**Backend engineer**

Освой серверную разработку: создавай мощные API, работай с базами данных и обеспечивай стабильность приложений. Стань мастером бэкенда!

**Технологии и языки, описанные в программе:**

1. **Ruby**
2. **Ruby on Rails**
3. **HTML5**
4. **CSS3**
5. **JavaScript**
   * Node.js
   * React
   * Другие библиотеки и фреймворки JavaScript
6. **SQL**
7. **C**
8. **Git**
9. **CI/CD инструменты**
   * Jenkins
   * Travis CI
10. **Облачные платформы**
    * Heroku
    * AWS

### **Навыки, которыми овладеют ученики:**

#### **Back-End Разработка**

* **Ruby и Ruby on Rails**
  + Использование Ruby для написания серверной логики.
  + Разработка приложений с использованием архитектуры MVC (Model-View-Controller).
  + Настройка RESTful маршрутов и работа с ActiveRecord ORM для взаимодействия с базами данных.
  + Имплементация бизнес-логики и API.
* **Работа с базами данных**
  + Проектирование схем баз данных и создание структурированных запросов SQL.
  + Работа с реляционными базами данных: проектирование, оптимизация хранения и управления данными.
* **Интеграция сторонних сервисов**
  + Настройка API и интеграция функций, таких как авторизация, платежные системы, синхронизация данных.
* **Развёртывание приложений**
  + Размещение приложений в облаке (Heroku, AWS) с учётом масштабируемости, безопасности и надёжности.

#### **Front-End Разработка**

* **HTML5 и CSS3**
  + Создание адаптивных и отзывчивых интерфейсов для различных устройств.
  + Использование современных стандартов для построения удобных пользовательских интерфейсов.
* **JavaScript и его экосистема**
  + Создание интерактивных элементов на сайте с использованием Vanilla JavaScript.
  + Разработка клиентских приложений на React или других JS-фреймворках.
  + Работа с Node.js для серверного JavaScript-программирования.

#### **Компьютерные науки и алгоритмы**

* Основы **структур данных** (массивы, списки, деревья, графы).
* **Алгоритмы**: сортировка, поиск, оптимизация.
* Работа с низкоуровневым языком **C** для изучения управления памятью и других ключевых концепций.

#### **DevOps и командная работа**

* **Системы контроля версий**
  + Использование Git для работы в команде, управления кодовой базой и автоматизации развёртывания.
* **CI/CD**
  + Настройка конвейеров автоматического тестирования, сборки и доставки приложений (Jenkins, Travis CI).
* **Agile-подходы**
  + Участие в спринт-планировании, ежедневных митингах и ретроспективах.

#### **Решение проблем и отладка**

* Использование инструментов отладки для поиска и исправления ошибок на всех уровнях стека.
* Эффективное устранение багов и оптимизация производительности приложений.

**Cloud Engineer**

Управляй облачными системами: автоматизируй процессы, используй Docker, Kubernetes и облачные платформы для создания надёжных и масштабируемых решений.

### **Технологии и языки, описанные в программе:**

#### **Технологии:**

1. **Облачные платформы:**
   * AWS (Amazon Web Services)
   * Microsoft Azure
   * Google Cloud Platform (GCP)
2. **CI/CD инструменты:**
   * Jenkins
   * GitLab CI/CD
   * Travis CI
3. **Контейнеризация:**
   * Docker
4. **Оркестрация контейнеров:**
   * Kubernetes
5. **Инфраструктура как код (IaC):**
   * Terraform
   * AWS CloudFormation
6. **Мониторинг и логирование:**
   * Prometheus
   * Grafana
   * ELK Stack (Elasticsearch, Logstash, Kibana)

#### **Языки программирования и сценарии:**

1. Python
2. Bash/Shell

### **Навыки, которыми овладеют ученики:**

#### **1. Работа с облачными платформами:**

* Настройка облачной инфраструктуры, включая виртуальные машины, базы данных, сети, и хранилища.
* Управление доступом и безопасность с использованием IAM.
* Оптимизация облачных ресурсов для обеспечения масштабируемости и высокой доступности.

#### **2. Автоматизация CI/CD процессов:**

* Создание и поддержка автоматизированных конвейеров для сборки, тестирования и развёртывания приложений.
* Интеграция CI/CD инструментов с системами контроля версий (Git).
* Настройка автоматического развертывания приложений в различных окружениях.

