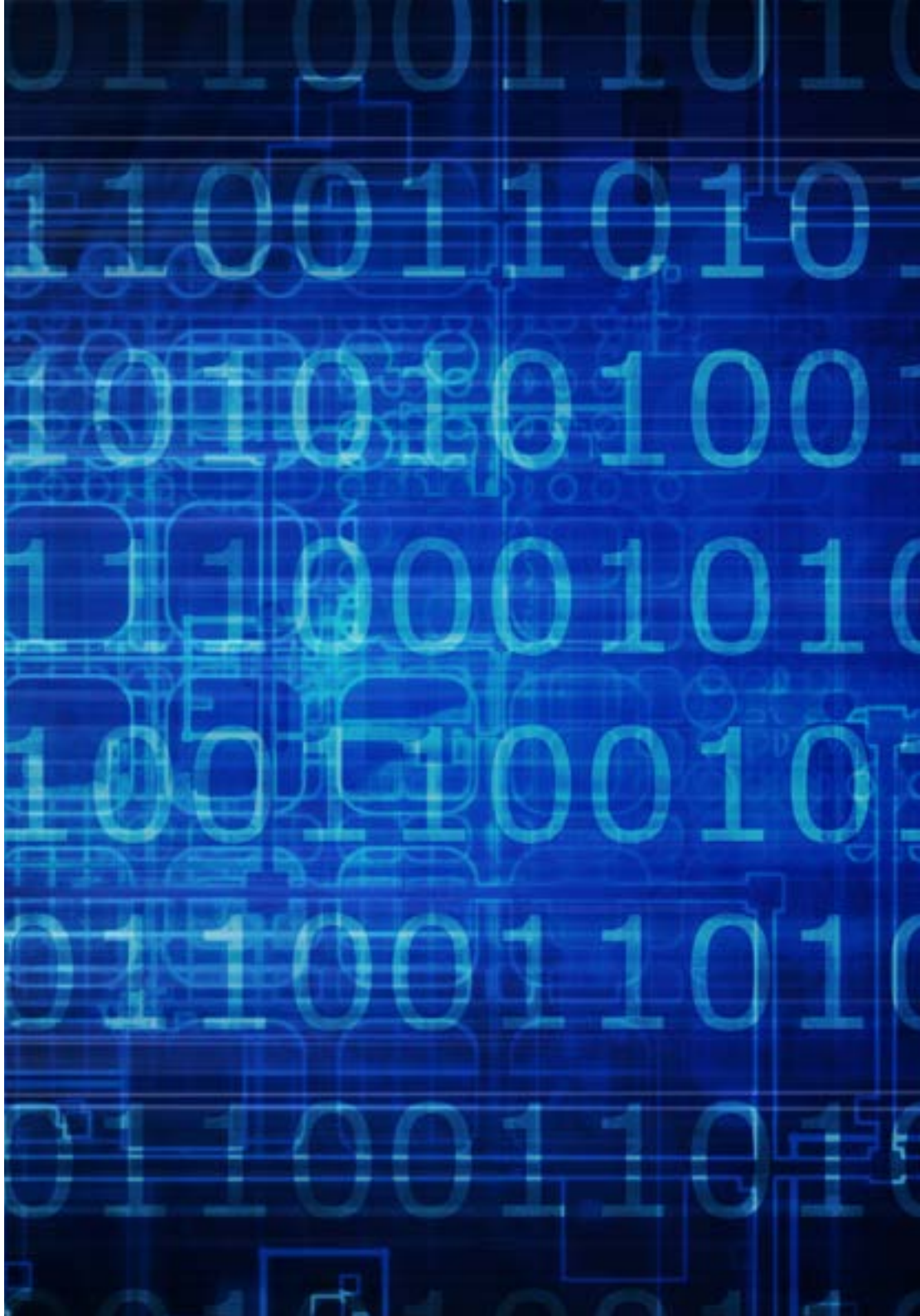
2.1. Введение в программирование на C++
- 2.1.1. Классы, конструкторы, методы и атрибуты
- 2.1.2. Переменные
- 2.1.3. Условные выражения и циклы
- 2.1.4. Предметы

2.2. Абстрактные типы данных (ADT)
- 2.2.1. Типы данных
- 2.2.2. Основные структуры и ADT
- 2.2.3. Векторы и Arrays

2.3. Линейные структуры данных
- 2.3.1. Список ADT. Определение
- 2.3.2. Связанные и дважды связанные списки
- 2.3.3. Упорядоченные списки
- 2.3.4. Списки в C++
- 2.3.5. Стек ADT
- 2.3.6. Очередь ADT
- 2.3.7. Стек и очередь на C++

2.4. Иерархический структуры данных
- 2.4.1. Дерево ADT
- 2.4.2. Маршруты
- 2.4.3. N-арные деревья
- 2.4.4. Бинарные деревья
- 2.4.5. Деревья бинарного поиска

2.5. Иерархические структуры данных: сложные деревья
  2.5.1. Идеально сбалансированные деревья или деревья минимальной высоты
  2.5.2. Многопутевые деревья
  2.5.3. Библиографические ссылки
2.6. Кучи и приоритетные очереди
  2.6.1. Кучи ADT
  2.6.2. Приоритетные очереди ADT
2.7. Хеш-таблицы
  2.7.1. *Хэш-таблица* ADT
  2.7.2. *Хэш-функции*
  2.7.3. *Хэш-функция* в *хэш-таблицах*
  2.7.4. Редисперсия
  2.7.5. Открытые *хэш-таблицы*
2.8. Графы
  2.8.1. ADT Графы
  2.8.2. Виды графов
  2.8.3. Графическое представление и основные операции
  2.8.4. Разработка графов
2.9. Алгоритмы и расширенные концепции графов
  2.9.1. Задачи на графах
  2.9.2. Алгоритмы на путях
  2.9.3. Алгоритмы поиска или пути
  2.9.4. Прочие алгоритмы
2.10. Прочие структуры данных
  2.10.1. Наборы
  2.10.2. Arrays параллельные
  2.10.3. Таблица символов
  2.10.4. *Пробы*



