МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ГЕОДЕЗИИ И КАРТОГРАФИИ
(МИИГАиК)

Факультет геоинформатики и информационной безопасности
Ни томский государственный университет

**ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ**

Сертификат: 5BFEE1D286E9C95A0C0D2C611E434BD5
Владелец: Степанченко Алексей Леонидович
Действителен: с 23.09.2026 по 17.12.2027

«УТВЕРЖДАЮ»
Проректор
А. Л. Степанченко
24 марта 2025 г.

# ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ПО ДИСЦИПЛИНЕ (МОДУЛЮ) «Б1.О.05 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И СКРИПТОВЫЕ ЯЗЫКИ»

Уровень высшего образования: магистратура

Направление подготовки: 09.04.02 Информационные системы и технологии

Направленность (профиль) подготовки: Использование открытого программного обеспечения для разработки геосервисов

Квалификация (степень) выпускника: магистр

Форма обучения: очная

Год набора (приема на обучение): 2025

Срок получения образования: 1 года 10 месяца(-ев)

Объем: в зачетных единицах: 4 з.е.
в академических часах: 144 ак.ч.

Москва, 2025

**Разработчики:**
Старший преподаватель кафедры теоретических основ информатики Овсянников М. С.

Фонд оценочных средств по дисциплине (модулю) составлен в соответствии с требованиями ФГОС ВО по направлению подготовки 09.04.02 Информационные системы и технологии, утвержденного приказом Минобрнауки от 19.09.2017 № 917

Согласование и утверждение

| № | Подразделение | Ответственное лицо | ФИО | Виза | Дата, протокол (при наличии) |
|---|---|---|---|---|---|
| 1 | факультет Геоинформатики и информационной безопасности | Председатель ученого совета факультета | Матерухин А. В. | Рассмотрено | 14.10.2024, № 10-24 |

## 1. Планируемые результаты обучения, соотнесенные с планируемыми результатами освоения образовательной программы

ПК-1 Способен управлять рисками разработки программного обеспечения

ПК-1.1 Идентифицирует и оценивает риски разработки программного обеспечения

*Знать:*

ПК-1.1/Зн1 Классификацию и источники рисков разработки программного обеспечения

ПК-1.1/Зн2 Методы идентификации, качественной и количественной оценки рисков

ПК-1.1/Зн3 Принципы ведения реестра рисков и назначения владельцев рисков

*Уметь:*

ПК-1.1/Ум1 Выявлять риски на этапах разработки, внедрения и сопровождения программного обеспечения

ПК-1.1/Ум2 Оценивать вероятность и последствия реализации рисков

ПК-1.1/Ум3 Формировать и актуализировать реестр рисков программного проекта

*Владеть:*

ПК-1.1/Нв1 Навыками идентификации и документирования рисков разработки программного обеспечения

ПК-1.1/Нв2 Навыками комплексной оценки рисков программного проекта

ПК-1.1/Нв3 Навыками оценки рисков, связанных с несовместимостью версий интерпретаторов, компиляторов и сторонних библиотек в геосервисах

ПК-1.2 Планирует меры реагирования и осуществляет мониторинг рисков разработки программного обеспечения

*Знать:*

ПК-1.2/Зн1 Стратегии реагирования на угрозы и возможности программного проекта

ПК-1.2/Зн2 Методы мониторинга рисков и контроля эффективности мер реагирования

ПК-1.2/Зн3 Подходы к резервированию ресурсов и планированию предупреждающих мероприятий

*Уметь:*

ПК-1.2/Ум1 Разрабатывать план реагирования на риски разработки программного обеспечения

ПК-1.2/Ум2 Выбирать меры снижения, передачи, принятия или исключения рисков

ПК-1.2/Ум3 Контролировать состояние рисков и корректировать меры реагирования

*Владеть:*

ПК-1.2/Нв1 Навыками планирования и реализации мер реагирования на риски

ПК-1.2/Нв2 Навыками мониторинга рисков и актуализации проектной документации

ПК-1.2/Нв3 Навыками практического применения инструментов управления рисками в процессе индустриальной разработки геосервисов на конкретном рабочем месте