#### **3. Контейнеризация и управление контейнерами:**

* Упаковка приложений в Docker-контейнеры для стандартизации и удобства развёртывания.
* Использование Kubernetes для оркестрации контейнеров, их масштабирования и управления нагрузкой.
* Разработка стратегий обновлений и отказоустойчивости для контейнерных приложений.

#### **4. Инфраструктура как код (IaC):**

* Автоматизация создания и конфигурации инфраструктуры с помощью Terraform или AWS CloudFormation.
* Версионирование инфраструктурных изменений для надёжного управления ресурсами.

#### **5. Мониторинг и логирование:**

* Настройка инструментов для отслеживания производительности систем, мониторинга ресурсов и получения метрик.
* Установка и настройка логирования для диагностики и анализа проблем.
* Использование визуализационных панелей (Grafana, Kibana) для анализа данных в реальном времени.

#### **6. Скриптование и автоматизация:**

* Использование Python или Bash для автоматизации рутинных задач.
* Разработка сценариев для управления инфраструктурой, резервного копирования и мониторинга.

#### **7. Безопасность:**

* Реализация лучших практик безопасности для облачных систем: шифрование данных, защита сетей и управление доступом.
* Проведение аудитов безопасности и настройка систем для предотвращения уязвимостей.

### **C++ Engineer**

Погрузись в системное программирование: создавай операционные системы, драйверы и высокопроизводительные приложения с C++.

### **Технологии и языки, описанные в программе:**

#### **Языки программирования:**

1. **C++**
2. **C**
3. **Ruby**
4. **SQL и NoSQL**
5. **Assembly**

### **Навыки, которыми овладеют ученики:**

#### **1. Основы C++:**

* Изучение синтаксиса C++ и особенностей языка.
* Освоение объектно-ориентированного программирования (ООП), включая инкапсуляцию, наследование и полиморфизм.
* Работа с шаблонами (templates) и стандартной библиотекой C++ (STL).

#### **2. Работа с памятью и ресурсами:**

* Управление памятью: динамическое выделение, освобождение памяти, оптимизация использования ресурсов.
* Понимание работы указателей, ссылок и концепции RAII (Resource Acquisition Is Initialization).

#### **3. Многопоточность и конкурентное программирование:**

* Реализация многопоточных приложений с использованием потоков (threads) и библиотек, таких как <thread> и <mutex>.
* Использование механизмов синхронизации для предотвращения гонок данных (race conditions) и других ошибок.
* Управление параллельными процессами для повышения производительности.

#### **4. Алгоритмы и структуры данных:**

* Разработка и оптимизация алгоритмов, включая сортировки, поиск и графовые алгоритмы.
* Реализация структур данных, таких как списки, стеки, очереди, деревья и хэш-таблицы.
* Применение алгоритмов для эффективного решения задач.

#### **5. Архитектура и дизайн программного обеспечения:**

* Использование шаблонов проектирования (Design Patterns) для разработки модульного и поддерживаемого кода.
* Проектирование архитектуры приложений, обеспечивающей масштабируемость и надежность.
* Оптимизация кода для повышения производительности и снижения сложности.

#### **6. Низкоуровневое программирование:**

* Создание системного ПО: операционные системы, драйверы устройств, встроенные системы реального времени.
* Работа с языком Assembly для глубокого понимания взаимодействия с аппаратным обеспечением.
* Интерфейс с оборудованием, управление ресурсами на уровне системы.

#### **7. Базы данных:**

* Использование SQL для взаимодействия с реляционными базами данных.
* Работа с NoSQL базами данных для обработки больших объемов данных и реализации нестандартных схем хранения.

#### **8. Междисциплинарное взаимодействие:**

* Работа в команде с использованием современных методов разработки ПО.
* Совместная работа над крупными проектами, внедрение оптимизированных решений.

### **Применение навыков:**

* Разработка системного ПО: операционные системы, драйверы и встроенные системы.
* Создание высокопроизводительных приложений для финансовой сферы, видеоигр и симуляторов.
* Работа с низкоуровневыми системами, включая оптимизацию алгоритмов и взаимодействие с оборудованием.