Модуль 3. Алгоритм и сложность

3.1. Введение в стратегии проектирования алгоритмов
- 3.1.1. Рекурсивность
- 3.1.2. Разделяй и властвуй
- 3.1.3. Прочие стратегии

3.2. Эффективность и анализ алгоритмов
- 3.2.1. Меры эффективности
- 3.2.2. Измерение размера входа
- 3.2.3. Измерение времени выполнения
- 3.2.4. Худший, лучший и средний случай
- 3.2.5. Асимптотическое обозначение
- 3.2.6. Критерии математического анализа нерекурсивных алгоритмов
- 3.2.7. Математический анализ рекурсивных алгоритмов
- 3.2.8. Эмпирический анализ алгоритмов

3.3. Алгоритмы сортировки
- 3.3.1. Концепция организации
- 3.3.2. Сортировка пузырьком
- 3.3.3. Сортировка по выбору
- 3.3.4. Сортировка вставками
- 3.3.5. Сортировка слиянием (Merge Sort)
- 3.3.6. Быстрая сортировка (QuickSort)

3.4. Алгоритмы с применением деревьев
- 3.4.1. Концепция деревьев
- 3.4.2. Бинарные деревья
- 3.4.3. Обход дерева
- 3.4.4. Представление выражений
- 3.4.5. Упорядоченные бинарные деревья
- 3.4.6. Сбалансированные бинарные деревья

3.5. Алгоритмы с *Heaps*
- 3.5.1. *Heaps*
- 3.5.2. Алгоритм HeapSort
- 3.5.3. Приоритетные очереди

3,6. Алгоритмы с применением графов
- 3.6.1. Представление
- 3.6.2. Обход по ширине
- 3.6.3. Углубленный обход
- 3.6.4. Топологическая сортировка

3.7. *Жадные* алгоритмы
- 3.7.1. *Жадная* стратегия
- 3.7.2. Элементы *жадной* стратегии
- 3.7.3. Обмен валюты
- 3.7.4. Задача коммивояжёра
- 3.7.5. Задача о рюкзаке

3.8. Поиск минимальных путей
- 3.8.1. Проблема минимального пути
- 3.8.2. Отрицательные дуги и циклы
- 3.8.3. Алгоритм Дейкстры

3.9. Жадные алгоритмы на графах
- 3.9.1. Дерево минимального покрытия
- 3.9.2. Алгоритм Прима
- 3.9.3. Алгоритм Крускала
- 3.9.4. Анализ сложности

3.10. *Backtracking*
- 3.10.1. *Backtracking*
- 3.10.2. Альтернативные методы

Модуль 4. Продвинутое проектирование алгоритмов

4.1. Анализ рекурсивных алгоритмов и алгоритмов «разделяй и властвуй»
- 4.1.1. Составление и решение однородных и неоднородных рекуррентных уравнений
- 4.1.2. Суть метода разделяй и властвуй

4.2. Амортизированный анализ
- 4.2.1. Агрегатный анализ
- 4.2.2. Метод учета
- 4.2.3. Метод потенциалов

4.3. Динамическое программирование и алгоритмы для NP-задач
- 4.3.1. Характеристики динамического программирования
- 4.3.2. Обратный ход: backtracking
- 4.3.3. Ветвление и подрезка

4.4. Комбинаторная оптимизация
- 4.4.1. Представление проблем
- 4.4.2. 1D оптимизация

4.5. Рандомизированные алгоритмы
- 4.5.1. Примеры алгоритмов рандомизации
- 4.5.2. Теорема Бюффона
- 4.5.3. Алгоритм Монте-Карло
- 4.5.4. Алгоритм Лас-Вегаса

4.6. Локальный поиск и с кандидатов
- 4.6.1. *Garcient Ascent*
- 4.6.2. *Поиск восхождением к вершине*
- 4.6.3. *Имитационный отжиг*
- 4.6.4. *Поиск с запретами*
- 4.6.5. Поиск с помощью кандидатов

4.7. Формальная верификация программ
- 4.7.1. Определение функциональных абстракций
- 4.7.2. Язык логики первого порядка
- 4.7.3. Формальная система Хоара

4.8. Верификация итеративных программ
- 4.8.1. Правила формальной системы Хоара
- 4.8.2. Концепция инвариантных итераций

4.9. Численные методы
- 4.9.1. Метод бисекции
- 4.9.2. Метод Ньютона-Рафсона
- 4.9.3. Метод секущей

4.10. Параллельные алгоритмы
- 4.10.1. Параллельные бинарные операции
- 4.10.2. Параллельные вычисления на графах
- 4.10.3. Параллелизм в «разделяй и властвуй»
- 4.10.4. Параллелизм в динамическом программировании

Модуль 5. Расширенное программирование

5.1. Введение в объектно-ориентированное программирование
- 5.1.1. Введение в объектно-ориентированное программирование
- 5.1.2. Разработка классов
- 5.1.3. Введение в UML для моделирования проблем

5.2. Отношения между классами
- 5.2.1. Абстракция и наследники
- 5.2.2. Расширенные концепции наследников
- 5.2.3. Полиморфизм
- 5.2.4. Состав и агрегация

5.3. Введение в шаблоны проектирования для решения объектно-ориентированных задач
- 5.3.1. Что такое шаблоны проектирования?
- 5.3.2. Шаблон *Factory*
- 5.3.3. Шаблон *Singleton*
- 5.3.4. Шаблон *Observer*
- 5.3.5. Шаблон *Composite*