## 2. Показатели и критерии оценивания компетенций

| Планируемые результаты освоения программы | Уровень овладения | Критерии оценивания (перечень сформированных ИДК/ЗУВ по уровням освоения) |
|---|---|---|
| ПК-1 Способен управлять рисками разработки программного обеспечения | Повышенный | ИДК ПК-1.1 Идентифицирует и оценивает риски разработки программного обеспечения<br><br>Знать:<br>ПК-1.1/Зн1 Классификацию и источники рисков разработки программного обеспечения<br>ПК-1.1/Зн2 Методы идентификации, качественной и количественной оценки рисков<br>ПК-1.1/Зн3 Принципы ведения реестра рисков и назначения владельцев рисков<br>Уметь:<br>ПК-1.1/Ум1 Выявлять риски на этапах разработки, внедрения и сопровождения программного обеспечения<br>ПК-1.1/Ум2 Оценивать вероятность и последствия реализации рисков<br>ПК-1.1/Ум3 Формировать и актуализировать реестр рисков программного проекта<br>Владеть:<br>ПК-1.1/Нв1 Навыками идентификации и документирования рисков разработки программного обеспечения<br>ПК-1.1/Нв2 Навыками комплексной оценки рисков программного проекта<br>ПК-1.1/Нв3 Навыками оценки рисков, связанных с несовместимостью версий интерпретаторов, компиляторов и сторонних библиотек в геосервисах<br><br>ИДК ПК-1.2 Планирует меры реагирования и осуществляет мониторинг рисков разработки программного обеспечения<br><br>Знать:<br>ПК-1.2/Зн2 Методы мониторинга рисков и контроля эффективности мер реагирования<br>ПК-1.2/Зн3 Подходы к резервированию ресурсов и планированию предупреждающих мероприятий<br>ПК-1.2/Зн1 Стратегии реагирования на угрозы и возможности программного проекта<br>Уметь:<br>ПК-1.2/Ум1 Разрабатывать план реагирования на риски разработки программного обеспечения<br>ПК-1.2/Ум2 Выбирать меры снижения, передачи, принятия или исключения рисков<br>ПК-1.2/Ум3 Контролировать состояние рисков и корректировать меры реагирования<br>Владеть:<br>ПК-1.2/Нв1 Навыками планирования и реализации мер реагирования на риски |

| | | |
|---|---|---|
| | | ПК-1.2/Нв2 Навыками мониторинга рисков и актуализации проектной документации<br>ПК-1.2/Нв3 Навыками практического применения инструментов управления рисками в процессе индустриальной разработки геосервисов на конкретном рабочем месте |
| | Базовый | ИДК ПК-1.1 Идентифицирует и оценивает риски разработки программного обеспечения<br><br>Знать:<br>ПК-1.1/Зн3 Принципы ведения реестра рисков и назначения владельцев рисков<br>ПК-1.1/Зн2 Методы идентификации, качественной и количественной оценки рисков<br>ПК-1.1/Зн1 Классификацию и источники рисков разработки программного обеспечения<br>Уметь:<br>ПК-1.1/Ум3 Формировать и актуализировать реестр рисков программного проекта<br>ПК-1.1/Ум2 Оценивать вероятность и последствия реализации рисков<br>ПК-1.1/Ум1 Выявлять риски на этапах разработки, внедрения и сопровождения программного обеспечения<br><br>ИДК ПК-1.2 Планирует меры реагирования и осуществляет мониторинг рисков разработки программного обеспечения<br><br>Знать:<br>ПК-1.2/Зн3 Подходы к резервированию ресурсов и планированию предупреждающих мероприятий<br>ПК-1.2/Зн2 Методы мониторинга рисков и контроля эффективности мер реагирования<br>ПК-1.2/Зн1 Стратегии реагирования на угрозы и возможности программного проекта<br>Уметь:<br>ПК-1.2/Ум1 Разрабатывать план реагирования на риски разработки программного обеспечения<br>ПК-1.2/Ум3 Контролировать состояние рисков и корректировать меры реагирования<br>ПК-1.2/Ум2 Выбирать меры снижения, передачи, принятия или исключения рисков |
| | Пороговый | ИДК ПК-1.1 Идентифицирует и оценивает риски разработки программного обеспечения<br><br>Знать:<br>ПК-1.1/Зн2 Методы идентификации, качественной и количественной оценки рисков<br>ПК-1.1/Зн3 Принципы ведения реестра рисков и назначения владельцев рисков<br>ПК-1.1/Зн1 Классификацию и источники рисков разработки программного обеспечения |

| | | |
|---|---|---|
| | | ИДК ПК-1.2 Планирует меры реагирования и осуществляет мониторинг рисков разработки программного обеспечения<br><br>Знать:<br>ПК-1.2/Зн3 Подходы к резервированию ресурсов и планированию предупреждающих мероприятий<br>ПК-1.2/Зн2 Методы мониторинга рисков и контроля эффективности мер реагирования<br>ПК-1.2/Зн1 Стратегии реагирования на угрозы и возможности программного проекта |