### **Data Scientist**

Анализируй данные: используй Python и машинное обучение для создания прогнозов, рекомендаций и мощной аналитики. Узнай, как делать данные полезными!

### **Технологии и языки, описанные в программе:**

#### **Языки программирования и инструменты:**

1. **Python**
   * Библиотеки: Pandas, NumPy, scikit-learn, TensorFlow, PyTorch
2. **SQL**
3. **C**
4. **Jupyter Notebook**

### **Навыки, которыми овладеют ученики:**

#### **1. Работа с данными:**

* **Сбор данных:**
  + Извлечение данных из различных источников: базы данных, API, файловые системы.
  + Обеспечение качества и целостности данных для анализа.
* **Очистка и предобработка данных:**
  + Устранение пропусков, дублирующихся записей и выбросов.
  + Преобразование данных в подходящий формат для анализа.
* **Манипуляция данными:**
  + Использование библиотеки Pandas для фильтрации, группировки и трансформации данных.
  + Создание сводных таблиц и агрегирование данных.

#### **2. Статистический анализ:**

* **Основы статистики:**
  + Проверка гипотез, регрессионный анализ, распределения вероятностей.
  + Применение методов описательной и инференциальной статистики.
* **Инструменты:**
  + NumPy для математических операций и работы с массивами.

#### **3. Машинное обучение:**

* **Основы алгоритмов:**
  + Классификация, регрессия, кластеризация и рекомендательные системы.
  + Использование scikit-learn, PyTorch и TensorFlow для построения моделей.
* **Оптимизация моделей:**
  + Оценка производительности с помощью метрик, таких как точность, F1-score, RMSE.
  + Настройка гиперпараметров и выбор лучших моделей.

#### **4. Визуализация данных:**

* **Создание визуализаций:**
  + Использование графиков, диаграмм и тепловых карт для представления данных.
  + Интеграция визуализаций в Jupyter Notebook для создания интерактивных отчетов.
* **Коммуникация с заинтересованными сторонами:**
  + Трансформация сложных технических результатов в понятные и практичные рекомендации.

#### **5. Работа с базами данных:**

* **SQL:**
  + Запрос данных, создание и управление реляционными базами данных.
  + Оптимизация запросов для повышения производительности.

#### **6. Основы компьютерных наук:**

* Понимание структур данных, таких как списки, деревья, графы.
* Управление памятью и написание эффективного кода.
* Использование языка C для изучения низкоуровневых концепций.

#### **7. Аналитическое мышление и решение задач:**

* Формулирование гипотез и проектирование экспериментов.
* Анализ данных для получения значимых выводов.
* Построение решений, основанных на эмпирических данных.

### **Применение навыков:**

* Разработка моделей для решения бизнес-задач: прогнозирование продаж, анализ клиентов, сегментация аудитории.
* Внедрение рекомендаций, основанных на данных, для улучшения стратегий и процессов.
* Создание интерактивных дашбордов и отчетов для мониторинга метрик.

**Frontend Engineer**

Создавай интерфейсы, которые захватывают: адаптивные, интерактивные и удобные веб-приложения с React, HTML, CSS и JavaScript.

### **Технологии и языки, описанные в программе:**

#### **Языки программирования и инструменты:**

1. **React.js**
2. **Redux**
3. **JavaScript (ES6+) и другие библиотеки**
4. **HTML5**
5. **CSS3**
6. **Bootstrap**
7. **Material-UI**
8. **Ruby, Ruby on Rails**
9. **C**
10. **SQL**

### **Навыки, которыми овладеют ученики:**

#### **1. Основы разработки пользовательского интерфейса:**

* **React.js:**
  + Работа с компонентами React и JSX-синтаксисом.
  + Управление состоянием приложений с использованием хуков (Hooks), контекстного API и библиотек, таких как Redux и MobX.
  + Реализация декларативного подхода к созданию интерфейсов.
* **HTML5 и CSS3:**
  + Семантическая разметка для структурированных и доступных веб-страниц.
  + Создание современных макетов с использованием Flexbox и Grid.
* **Фреймворки и библиотеки для стилей:**
  + Использование Bootstrap и Material-UI для создания адаптивных и визуально привлекательных интерфейсов.

