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
15 мая 2026 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ) «Б1.О.04 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И СКРИПТОВЫЕ ЯЗЫКИ»

Уровень высшего образования: магистратура

Направление подготовки: 09.04.03 Прикладная информатика

Направленность (профиль) подготовки: Сквозные цифровые технологии в управлении недвижимостью

Квалификация (степень) выпускника: магистр

Форма обучения: очная

Год набора (приема на обучение): 2026

Срок получения образования: 2 года

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

Москва, 2026

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

Рабочая программа дисциплины (модуля) составлена в соответствии с требованиями ФГОС ВО по направлению подготовки 09.04.03 Прикладная информатика, утвержденного приказом Минобрнауки от 19.09.2017 № 916

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	05.11.2025, № 12-25

Содержание

1. Цель и задачи освоения дисциплины (модуля)

2. Планируемые результаты обучения по дисциплине (модулю), соотнесенные с планируемыми результатами освоения образовательной программы

3. Место дисциплины в структуре ОП

4. Объем дисциплины и виды учебной работы

5. Содержание дисциплины
   5.1. Разделы, темы дисциплины и виды занятий
   5.2. Содержание разделов, тем дисциплины и формы текущего контроля

6. Оценочные материалы текущего контроля

7. Оценочные материалы промежуточной аттестации

8. Порядок проведения промежуточной аттестации

9. Материально-техническое и учебно-методическое обеспечение дисциплины
   9.1. Перечень основной и дополнительной учебной литературы
   9.2. Профессиональные базы данных и ресурсы «Интернет», к которым обеспечивается доступ обучающихся
   9.3. Программное обеспечение и информационно-справочные системы, используемые при осуществлении образовательного процесса по дисциплине
   9.4. Оборудование и технические средства обучения

10. Методические указания по освоению дисциплины (модуля)

1. Цель и задачи освоения дисциплины (модуля)

Цель освоения дисциплины - формирование у обучающихся способности решать задачи, связанные с разработкой алгоритмических и программных решений с использованием современных компилируемых и интерпретируемых языков, а также созданием программного обеспечения, взаимодействующего с вычислительными процессами и устройствами на низком уровне.

Задачи изучения дисциплины:

- изучить классификации языков программирования по поколениям, парадигмам;
- освоить принципы построения программ с использованием различных концептуальных подходов;
- изучить принципы работы знаковых систем, основы функционирования вычислительных процессов;
- освоить основной функционал стандартных библиотек, предоставляющих алгоритмы и средства сетевого взаимодействия, API к БД, математических вычислений.

2. Планируемые результаты обучения по дисциплине (модулю), соотнесенные с планируемыми результатами освоения образовательной программы

Компетенции, индикаторы и результаты обучения

ПК-2 Способен управлять рисками разработки программного обеспечения

ПК-2.1 Идентифицирует и оценивает риски разработки программного обеспечения

Знать:

ПК-2.1/Зн1 Классификацию и источники рисков разработки программного обеспечения

ПК-2.1/Зн2 Методы идентификации и оценки вероятности и последствий рисков

Уметь:

ПК-2.1/Ум1 Выявлять риски на этапах разработки, внедрения и сопровождения программного обеспечения

ПК-2.1/Ум2 Формировать и приоритизировать реестр рисков программного проекта

Владеть:

ПК-2.1/Нв1 Навыками идентификации и документирования рисков разработки программного обеспечения

ПК-2.1/Нв2 Навыками качественной и количественной оценки рисков проекта

ПК-2.2 Планирует меры реагирования и осуществляет мониторинг рисков разработки программного обеспечения

Знать:

ПК-2.2/Зн1 Стратегии реагирования на угрозы и возможности программного проекта

ПК-2.2/Зн2 Методы мониторинга рисков и оценки эффективности мер реагирования

ПК-2.2/Зн3 Правила ведения и актуализации проектной документации по рискам

Уметь:

ПК-2.2/Ум1 Разрабатывать и обосновывать план реагирования на риски разработки программного обеспечения

ПК-2.2/Ум2 Контролировать состояние рисков и корректировать меры реагирования

Владеть:

ПК-2.2/Нв1 Навыками планирования и реализации мер реагирования на риски
ПК-2.2/Нв2 Навыками мониторинга рисков и актуализации реестра рисков

3. Место дисциплины в структуре ОП

Дисциплина (модуль) Б1.О.04 «Разработка программного обеспечения и скриптовые языки» относится к обязательной части образовательной программы и изучается в семестре(ах): 2.