## 3. Шкала оценивания

| Оценка | Критерии оценивания (перечень сформированных ИДК/ЗУВ) |
|---|---|
| Отлично | ПК-1.1 Идентифицирует и оценивает риски разработки программного обеспечения<br>ПК-1.1/Зн1 Классификацию и источники рисков разработки программного обеспечения<br>ПК-1.1/Зн2 Методы идентификации, качественной и количественной оценки рисков<br>ПК-1.1/Зн3 Принципы ведения реестра рисков и назначения владельцев рисков<br>ПК-1.1/Ум1 Выявлять риски на этапах разработки, внедрения и сопровождения программного обеспечения<br>ПК-1.1/Ум2 Оценивать вероятность и последствия реализации рисков<br>ПК-1.1/Ум3 Формировать и актуализировать реестр рисков программного проекта<br>ПК-1.1/Нв1 Навыками идентификации и документирования рисков разработки программного обеспечения<br>ПК-1.1/Нв2 Навыками комплексной оценки рисков программного проекта<br>ПК-1.1/Нв3 Навыками оценки рисков, связанных с несовместимостью версий интерпретаторов, компиляторов и сторонних библиотек в геосервисах |

| | |
|---|---|
| | ПК-1.2 Планирует меры реагирования и осуществляет мониторинг рисков разработки программного обеспечения<br>ПК-1.2/Зн1 Стратегии реагирования на угрозы и возможности программного проекта<br>ПК-1.2/Зн2 Методы мониторинга рисков и контроля эффективности мер реагирования<br>ПК-1.2/Зн3 Подходы к резервированию ресурсов и планированию предупреждающих мероприятий<br>ПК-1.2/Ум1 Разрабатывать план реагирования на риски разработки программного обеспечения<br>ПК-1.2/Ум2 Выбирать меры снижения, передачи, принятия или исключения рисков<br>ПК-1.2/Ум3 Контролировать состояние рисков и корректировать меры реагирования<br>ПК-1.2/Нв1 Навыками планирования и реализации мер реагирования на риски<br>ПК-1.2/Нв2 Навыками мониторинга рисков и актуализации проектной документации<br>ПК-1.2/Нв3 Навыками практического применения инструментов управления рисками в процессе индустриальной разработки геосервисов на конкретном рабочем месте |
| Хорошо | ПК-1.1 Идентифицирует и оценивает риски разработки программного обеспечения<br>ПК-1.1/Зн1 Классификацию и источники рисков разработки программного обеспечения<br>ПК-1.1/Зн2 Методы идентификации, качественной и количественной оценки рисков<br>ПК-1.1/Зн3 Принципы ведения реестра рисков и назначения владельцев рисков<br>ПК-1.1/Ум1 Выявлять риски на этапах разработки, внедрения и сопровождения программного обеспечения<br>ПК-1.1/Ум2 Оценивать вероятность и последствия реализации рисков<br>ПК-1.1/Ум3 Формировать и актуализировать реестр рисков программного проекта<br>ПК-1.2 Планирует меры реагирования и осуществляет мониторинг рисков разработки программного обеспечения<br>ПК-1.2/Зн1 Стратегии реагирования на угрозы и возможности программного проекта<br>ПК-1.2/Зн2 Методы мониторинга рисков и контроля эффективности мер реагирования<br>ПК-1.2/Зн3 Подходы к резервированию ресурсов и планированию предупреждающих мероприятий<br>ПК-1.2/Ум1 Разрабатывать план реагирования на риски разработки программного обеспечения<br>ПК-1.2/Ум2 Выбирать меры снижения, передачи, принятия или исключения рисков<br>ПК-1.2/Ум3 Контролировать состояние рисков и корректировать меры реагирования |