5.4. Исключения
- 5.4.1. Что такое исключения?
- 5.4.2. Перехват и обработка исключений
- 5.4.3. Запуск исключений
- 5.4.4. Создание исключений

5.5. Пользовательские интерфейсы
- 5.5.1. Введение в Qt

5.5.2. Позиционирование
5.5.3. Что такое события?
5.5.4. События: определение и фиксация
5.5.5. Разработка пользовательского интерфейса
5.6. Введение в параллельное программирование
5.6.1. Введение в параллельное программирование
5.6.2. Понятие процесса и потока
5.6.3. Взаимодействие между процессами или потоками
5.6.4. Потоки в C++
5.6.5. Преимущества и недостатки параллельного программирования
5.7. Управление и синхронизация потоков
5.7.1. Жизненный цикл потока
5.7.2. Класс *Thread*
5.7.3. Планирование потоков
5.7.4. Группы потоков
5.7.5. Потоки демонического типа
5.7.6. Синхронизация
5.7.7. Механизмы блокировки
5.7.8. Механизмы коммуникации
5.7.9. Мониторы
5.8. Распространенные проблемы параллельного программирования
5.8.1. Проблема производитель-потребитель
5.8.2. Задача о читателях-писателях
5.8.3. Задача об обедающих философах
5.9. Документация и тестирование ПО
5.9.1. Почему важно документировать ПО?
5.9.2. Проектный документ
5.9.3. Использование инструментов для документирования
5.10. Тестирование ПО
5.10.1. Введение в тестирование ПО
5.10.2. Виды тестирования
5.10.3. Модульный тест
5.10.4. Интеграционный тест
5.10.5. Валидационный тест
5.10.6. Системный тест

Модуль 6. Теоретическая информатика

6.1. Используемые математические понятия
6.1.1. Введение в пропозициональную логику
6.1.2. Теория взаимоотношений
6.1.3. Нумеруемые и ненумеруемые множества
6.2. Формальные языки и грамматики и введение в машины Тьюринга
6.2.1. Формальные языки и грамматики
6.2.2. Проблема принятия решения
6.2.3. Машина Тьюринга
6.3. Расширения для машин Тьюринга, ограниченные машины Тьюринга и компьютеры
6.3.1. Методы программирования для машин Тьюринга
6.3.2. Расширения для машин Тьюринга
6.3.3. Машины Тьюринга с ограничениями
6.3.4. Машины Тьюринга и компьютеры
6.4. Неразрешимость
6.4.1. Нерекурсивно перечислимый язык
6.4.2. Неразрешимая рекурсивно перечислимая проблема
6.5. Другие неразрешимые проблемы
6.5.1. Неразрешимые проблемы для машин Тьюринга
6.5.2. Проблема соответствия Поста (PCP)
6.6. Неразрешимые проблемы
6.6.1. Классы P и NP
6.6.2. Полная NP-задача
6.6.3. Ограниченная проблема удовлетворительности
6.6.4. Другие завершенные NP-задачи
6.7. Задачи совместного NP и PS
6.7.1. Дополнение к языкам NP
6.7.2. Решаемые задачи в полиномиальном пространстве
6.7.3. Решение задач по PS

6.8. Классы языков, основанных на рандомизации
- 6.8.1. MT-модель со случайностью
- 6.8.2. Классы RP и ZPP
- 6.8.3. Тест на первичность
- 6.8.4. Сложность проверки на первичность

6.9. Другие классы и грамматики
- 6.9.1. Вероятностные конечные автоматы
- 6.9.2. Клеточные автоматы
- 6.9.3. Клетки МакКаллока и Питтса
- 6.9.4. Грамматики Линденмайера

6.10. Передовые вычислительные системы
- 6.10.1. Мембранные вычисления: P-системы
- 6.10.2. ДНК-вычисления
- 6.10.3. Квантовые вычисления

Модуль 7. Теория автоматов и формальных языков

7.1. Введение в теорию автоматов
- 7.1.1. Зачем изучать теорию автоматов?
- 7.1.2. Введение в формальные демонстрации
- 7.1.3. Другие формы демонстрации
- 7.1.4. Математическая индукция
- 7.1.5. Алфавиты, строки и языки

7.2. Детерминированные конечные автоматы
- 7.2.1. Введение в конечные автоматы
- 7.2.2. Детерминированные конечные автоматы

7.3. Недетерминированные конечные автоматы (DFA)
- 7.3.1. Недетерминированные конечные автоматы (NFA)
- 7.3.2. Эквивалентность между DFA и NFA
- 7.3.3. Конечные автоматы с переходами

7.4. Языки и регулярные выражения (I)
- 7.4.1. Языки и регулярные выражения
- 7.4.2. Конечные автоматы и регулярные выражения

7.5. Языки и регулярные выражения (II)
- 7.5.1. Преобразование регулярных выражений в автоматы
- 7.5.2. Применение регулярных выражений
- 7.5.3. Алгебра регулярных выражений

7.6. Накачка леммы и закрытие регулярных языков
- 7.6.1. Накачка леммы
- 7.6.2. Свойства закрытия регулярных языков

7.7. Эквивалентность и минимизация автоматов
- 7.7.1. Эквивалентность конечных автоматов
- 7.7.2. Минимизация конечных автоматов

7.8. Контекстно-свободная грамматика (CFL)
- 7.8.1. Контекстно-свободная грамматика (CFG)
- 7.8.2. Деривационные деревья
- 7.8.3. Применение CFG
- 7.8.4. Неоднозначность в грамматиках и языках

7.9. Автомат со стеком и CFG
- 7.9.1. Определение автоматов со стеком
- 7.9.2. Языки, принимаемые стековым автоматом
- 7.9.3. Эквивалентность между стековыми автоматами и CFG
- 7.9.4. Детерминированный стековый автомат