#### **2. Создание адаптивных и кроссбраузерных интерфейсов:**

* **Адаптивный дизайн:**
  + Работа с медиазапросами для поддержки различных размеров экранов и устройств.
  + Оптимизация макетов для мобильных устройств.
* **Кроссбраузерная совместимость:**
  + Устранение различий в поведении браузеров и обеспечение корректного отображения во всех популярных браузерах.

#### 

#### **3. Взаимодействие с backend:**

* **Интеграция с API:**
  + Получение данных с серверов и их обработка для отображения в интерфейсе.
  + Работа с форматами данных, такими как JSON, и их обработка через Fetch API или Axios.
* **Работа с backend-технологиями:**
  + Ruby on Rails и SQL для взаимодействия с серверной частью приложений.

#### **4. Оптимизация производительности:**

* **Техники оптимизации:**
  + Code splitting и lazy loading для уменьшения времени загрузки.
  + Кэширование на стороне клиента для улучшения скорости работы приложения.
* **Инструменты:**
  + Использование Webpack и Babel для сборки и оптимизации приложений.

#### **5. Тестирование и отладка:**

* **Поиск и исправление ошибок:**
  + Использование инструментов разработчика браузера для отладки.
  + Отслеживание и устранение проблем с производительностью и функциональностью.

#### **6. Основы проектирования пользовательского опыта (UX):**

* **Удобство и доступность:**
  + Создание интуитивно понятных интерфейсов.
  + Реализация стандартов доступности, чтобы интерфейсы были удобны для всех пользователей, включая людей с ограниченными возможностями.
* **Визуальная эстетика:**
  + Применение принципов дизайна для создания привлекательных и функциональных интерфейсов.

#### **7. Командная работа и CI/CD:**

* **Agile-методологии:**
  + Участие в планировании спринтов, код-ревью и ретроспективах.
* **CI/CD:**
  + Настройка автоматических процессов тестирования и развёртывания приложений.

### 

### 

### **Применение навыков:**

* Разработка современных веб-приложений с интерактивными интерфейсами для различных устройств.
* Интеграция пользовательских интерфейсов с серверной частью для создания полноценных приложений.
* Создание адаптивных и оптимизированных веб-страниц для улучшения пользовательского опыта.

**Go Software Engineer**

Стань экспертом по Go: разрабатывай быстрые серверы, микросервисы и эффективные инструменты для современных систем.

### **Технологии и языки, описанные в программе:**

#### **Языки программирования:**

1. **Go (Golang)**
2. **C**
3. **JavaScript**
4. **SQL**
5. **Python**

### **Навыки, которыми овладеют ученики:**

#### **1. Основы Go и программирования:**

* **Go Syntax и Idiomatic Patterns:**
  + Освоение синтаксиса Go, включая строгую типизацию, структурированность и стандартные практики написания кода.
  + Понимание идиоматического подхода Go к разработке для создания чистого и поддерживаемого кода.
* **Concurrency:**
  + Работа с goroutines для создания лёгких потоков.
  + Использование каналов (channels) для безопасного и эффективного обмена данными между потоками.
  + Управление синхронизацией с помощью встроенных примитивов, таких как sync.Mutex.
* **Оптимизация производительности:**
  + Улучшение скорости и эффективности приложений на Go за счёт анализа производительности и оптимизации кода.
  + Минимизация использования памяти и сокращение времени компиляции.

#### **2. Разработка серверной части:**

* **Разработка бэкенд-сервисов и микросервисов:**
  + Использование стандартной библиотеки Go для обработки HTTP-запросов и работы с API.
  + Использование фреймворков, таких как Gin или Echo, для упрощения разработки RESTful приложений.
* **Сетевые приложения:**
  + Создание высокопроизводительных серверов TCP/IP и HTTP.
  + Разработка серверов и клиентов для сетевого взаимодействия.

#### **3. Облачные технологии и распределённые системы:**

* **Cloud-Native Разработка:**
  + Создание масштабируемых распределённых систем с использованием Go.
  + Интеграция с Kubernetes и Docker для управления контейнеризированными приложениями.
* **Разработка надёжных систем:**
  + Обеспечение отказоустойчивости и масштабируемости приложений.
  + Использование стратегий распределённого кеширования и управления состоянием.