В процессе изучения дисциплины студент готовится к решению типов задач профессиональной деятельности, предусмотренных ФГОС ВО и образовательной программой.

4. Объем дисциплины (модуля) и виды учебной работы

Период обучения	Общая трудоемкость (часы)	Общая трудоемкость (ЗЕТ)	Лабораторные занятия (часы)	Лекционные занятия (часы)	Самостоятельная работа (часы)	Промежуточная аттестация (часы)
2 семестр	144	4	16	16	85	Экзамен (27)
Всего	144	4	16	16	85	27

5. Содержание дисциплины (модуля)

5.1. Разделы, темы дисциплины и виды занятий

(часы промежуточной аттестации не указываются)

Наименование раздела, темы	Всего	Лабораторные занятия	Лекционные занятия	Самостоятельная работа	Планируемые результаты обучения, соотнесенные с результатами освоения программы
Раздел 1. Теоретические основы разработки языков и компиляции	**62**	**4**	**16**	**42**	ПК-2.1 ПК-2.2
Тема 1.1. Общее введение в теорию компиляции.	3		1	2	
Тема 1.2. Языки программирования, парадигмы и классификация.	6		2	4	
Тема 1.3. Архитектура языков программирования.	5		1	4	
Тема 1.4. Обзор современных языков и их ключевые особенности.	12		4	8	

Тема 1.5. Введение в лексический и синтаксический анализ как фазы компилятора.	6		2	4	
Тема 1.6. Семантический анализ и промежуточные языки представлений.	7	1	2	4	
Тема 1.7. Оптимизация.	12	2	2	8	
Тема 1.8. Низкоуровневая оптимизация и кодогенерация.	11	1	2	8	
Раздел 2. Изучение современных скриптовых и компилируемых языков программирования	**55**	**12**		**43**	ПК-2.1 ПК-2.2
Тема 2.1. Скриптовый язык Python 3.	11	3		8	
Тема 2.2. Скриптовый язык Javascript.	14	3		11	
Тема 2.3. Компилируемый язык с сборщиком мусора Go (golang).	15	3		12	
Тема 2.4. Компилируемый язык Rust.	15	3		12	
Итого	**117**	**16**	**16**	**85**	

5.2. Содержание разделов, тем дисциплин и формы текущего контроля

Раздел 1. Теоретические основы разработки языков и компиляции

Тема 1.1. Общее введение в теорию компиляции.

Классификация подходов при создании трансляторов. Компиляторы. Интерпретаторы. Виртуальные машины. Jit и Aot-компиляторы. Ассемблер и объектный код.

Текущий контроль

Вид (форма) контроля
Тестирование

Тема 1.2. Языки программирования, парадигмы и классификация.

Языки и грамматики. Грамматики Хомского. Поколения языков программирования. Парадигмы языков программирования. Языки с множественными парадигмами. Обзор парадигм и ЯП к ним относящиеся.

Текущий контроль

Вид (форма) контроля
Тестирование

Тема 1.3. Архитектура языков программирования.

Стандартизация языков. Интернационализация. Типы данных и виды типизации. Управление последовательностью действий. Управление памятью. Синтаксический сахар/соль.

Текущий контроль

Вид (форма) контроля

Тестирование

Тема 1.4. Обзор современных языков и их ключевые особенности.

Рейтинги и классификаторы языков программирования. Компилируемые, интерпретируемые языки и Виртуальные машины. Обзор языков C++, Rust, Scala, Go.