7.10. Нормальные формы, схема прокачки CFG и свойства CFL
- 7.10.1. Обычные формы CFG
- 7.10.2. Лемма о разрастании
- 7.10.3. Свойства закрытия языков
- 7.10.4. Свойства решений CFL

Модуль 8. Языковые процессоры

8.1. Введение в процесс компиляции
- 8.1.1. Составление и интерпретация
- 8.1.2. Среда выполнения компилятора
- 8.1.3. Процесс анализа
- 8.1.4. Процесс синтеза

8.2. Лексический анализатор
- 8.2.1. Что такое лексический анализатор?
- 8.2.2. Реализация лексического анализатора
- 8.2.3. Семантические действия
- 8.2.4. Устранение ошибок
- 8.2.5. Вопросы реализации

8.3. Синтактический анализ
- 8.3.1. Что такое синтаксический анализатор?
- 8.3.2. Предварительные концепции
- 8.3.3. Нисходящие анализаторы
- 8.3.4. Восходящие анализаторы

8.4. Разбор нисходящий и разбор восходящий
- 8.4.1. LL(1) Анализатор
- 8.4.2. LR(0) Анализатор
- 8.4.3. Пример анализатора

8.5. Расширенный восходящий синтаксический анализ
- 8.5.1. Анализатор SLR
- 8.5.2. Анализатор LR(1)
- 8.5.3. Анализатор LR(k)
- 8.5.4. Анализатор LALR

8.6. Семантический анализ (I)
- 8.6.1. Перевод на основе синтаксиса
- 8.6.2. Таблица символов

8.7. Семантический анализ (II)
- 8.7.1. Проверка типов
- 8.7.2. Подсистема типов
- 8.7.3. Эквивалентность ставок и конвертация

8.8. Среда генерации и выполнения кода
- 8.8.1. Аспекты проектирования
- 8.8.2. Среда выполнения
- 8.8.3. Организация памяти
- 8.8.4. Распределение памяти

8.9. Генерация промежуточного кода
- 8.9.1. Перевод на основе синтеза
- 8.9.2. Промежуточные представления
- 8.9.3. Примеры переводов

8.10. Оптимизация кода
- 8.10.1. Назначение регистров
- 8.10.2. Устранение мертвых распределений
- 8.10.3. Выполнение во время компиляции
- 8.10.4. Перестановка выражений
- 8.10.5. Оптимизация циклов

Модуль 9. Компьютерная графика и визуализация

9.1. Теория цвета
- 9.1.1. Свойства света
- 9.1.2. Модели цветов
- 9.1.3. Стандарт CIE
- 9.1.4. *Профилирование*

9.2. Примитивы вывода
- 9.2.1. Видеоконтроллер
- 9.2.2. Алгоритмы рисования линий
- 9.2.3. Алгоритмы для рисования окружностей
- 9.2.4. Алгоритмы заполнения

9.3. 2D-преобразования и системы координат и 2D-обрезка
- 9.3.1. Основные геометрические преобразования
- 9.3.2. Однородные координаты
- 9.3.3. Обратное преобразование
- 9.3.4. Композиция преобразований
- 9.3.5. Другие преобразования
- 9.3.6. Изменения координата
- 9.3.7. 2D-системы координат
- 9.3.8. Изменения координат
- 9.3.9. Нормализация
- 9.3.10. Алгоритмы обрезки

9.4. 3D-преобразования
- 9.4.1. Конверсия
- 9.4.2. Ротация
- 9.4.3. Масштабирование
- 9.4.4. Размышления
- 9.4.5. Shear

9.5. Визуализация и изменение 3D-координат
- 9.5.1. 3D-системы координат
- 9.5.2. Визуализация
- 9.5.3. Изменения координат
- 9.5.4. Проекция и стандартизация

9.6. 3D-проекция и обрезка
- 9.6.1. Ортогональная проекция
- 9.6.2. Косая параллельная проекция
- 9.6.3. Перспективная проекция
- 9.6.4. Алгоритмы 3D-обрезки

9.7. Удаление скрытых поверхностей
- 9.7.1. *Back face removal*
- 9.7.2. Z-buffer
- 9.7.3. Алгоритм художника
- 9.7.4. Алгоритм Варнока
- 9.7.5. Выявление скрытых линий

9.8. Интерполяция и параметрические кривые
- 9.8.1. Интерполяция и полиномиальная аппроксимация
- 9.8.2. Параметрическое представление
- 9.8.3. Многочлен Лагранжа
- 9.8.4. Естественные кубические *сплайны*
- 9.8.5. Основные функции
- 9.8.6. Матричное представление

9.9. Кривые Безье
- 9.9.1. Алгебраическое построение
- 9.9.2. Матричная форма
- 9.9.3. Состав
- 9.9.4. Геометрическое построение
- 9.9.5. Алгоритм рисования

9.10. *B-сплайны*
- 9.10.1. Проблема местного контроля
- 9.10.2. Равномерные кубические B-сплайны
- 9.10.3. Базовые функции и контрольные точки
- 9.10.4. Переход к происхождению и множественности
- 9.10.5. Матричное представление
- 9.10.6. Неравномерные кубические *B-сплайны*

Модуль 10. Биоинспирированные вычисления

10.1. Введение в биоинспирированные вычисления
- 10.1.1. Введение в биоинспирированные вычисления

10.2. Алгоритмы социальной адаптации
- 10.2.1. Биоинспирированные вычисления на основе муравьиных колоний
- 10.2.2. Разновидности алгоритмов муравьиных колоний
- 10.2.3. Облачные вычисления с частицами

10.3. Генетические алгоритмы
- 10.3.1. Общая структура
- 10.3.2. Внедрения основных операторов

10.4. Стратегии освоения и использования пространства для генетических алгоритмов
- 10.4.1. Алгоритм CHC
- 10.4.2. Мультимодальные задачи