| | |
|---|---|
| Удовлетворительно | ПК-1.1 Идентифицирует и оценивает риски разработки программного обеспечения<br>ПК-1.1/Зн1 Классификацию и источники рисков разработки программного обеспечения<br>ПК-1.1/Зн2 Методы идентификации, качественной и количественной оценки рисков<br>ПК-1.1/Зн3 Принципы ведения реестра рисков и назначения владельцев рисков<br>ПК-1.2 Планирует меры реагирования и осуществляет мониторинг рисков разработки программного обеспечения<br>ПК-1.2/Зн1 Стратегии реагирования на угрозы и возможности программного проекта<br>ПК-1.2/Зн2 Методы мониторинга рисков и контроля эффективности мер реагирования<br>ПК-1.2/Зн3 Подходы к резервированию ресурсов и планированию предупреждающих мероприятий |
| Неудовле-творительно | Результаты обучения не сформированы на достаточном уровне |

## 4. Контрольные мероприятия по дисциплине

| Вид контроля | Форма контроля/Оценочное средство |
|---|---|
| Текущий контроль | Практическая работа<br>Тестирование |
| Промежуточная аттестация | Экзамен |

| № п/п | Наименование раздела | Контролируемые ИДК | Вид контроля/ используемые оценочные материалы | |
|---|---|---|---|---|
| | | | Текущий | Промежут. аттестация |
| 1 | Теоретические основы разработки языков и компиляции | ПК-1.1 | Тестирование | Экзамен |
| 2 | Изучение современных скриптовых и компилируемых языков программирования | ПК-1.2 | Практическая работа<br>Тестирование | Экзамен |

## 5. Порядок проведения промежуточной аттестации

Порядок проведения промежуточной аттестации регламентируется локальными нормативными актами Университета:

- Положением о формах, периодичности, порядке текущего контроля успеваемости и промежуточной аттестации обучающихся по основным образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры в федеральном государственном бюджетном образовательном учреждении высшего образования «Московский государственный университет геодезии и картографии», п.6 «Организация и проведение промежуточной аттестации».
- Положением о порядке осуществления образовательного процесса, проведения текущего контроля успеваемости и промежуточной аттестации обучающихся с применением электронного обучения и дистанционных образовательных технологий в федеральном государственном бюджетном образовательном учреждении высшего образования «Московский государственный университет геодезии и картографии», п.8 «Проведение промежуточной аттестации с применением ЭО и ДОТ».

*Промежуточная аттестация: Экзамен, 2 семестр.*

Экзамен оценивается по пятибалльной шкале:

- зачтено с оценкой «отлично» выставляется, если обучающийся имеет глубокие, исчерпывающие знания в объеме освоенной программы; знание основной (обязательной) литературы; дает правильные и уверенные ответы, свидетельствующие о наличии твердых знаний и навыков в использовании технических средств; полное, четкое, грамотное и логически стройное изложение материала; демонстрирует свободное применение теоретических знаний при анализе практических вопросов;
- зачтено с оценкой «хорошо» выставляется обучающемуся в случае соблюдения тех же требований, но наличии в ответе обучающегося по некоторым перечисленным показателям недостатков принципиального характера, что вызывает замечания или поправки преподавателя;
- зачтено с оценкой «удовлетворительно» выставляется обучающемуся в случае соблюдения тех же требований, но в случае наличия в ответе ошибок, необходимости существенной помощи в виде поправок и наводящих вопросов преподавателя;
- не зачтено («не удовлетворительно») – обучающийся не смог ответить на 2/3 вопросов; не справился с практическим заданием или выполнено менее 50 % практических заданий.

## 6. Оценочные материалы текущего контроля

***Раздел 1. Теоретические основы разработки языков и компиляции***

*Контролируемые ИДК: ПК-1.1*

*Тема 1.1. Общее введение в теорию компиляции.*

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Чем отличается компилятор от интерпретатора?
2. Что такое JIT-компиляция?
3. Какую роль выполняет ассемблер в процессе разработки?

*Тема 1.2. Языки программирования, парадигмы и классификация.*

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Что такое грамматика языка программирования и какова роль иерархии Хомского?
2. Назовите основные парадигмы программирования.
3. Что означает, что язык является мультипарадигменным? Приведите пример.

*Тема 1.3. Архитектура языков программирования.*

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Чем статическая типизация отличается от динамической?
2. Что такое «сборка мусора»?
3. Что такое «синтаксический сахар» в языке программирования?