Текущий контроль

Вид (форма) контроля
Тестирование

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

Классическая схема компилятора. Схема работы лексического анализатора. Токены, шаблоны, лексемы. Таблицы представлений и идентификаторов. Классификация подходов синтаксического анализа. Классы КСграмматик. Нисходящие и восходящие методы анализа. LL(k) -грамматика. Конфликты.

Текущий контроль

Вид (форма) контроля
Тестирование

Тема 1.6. Семантический анализ и промежуточные языки представлений.

Атрибутное дерево, триады и тетраиды. Инфиксная и постфиксная запись. Хранение и анализ пользовательских типов. Использование байткода как промежуточного представления. Архитектура виртуальных машин .Net, JVM, LLVM.

Текущий контроль

Вид (форма) контроля
Тестирование

Тема 1.7. Оптимизация.

Классификация подходов оптимизации и стадии оптимизации. Граф потока управления. Граф Def-Use данных. Высокоуровневая оптимизация, преобразование циклов и выражений. Оконная, локальная оптимизация, глобальная оптимизация.

Текущий контроль

Вид (форма) контроля
Тестирование

Тема 1.8. Низкоуровневая оптимизация и кодогенерация.

Низкоуровневая оптимизация и распределение регистров. Функции кодогенерации и подзадачи генерации кода. Регистровая и стековая машины. Стоимость инструкций. Системы восходящего переписывания деревьев.

Текущий контроль

Вид (форма) контроля
Тестирование

Раздел 2. Изучение современных скриптовых и компилируемых языков программирования

Тема 2.1. Скриптовый язык Python 3.

Обзор базовых типов Python 3. Списки, Comprehensions, операции map и filter. Словари и Dict Comprehensions. Функции и Lambda-функции. Функции с переменным числом параметров. Ошибки и Exceptions. Форматирование строк. Объекты и классы. Методы, поля и property.

Текущий контроль

Вид (форма) контроля
Тестирование
Практическая работа

Тема 2.2. Скриптовый язык Javascript.

Обзор интерпретаторов для браузеров и серверов. Базовые типы, строки, числа, NaN, undefined. Типы-объекты: Функции, Ошибки, Даты. Конструкторы и прототипы. Ajax запросы, XMLHttpRequest и fetch.

Текущий контроль

Вид (форма) контроля
Тестирование
Практическая работа

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

Типы данных. Функции. Конструкции потока управления. Структуры и типы. Методы и интерфейсы. Примитивы многозадачности: light-weight processes (goroutines), channels, и select statement.

Текущий контроль

Вид (форма) контроля
Тестирование
Практическая работа

Тема 2.4. Компилируемый язык Rust.

Объявление переменных, изменяемые и неизменяемые переменные и типы данных, жизненный цикл данных. Базовые типы и вывод типов. Конструкции потока управления. Владение переменными. Классы, примеси и методы. Многозадачность без состояния гонки.

Текущий контроль

Вид (форма) контроля
Тестирование
Практическая работа

6. 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.

7. Оценочные материалы текущего контроля

Раздел 1. Теоретические основы разработки языков и компиляции
Контролируемые ИДК: ПК-2.1 ПК-2.2

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-2.1 ПК-2.2

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.

8. Оценочные материалы промежуточной аттестации

2 семестр, Экзамен
Контролируемые ИДК: ПК-2.1 ПК-2.2
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.

9. Материально-техническое и учебно-методическое обеспечение дисциплины

9.1. Перечень основной и дополнительной учебной литературы

Основная литература

1. Карякин М. И. Технологии программирования и компьютерный практикум на языке Python: учебное пособие / М. И. Карякин, К. А. Ватульян, Р. М. Мнухин. - Технологии программирования и компьютерный практикум на языке Python - Ростов-на-Дону, Таганрог: Издательство Южного федерального университета, 2022. - 241 с. - 978-5-9275-4108-9. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/125718.html - Режим доступа: по подписке

2. Саблина В. А. Основы программирования на JavaScript: учебное пособие / В. А. Саблина, Е. А. Трушина. - Рязань: РГРТУ, 2022. - 96 с. - Текст: электронный // Издательство Лань: [сайт]. - URL: https://e.lanbook.com/book/380477 - Режим доступа: по подписке