10.5. Модели эволюционных вычислений (I)
- 10.5.1. Эволюционные стратегии
- 10.5.2. Эволюционное программирование
- 10.5.3. Алгоритмы, основанные на дифференциальной эволюции

10.6. Модели эволюционных вычислений (II)
- 10.6.1. Модели эволюции, основанные на оценке распределений (EDA)
- 10.6.2. Генетическое программирование

10.7. Применение развивающего программирования при проблемах с обучением
- 10.7.1. Обучение на основе правил
- 10.7.2. Эволюционные методы в задачах выбора экземпляра

10.8. Многоцелевые задачи
- 10.8.1. Концепция доминирования
- 10.8.2. Применение эволюционных алгоритмов для решения многоцелевых задач

10.9. Нейросети (I)
- 10.9.1. Введение в нейросети
- 10.9.2. Практический пример с нейросетями

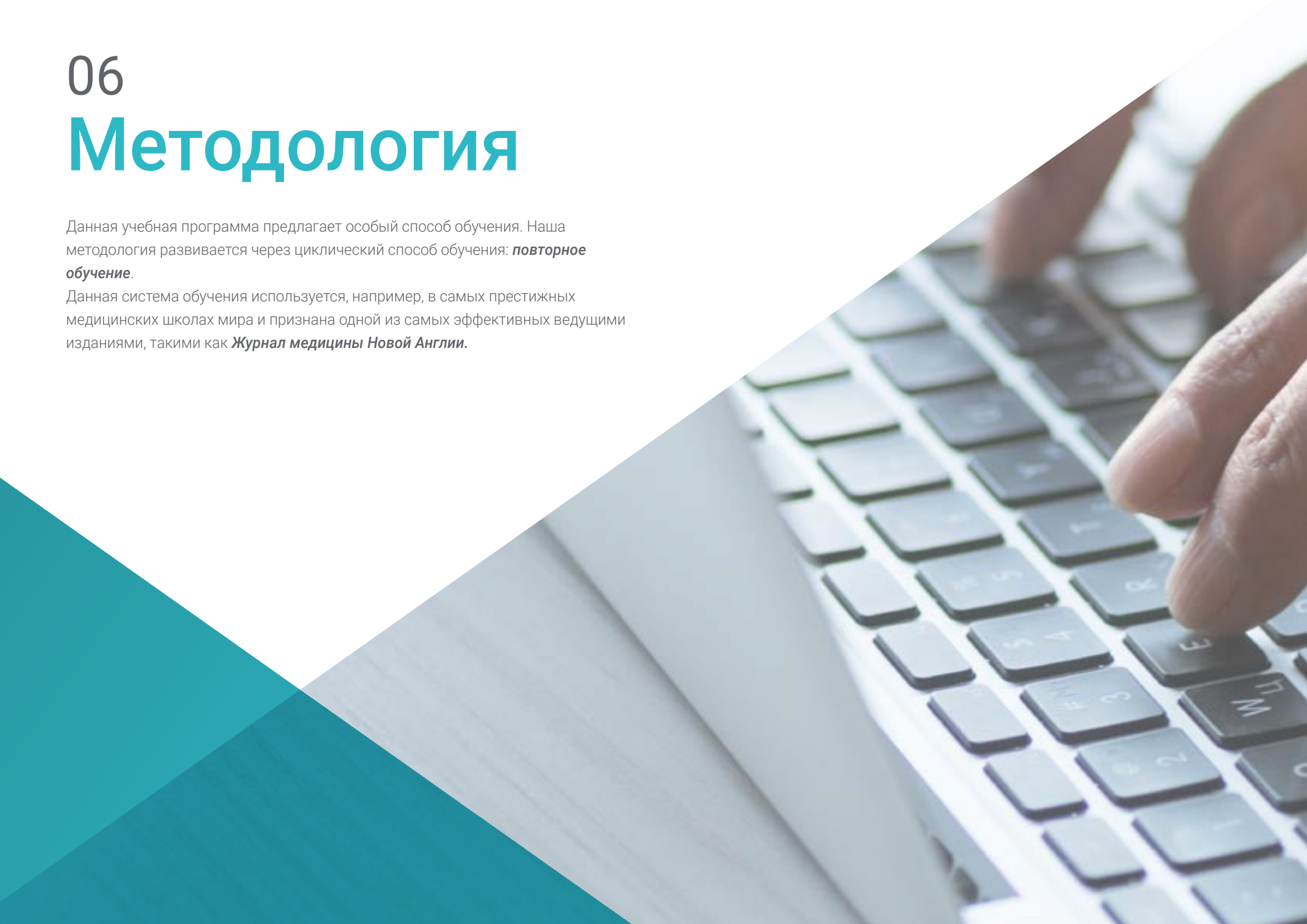
10.10. Нейросети (II)
- 10.10.1. Примеры использования нейросетей в медицинских исследованиях
- 10.10.2. Примеры использования нейросетей в экономике
- 10.10.3. Примеры использования нейросетей в компьютерном зрении

06

Методология

Данная учебная программа предлагает особый способ обучения. Наша методология развивается через циклический способ обучения: ***повторное обучение***.