*Тема 1.4. Обзор современных языков и их ключевые особенности.*

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Каковы ключевые особенности языка Rust в сравнении с C++?
2. Для каких задач чаще всего используется язык Go (Golang)?

*Тема 1.5. Введение в лексический и синтаксический анализ как фазы компилятора.*

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Что такое лексема и токен в контексте лексического анализа?

2. В чем разница между нисходящим (top-down) и восходящим (bottom-up) синтаксическим анализом?

*Тема 1.6. Семантический анализ и промежуточные языки представлений.*

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Для чего используется промежуточное представление (IR) в компиляторе?

2. Что такое байт-код и для чего используются виртуальные машины?

*Тема 1.7. Оптимизация.*

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Что такое «граф потока управления» (CFG) и для чего он используется при оптимизации?

2. Приведите пример оптимизации на уровне циклов.

*Тема 1.8. Низкоуровневая оптимизация и кодогенерация.*

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Какова основная задача кодогенерации в компиляторе?

2. В чем отличие регистровой машины от стековой при выполнении кода?

***Раздел 2. Изучение современных скриптовых и компилируемых языков программирования***

*Контролируемые ИДК: ПК-1.2*

*Тема 2.1. Скриптовый язык Python 3.*

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Что такое list comprehension в Python? Приведите простейший пример.

2. Как в Python обрабатываются исключения?

3. Что такое property (свойство) в классах Python и зачем оно нужно?

Форма контроля/оценочное средство: Практическая работа

Вопросы/Задания:

1. Wget на минималках

Используя примитивы многозадачности и синхронизации реализовать скачивание файла по http с выводом количества уже принятых байтов каждую секунду. Для реализации рекомендуется использовать библиотеку http.client. Программа получает в качестве аргумента командной строки URL файла и скачивает его в текущую папку сохраняя исходное имя. Каждую секунду по таймеру выводится размер принятых данных. Язык для реализации – Python (допускается другой скриптовый язык кроме JS).

2. Сервер времени

Разработать веб-приложение (например через интерфейс wsgi), реализующее аналог сервиса time.is и предоставляющее работу с временными зонами на базе библиотеки pytz. Необходимо реализовать следующие конечные пути:

1) Веб приложение по запросу GET / отдает текущее время во временной зоне сервера в формате html

2) по запросу GET /<tz name> отдает текущее время в запрошенной зоне в формате html
3) по запросу POST /api/v1/time - отдает в формате json текущее время в зоне определенной параметром tz(если нет - то зона сервера)
4) по запросу POST /api/v1/date- отдает в формате json текущую дату в зоне определенной параметром tz(если нет - то зона сервера)
5) по запросу POST /api/v1/datediff- отдает в формате json время между датами определенными параметрами start и end (каждый - это json формата {"date":"12.20.2021 22:21:05", "tz": "EST"} или {"date":"12:30pm 2020-12-01", "tz": "Europe/Moscow"}, tz - опциональна).
Дополнительные требования к реализации:
1) Запрещено использовать веб-фреймворки
2) Код лабораторной должен представлять из себя один (!) файл
3) К решению должен прилагаться тест с использованием библиотеки requests с тестированием каждого апи
Язык для реализации – Python (допускается другой скриптовый язык кроме JS).

*Тема 2.2. Скриптовый язык Javascript.*

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Чем отличается undefined от null в JavaScript?
2. Что такое прототипное наследование в JavaScript?
3. Для чего используется функция fetch в JavaScript?

Форма контроля/оценочное средство: Практическая работа

Вопросы/Задания:

1. Мобильное приложение (только PWA) - менеджер паролей

Необходимо разработать прогрессивное веб-приложение (PWA), которое возможно установить, как мобильное приложение. Требуется реализовать минимальный комплект из манифеста, service-worker.js, иконки и html разметки с кодом (или отдельный js файл с кодом). Реализовывать отдельный сервер backend не требуется. Приложение должно представлять из себя менеджер паролей с локальным хранением данных в localStorage(и/или в IndexedDB) и возможностью сгенерировать пароль с требуемой сложностью. Таким образом, веб приложение хранит данные в браузере пользователя, без бэкенда. При открытии пользователю должен показаться список сохраненных паролей и форма для добавления нового (логин, пароль, url). Язык для реализации – Javascript или TypeScript.