#### **4. Создание CLI и утилит:**

* Разработка удобных и кроссплатформенных командных утилит.
* Использование Go для быстрой компиляции и создания исполняемых файлов для различных операционных систем.

#### **5. Основы проектирования программного обеспечения:**

* Применение шаблонов проектирования для построения модульных и расширяемых приложений.
* Создание тестируемого и легко поддерживаемого кода.

#### **6. Работа с базами данных:**

* **SQL:**
  + Подключение и работа с реляционными базами данных через Go.
  + Оптимизация запросов для повышения производительности.
* **NoSQL:**
  + Подключение и работа с нереляционными базами данных (например, Redis, MongoDB).

#### **7. Интеграция и сообщество:**

* **Открытый исходный код:**
  + Вклад в существующие проекты на Go.
  + Расширение фреймворков, библиотек и инструментов для сообщества.

#### **8. Дополнительные языки и технологии:**

* Использование Python для создания утилит и скриптов.
* Интеграция Go-приложений с фронтендом, разработанным на JavaScript.
* Создание многозадачных решений на C для более глубокого понимания системного программирования.

### **Применение навыков:**

* Разработка высоконагруженных бэкенд-систем для веб-приложений.
* Создание масштабируемых распределённых облачных систем.
* Интеграция микросервисов и управление контейнерами.
* Разработка эффективных командных утилит для автоматизации задач.

### **Machine Learning Engineer**

Освой искусственный интеллект: создавай модели машинного обучения и AI-решения для анализа данных, текстов и изображений.

### **Технологии и языки, описанные в программе:**

#### **Языки программирования:**

1. **Python**
   * Библиотеки: TensorFlow, PyTorch, Scikit-learn, NumPy, Pandas, Matplotlib, Seaborn
2. **C/C++**
3. **SQL**
4. **NoSQL**

### **Навыки, которыми овладеют ученики:**

#### **1. Основы машинного обучения:**

* **Алгоритмы машинного обучения:**
  + **Supervised Learning:** Классификация, регрессия.
  + **Unsupervised Learning:** Кластеризация, понижение размерности.
  + **Reinforcement Learning:** Обучение с подкреплением для задач оптимизации и последовательного принятия решений.
* **Глубокое обучение:**
  + Использование TensorFlow и PyTorch для создания нейронных сетей.
  + Реализация моделей для компьютерного зрения, обработки естественного языка (NLP) и временных рядов.

#### **2. Работа с данными:**

* **Предобработка данных:**
  + Очистка, нормализация и обработка отсутствующих значений.
  + Преобразование данных через методы feature engineering и dimensionality reduction (PCA, t-SNE).
* **Анализ данных:**
  + Использование Pandas и NumPy для обработки данных.
  + Визуализация данных с помощью Matplotlib и Seaborn для выявления закономерностей, трендов и аномалий.

#### **3. Разработка и оптимизация моделей:**

* **Обучение и настройка моделей:**
  + Выбор гиперпараметров, кросс-валидация и методы регуляризации.
  + Оценка моделей с использованием метрик, таких как точность, F1-скор, ROC-AUC.
* **Построение конвейеров машинного обучения:**
  + Автоматизация процессов обработки данных и обучения моделей.
  + Использование платформ распределённого вычисления, таких как Apache Spark или Dask, для масштабируемости.

#### **4. Инженерные навыки:**

* **Программирование:**
  + Профессиональное владение Python для создания и тестирования моделей.
  + Использование C++ для высокопроизводительных вычислений.
* **Модульное и тестируемое программирование:**
  + Создание легко поддерживаемого кода с применением лучших инженерных практик.

#### **5. Специализация по направлениям:**

* **Обработка естественного языка (NLP):**
  + Работа с текстовыми данными, включая обработку текста, векторизацию, создание моделей для задач, таких как классификация текста или генерация текста.
* **Компьютерное зрение:**
  + Использование сверточных нейронных сетей (CNN) для анализа изображений и видео.
* **Анализ временных рядов:**
  + Прогнозирование временных данных с помощью рекуррентных нейронных сетей (RNN) или трансформеров.
* **Рекомендательные системы:**
  + Разработка алгоритмов для персонализированных рекомендаций.