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, как на национальном, так и 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 метода кейсов - 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

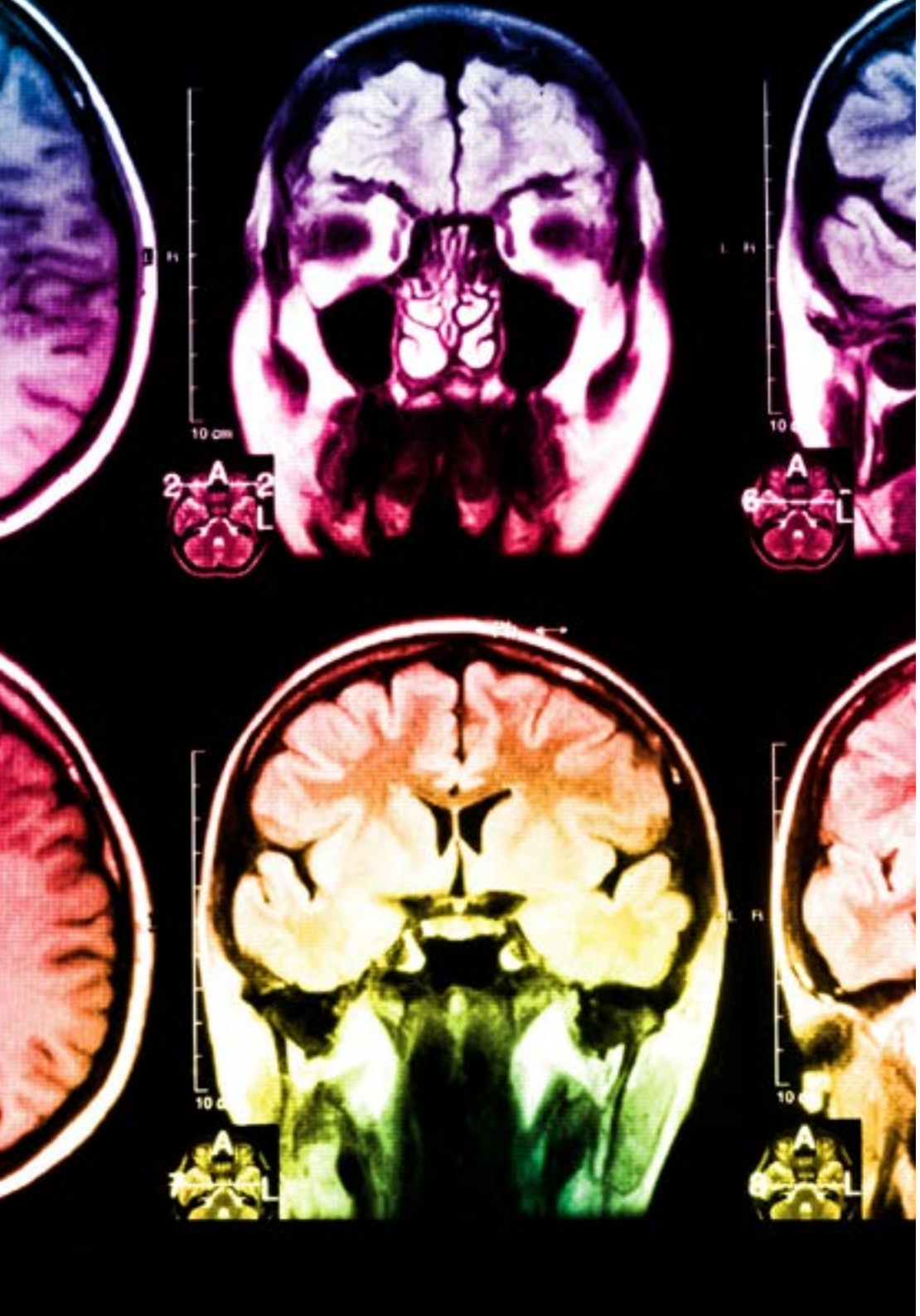
TECH эффективно объединяет метод кейсов с системой 100% онлайн-обучения, основанной на повторении, которая сочетает различные дидактические элементы в каждом уроке.

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

В 2019 году мы достигли лучших результатов обучения среди всех испаноязычных онлайн-университетов мира.

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 испаноязычного 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, инженерия, 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.

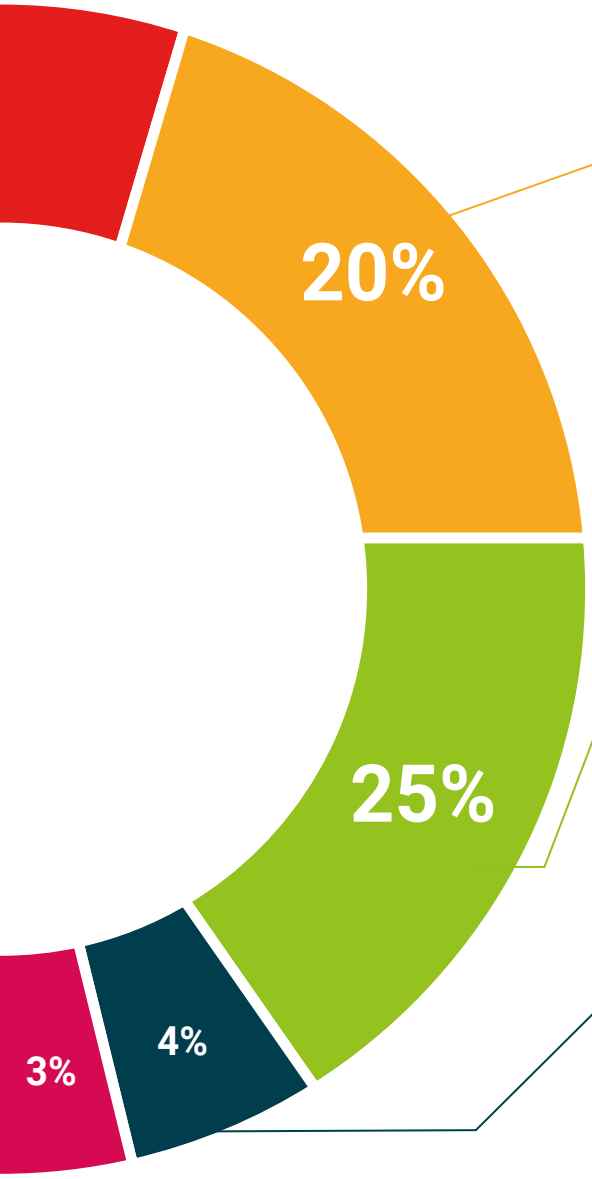
Так называемый метод обучения у эксперта укрепляет знания и память, а также формирует уверенность в наших будущих сложных решениях.