*Тема 2.3. Компилируемый язык с сборщиком мусора Go (golang).*

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Что такое горутина (goroutine) в Go?
2. Как устроены каналы в Go и для чего они используются?
3. Что такое select statement в Go?

Форма контроля/оценочное средство: Практическая работа

Вопросы/Задания:

1. Секретный протокол

Реализовать на языке Go tcp клиент и сервер (общее приложение, режим определяется параметрами), способное установить сокет соединение и последовательно обмениваться сообщениями. На каждой стороне реализуется алгоритм переменных ключей. На каждом шаге обмена вычисляется следующий ключ и сравнивается с полученным от второй стороны.
Шаг 1. Установление соединения. Клиент подключается к серверу и передает стартовую строку и первый ключ.
Шаг 2. Сервер на основе строки и ключа генерирует новый ключ и отдает его клиенту.
Шаг 3. Клиент сравнивает полученный ключ со следующим ключом, и, если все успешно,

создает новый ключ и отправляет следующее сообщение на сервер.
Шаг 4..10 - аналогично
На каждом шагу приложение должно выводить в консоли текущий статус, текущий ключ и отправленное/полученное сообщение. Приложение представляет из себя аналог консольного чата и позволяет вводить сообщение и отображать ответ.
При запуске программа должна принимать два параметра командной строки:
1) порт - режим сервера или ip:port - режим клиента
2) -n 100 - кол-во одновременных подключений
Язык для реализации – Go.

*Тема 2.4. Компилируемый язык Rust.*

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Объясните концепцию владения в Rust.
2. Что такое заимствование в Rust?

Форма контроля/оценочное средство: Практическая работа

Вопросы/Задания:

1. Wget на минималках

Используя примитивы многозадачности и синхронизации реализовать скачивание файла по http(s) с выводом количества уже принятых байтов каждую секунду. Для реализации рекомендуется использовать раздвоения потоков данных на базе TeeReader. Программа получает в качестве аргумента командной строки URL файла и скачивает его в текущую папку сохраняя исходное имя. Каждую секунду (по таймеру) выводится размер принятых данных. Язык для реализации – Rust.

## 7. Оценочные материалы промежуточной аттестации

*2 семестр, Экзамен*
*Контролируемые ИДК: ПК-1.1 ПК-1.2*
Вопросы/Задания:

1. Классификация компиляторов по количеству проходов.
2. Различия между синтаксическим и семантическим анализом.
3. Методы устранения левой рекурсии в грамматиках.
4. Построение таблиц синтаксического анализатора: LL(k) против LR(k).
5. Промежуточное представление кода: тетрады и триады.
6. Статический анализ типов и механизмы вывода типов.
7. Оптимизация хвостовой рекурсии и её реализация.
8. Алгоритмы распределения регистров на основе раскраски графа.
9. Алгоритмы сборки мусора: mark-and-sweep и copying.
10. Сравнение AOT-компиляции и JIT-компиляции.

11. Дескрипторы (descriptors) в Python и реализация property.

12. Метапрограммирование с помощью метаклассов в Python.

13. Глубокое копирование объектов в Python: copy.deepcopy и его особенности.

14. Асинхронное программирование в Python: asyncio, корутины и event loop.

15. Внутреннее устройство словарей (hash table) в CPython.

16. Замыкания и лексическое окружение в JavaScript.

17. Event Loop: микро- и макрозадачи в JavaScript.

18. Применение Symbol в JavaScript как уникальных идентификаторов.

19. Прокси и рефлексия для перехвата операций.

20. Цепочки промисов и обработка ошибок в JavaScript.

21. Интерфейсы в Go: утиная типизация и пустые интерфейсы.

22. Планировщик горутин и модель M:N (потоки ОС и горутины).

23. Пакет context в Go для управления временем жизни горутин.

24. Паника и recover в Go как механизм обработки исключительных ситуаций.

25. Каналы с буфером и без буфера: различия в синхронизации.

26. Времена жизни (lifetimes) в сложных структурах данных Rust.

27. Смарт-указатели: Box, Rc, Arc, RefCell и их назначение.

28. Гарантии потокобезопасности: типажи Send и Sync в Rust.

29. Макросы в Rust: декларативные (macro_rules!) и процедурные.

30. FFI (Foreign Function Interface) и безопасность вызовов C из Rust.