#### **6. Работа с базами данных:**

* **SQL и NoSQL:**
  + Создание запросов для получения данных из реляционных баз.
  + Обработка больших данных с использованием NoSQL-баз (например, MongoDB).

#### **7. Бизнес-аналитика и взаимодействие с командами:**

* Формулирование бизнес-задач и перевод их в задачи машинного обучения.
* Оценка влияния решений на бизнес-цели и их объяснение заинтересованным сторонам.

### **Применение навыков:**

* Построение систем рекомендаций, прогнозирования продаж или анализа поведения пользователей.
* Разработка моделей для компьютерного зрения (например, системы распознавания лиц или объектов).
* Работа с большими данными для создания аналитических дашбордов и прогнозов.
* Внедрение инновационных AI-решений в бизнесе, таких как автоматизация процессов или интеллектуальный анализ данных.

### **Rust Software Engineer**

Изучи Rust — язык будущего: разрабатывай безопасные, высокопроизводительные системы и приложения. Отличный выбор для настоящих профи!

### **Технологии и языки, описанные в программе:**

#### **Языки программирования:**

1. **Rust**
2. **C**
3. **Ruby**
4. **JavaScript**
5. **Assembly**

### **Навыки, которыми овладеют ученики:**

#### **1. Основы Rust и его особенности:**

* **Rust Syntax и Ownership Model:**
  + Освоение ключевых особенностей языка, включая строгую типизацию, систему владения (ownership), заимствования (borrowing) и времени жизни (lifetimes).
  + Создание безопасного кода без необходимости в сборке мусора.
* **Безопасность памяти:**
  + Разработка приложений с защитой от ошибок, связанных с доступом к памяти, таких как сегментационные ошибки или утечки памяти.

#### **2. Многопоточность и конкурентное программирование:**

* **Concurrency:**
  + Использование примитивов Rust, таких как async/await, futures, и многопоточность для создания эффективных конкурентных приложений.
  + Решение задач параллелизма для увеличения производительности.
* **Асинхронное программирование:**
  + Разработка высоконагруженных серверов с использованием асинхронных подходов.

#### **3. Системное программирование:**

* **Низкоуровневый доступ:**
  + Работа с системными API, управление памятью, взаимодействие с оборудованием.
  + Разработка операционных систем, драйверов устройств и встроенных систем.
* **Оптимизация производительности:**
  + Поиск и устранение узких мест в производительности программ.
  + Оптимизация использования ресурсов, таких как процессорное время и память.

#### **4. Разработка приложений:**

* **Высокопроизводительные сетевые приложения:**
  + Создание серверов, реализация сетевых протоколов.
  + Разработка приложений для распределённых систем и сервисов.
* **Web-разработка:**
  + Создание бэкенд-сервисов на Rust с использованием фреймворков, таких как Actix или Rocket.
  + Интеграция с фронтендом, написанным на JavaScript.
* **Игровые движки и криптографические библиотеки:**
  + Работа над производительными графическими движками.
  + Реализация безопасных и эффективных криптографических алгоритмов.

#### **5. Алгоритмы и структуры данных:**

* Понимание алгоритмов сортировки, поиска, работы с графами и деревьями.
* Реализация эффективных структур данных для создания масштабируемых решений.

#### **6. Проектирование и оптимизация:**

* **Шаблоны проектирования:**
  + Применение лучших практик для создания модульного, поддерживаемого и расширяемого кода.
* **Отладка и устранение ошибок:**
  + Умение находить сложные баги, используя инструменты отладки и тестирования, такие как cargo test и clippy.

#### **7. Открытые проекты и работа с сообществом:**

* **Open Source:**
  + Вклад в библиотеки и фреймворки, написанные на Rust.
  + Работа с инструментами, такими как Cargo, для управления проектами.

#### **8. Дополнительные языки:**

* **C и Assembly:**
  + Интеграция с низкоуровневыми компонентами и библиотеками.
  + Изучение основ системного программирования.
* **Ruby и JavaScript:**
  + Работа с дополнительными языками для интеграции разных частей приложений.

### **Применение навыков:**

* Разработка системного ПО: драйверы, операционные системы, встроенные устройства.
* Создание высокопроизводительных веб-сервисов и микросервисов.
* Разработка игр, графических движков и криптографических библиотек.
* Работа в облачных средах и распределённых системах.