3. Малявко А. А. Формальные языки и компиляторы: учебник / А. А. Малявко. - Формальные языки и компиляторы - Новосибирск: Новосибирский государственный технический университет, 2014. - 431 с. - 978-5-7782-2318-9. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/47725.html - Режим доступа: по подписке

Дополнительная литература

1. Жилко Е. П. Информатика и программирование. Часть 1: учебное пособие / Е. П. Жилко, Л. Н. Титова, Э. И. Дяминова. - Информатика и программирование. Часть 1 - Москва: Ай Пи Ар Медиа, 2024. - 195 с. - 978-5-4497-3463-1 (ч. 1), 978-5-4497-3462-4. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/142557.html - Режим доступа: по подписке

2. Рощин П. Г. Командная разработка программного обеспечения с помощью системы контроля версий GIT: конспект лекций: учебное пособие / П. Г. Рощин. - Командная разработка программного обеспечения с помощью системы контроля версий GIT: конспект лекций - Москва: Национальный исследовательский ядерный университет «МИФИ», 2022. - 106 с. - 978-5-7262-2846-4. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/132682.html - Режим доступа: по подписке

3. Гегечкори Е. Т. Программирование на языке Python: учебное пособие / Е. Т. Гегечкори. - Программирование на языке Python - Омск: Омский государственный технический университет, 2023. - 172 с. - 978-5-8149-3617-2. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/140856.html - Режим доступа: по подписке

9.2. Профессиональные базы данных и ресурсы «Интернет», к которым обеспечивается доступ обучающихся

Профессиональные базы данных

Не используются.

Ресурсы «Интернет»

1. http://chamo.lib.tsu.ru/search/query?locale=ru&theme=system - Электронный каталог Научной библиотеки ТГУ

2. http://vital.lib.tsu.ru/vital/access/manager/Index - Электронная библиотека (репозиторий) ТГУ

3. https://znanium.ru/ - Электронно-библиотечная система "Znanium"

4. https://e.lanbook.com/ - Электронная библиотечная система "Лань"

5. https://rusneb.ru/ - Национальная электронная библиотека (НЭБ)
6. https://www.elibrary.ru/ - НЭБ eLIBRARY

9.3. Программное обеспечение и информационно-справочные системы, используемые при осуществлении образовательного процесса по дисциплине

Перечень программного обеспечения
(обновление производится по мере появления новых версий программы)
1. Microsoft Visual Studio;

Перечень информационно-справочных систем
(обновление выполняется еженедельно)
Не используется.

9.4. Оборудование и технические средства обучения

Компьютерный класс
Маркерная доска - 0 шт.
Моноблок - 64 шт.
Проектор - 4 шт.

Учебная аудитория
Лекционный мультимедийный комплекс - 1 шт.
Проектор - 1 шт.

10. Методические указания по освоению дисциплины (модуля)

С целью успешного освоения данного предмета студенту необходимо выполнять домашние работы по указанным преподавателям темам и представленным на занятиях материалам. Для более успешного выполнения работы студенту предлагается использование учебно-методических материалов, соответствующих указанной теме.

Самостоятельная работа студентов описывается и регулируется методическими рекомендациями по дисциплине.

Описание возможностей изучения дисциплины лицами с ОВЗ и инвалидами

При необходимости обучения студентов-инвалидов и лиц с ограниченными возможностями здоровья аудиторные занятия могут быть заменены или дополнены изучением полнотекстовых лекций, презентаций, видео- и аудиоматериалов в электронной информационно-образовательной среде (ЭИОС) университета. Индивидуальные задания подбираются в адаптированных к ограничениям здоровья формах (письменно или устно, в форме презентаций). Выбор методов обучения зависит от их доступности для инвалидов и лиц с ограниченными возможностями здоровья.

Рабочая программа дисциплины для инвалидов и лиц с ограниченными возможностями здоровья разрабатывается индивидуально с учетом особенностей психофизического развития, индивидуальных возможностей и состояния здоровья таких обучающихся.