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".

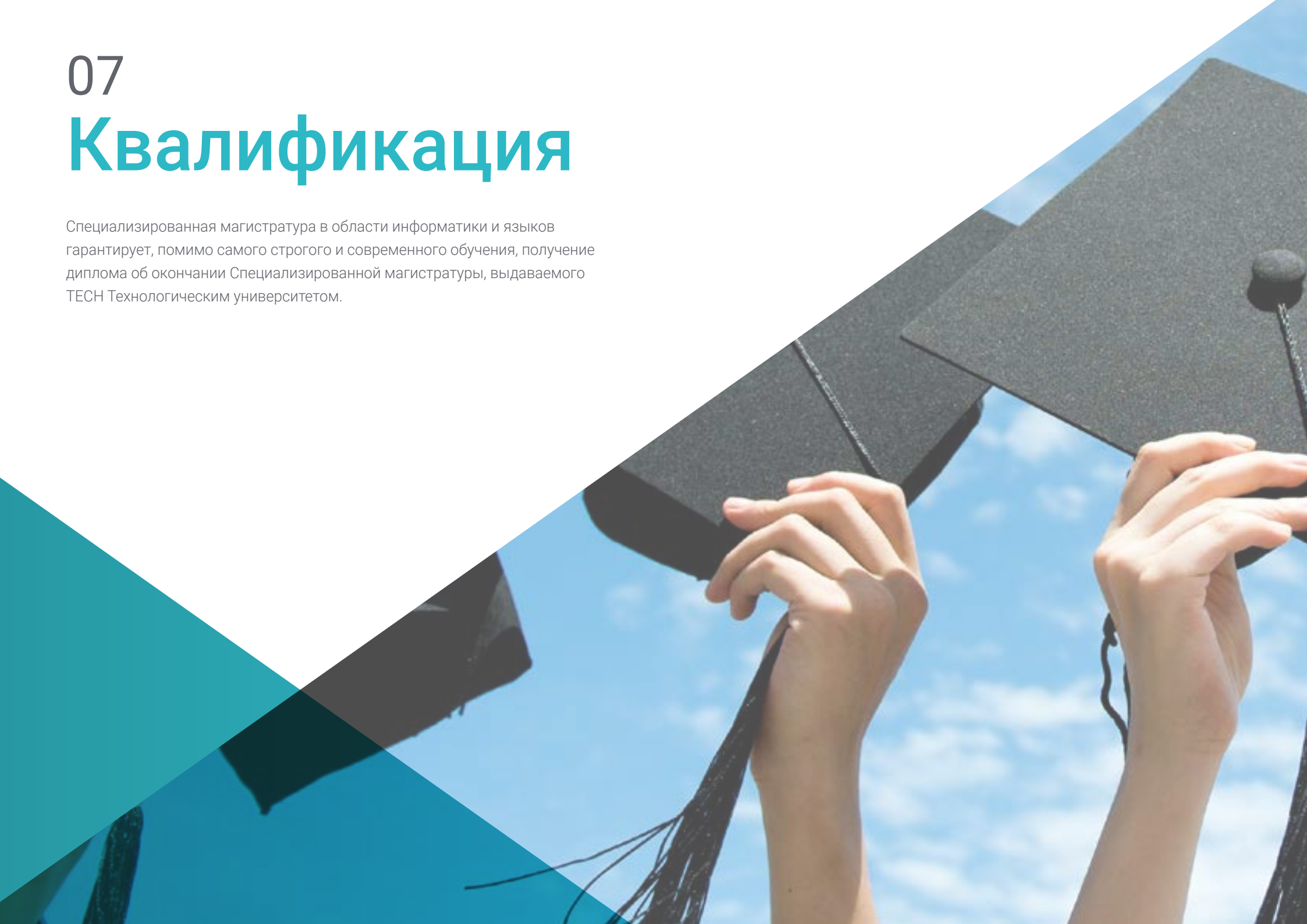
Тестирование и повторное тестирование

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 информатики и языков 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 информатики и языков** 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 информатики и языков**

Формат: **онлайн**

Продолжительность: **12 месяцев**


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

информатики и языков

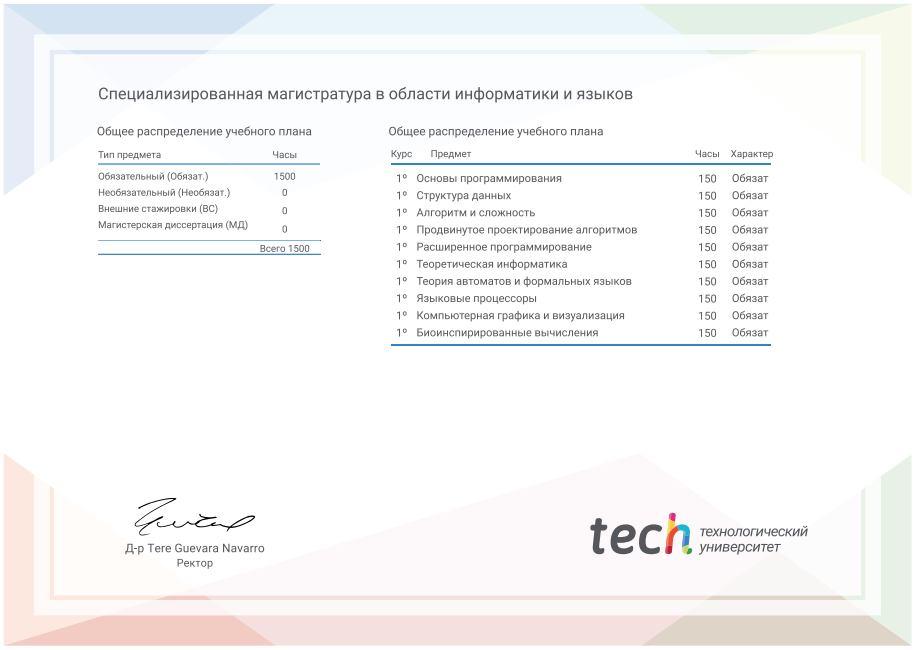
Данный диплом специализированной программы, присуждаемый Университетом, соответствует 1500 учебным часам, с датой начала дд/мм/гггг и датой окончания дд/мм/гггг.

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: AFWOR235 techtitute.com/certificates


Специализированная магистратура в области информатики и языков

Общее распределение учебного плана

Тип предмета	Часы
Обязательный (Обязат.)	1500
Необязательный (Необязат.)	0
Внешние стажировки (ВС)	0
Магистерская диссертация (МД)	0
	Всего 1500

Общее распределение учебного плана

Курс	Предмет	Часы	Характер
1°	Основы программирования	150	Обязат
1°	Структура данных	150	Обязат
1°	Алгоритм и сложность	150	Обязат
1°	Продвинутое проектирование алгоритмов	150	Обязат
1°	Расширенное программирование	150	Обязат
1°	Теоретическая информатика	150	Обязат
1°	Теория автоматов и формальных языков	150	Обязат
1°	Языковые процессоры	150	Обязат
1°	Компьютерная графика и визуализация	150	Обязат
1°	Биоинспирированные вычисления	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

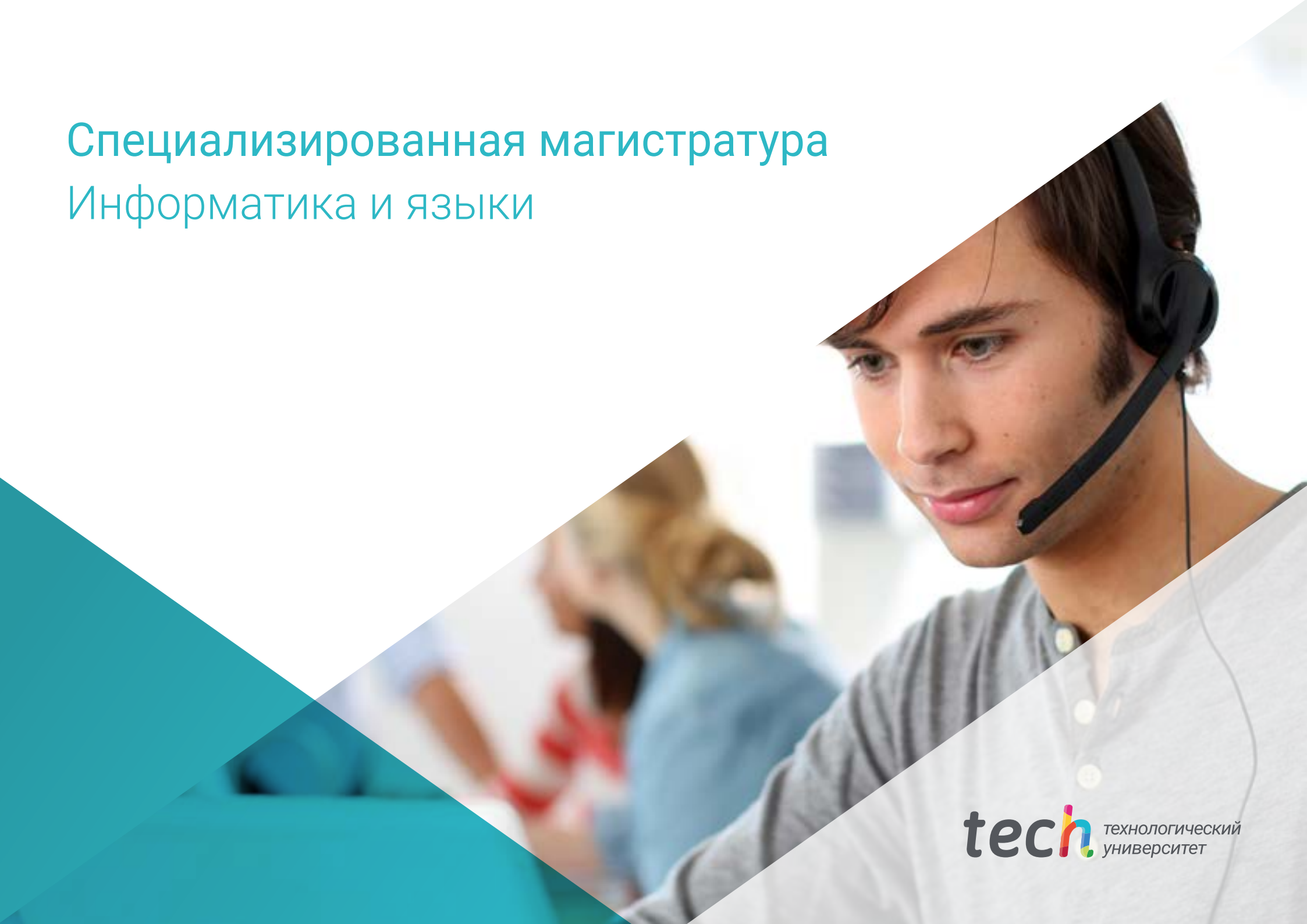
Специализированная магистратура

Информатика и языки

- Формат: **онлайн**
- Продолжительность: **12 месяцев**
- Учебное заведение: **TECH Технологический университет**
- Расписание: **по своему усмотрению**
- Экзамены: **онлайн**

Специализированная магистратура

Информатика и языки



tech технологический университет