Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский ядерный университет «МИФИ»

ИНСТИТУТ ФИНАНСОВЫХ ТЕХНОЛОГИЙ И ЭКОНОМИЧЕСКОЙ БЕЗОПАСНОСТИ

КАФЕДРА ФИНАНСОВОГО МОНИТОРИНГА

ОДОБРЕНО УМС ИФТЭБ

Протокол № 545-6

от 28.08.2025 г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Направление подготовки (специальность): [1] 10.05.04 Информационно-аналитические системы безопасности

Семестр	Трудоемкость, кред.	Общий объем курса, час.	Лекции, час.	Практич. занятия, час.	Лаборат. работы, час.	В форме практической подготовки, час.	СРС, час.	КСР, час.	Форма(ы) контроля, экз./зач./КР/КП
2	2-3	72-108	30	0	30		12-48	0	З
Итого	2-3	72-108	30	0	30	15	12-48	0	

АННОТАЦИЯ

Дисциплина направлена на получение целостного представления об основах алгоритмизации и программирования на языке С, приобретение навыков разработки программного кода с использованием современных инструментальных средств.

1. ЦЕЛИ И ЗАДАЧИ ОСВОЕНИЯ УЧЕБНОЙ ДИСЦИПЛИНЫ

Цели курса – освоение принципов создания программ в рамках структурной и процедурной парадигм программирования с использованием языка программирования С,

формирование у студентов совокупности профессиональных компетенций, обеспечивающих профессиональное решение задач с помощью компьютера, построение компьютерных моделей

и изучение основных способов алгоритмизации.

2. МЕСТО УЧЕБНОЙ ДИСЦИПЛИНЫ В СТРУКТУРЕ ООП ВО

Изучение данной дисциплины базируется на использовании знаний, полученных в процессе освоения курса информатики, а также ряда дисциплин естественно-научного цикла (математический анализ, линейная алгебра и геометрия, математическая логика, дискретная математика). Кроме того, желательно знание английского языка для чтения иностранных публикаций по теме дисциплины и обязательно наличие навыков программирования на любом языке.

Знания, умения и навыки, полученные студентами в процессе изучения данного курса, необходимы при освоении таких дисциплин, как:

Базы данных и экспертные системы;

Объектно-ориентированный анализ и программирование;

Технологии и методы программирования;

Прикладная математика (теория принятия решений);

Машинное обучение;

при прохождении учебной практики (ознакомительной), производственной практики (преддипломной) а также для подготовки выпускной квалификационной работы (ВКР).

3. ФОРМИРУЕМЫЕ КОМПЕТЕНЦИИ И ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

Универсальные и(или) общепрофессиональные компетенции:

Код и наименование компетенции	**Код и наименование индикатора достижения компетенции**
ОПК-1 [1] – Способен оценивать роль информации, информационных технологий и информационной безопасности в современном обществе, их значение для обеспечения	З-ОПК-1 [1] – знать роль информации, информационных технологий и информационной безопасности в современном обществе, их значение для обеспечения объективных потребностей личности, общества и государства У-ОПК-1 [1] – уметь определять роль информации,

объективных потребностей личности, общества и государства	информационных технологий и информационной безопасности в современном обществе, их значение для обеспечения объективных потребностей личности, общества и государства В-ОПК-1 [1] – владеть основными методами оценки информации, информационных технологий и информационной безопасности в современном обществе, их значение для обеспечения объективных потребностей личности, общества и государства
ОПК-7 [1] – Способен создавать программы на языках высокого уровня, применять методы и инструментальные средства программирования для решения профессиональных задач, осуществлять обоснованный выбор инструментария программирования	З-ОПК-7 [1] – знать языки программирования высокого и низкого уровня, инструментальные средства программирования для решения профессиональных задач У-ОПК-7 [1] – уметь создавать программы на языках высокого и низкого уровня, применять методы и инструментальные средства программирования для решения профессиональных задач, осуществлять обоснованный выбор инструментария программирования и способов организации программ В-ОПК-7 [1] – владеть методами и инструментальными средствами программирования для решения профессиональных задач

4. ВОСПИТАТЕЛЬНЫЙ ПОТЕНЦИАЛ ДИСЦИПЛИНЫ

Направления/цели воспитания	**Задачи воспитания (код)**
Профессиональное воспитание	Создание условий, обеспечивающих, формирование культуры информационной безопасности (В23)
Профессиональное воспитание	Создание условий, обеспечивающих, формирование ориентации на неукоснительное соблюдение нравственных и правовых норм в профессиональной деятельности (В45)

5. СТРУКТУРА И СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

Разделы учебной дисциплины, их объем, сроки изучения и формы контроля:

№ п.п	**Наименование раздела учебной дисциплины**	**Недели**	**Лекции/ Практ. (семинары)/ Лабораторные работы, час.**	**Обязат. текущий контроль (форма*, неделя)**	**Максимальный балл за раздел****	**Аттестация раздела (форма*, неделя)**	**Индикаторы освоения компетенции**
	2 Семестр						

1	Введение в язык программирования С	1-8	16/0/16	Отч-8 (25)	25	КИ-8	З-ОПК-1, У-ОПК-1, В-ОПК-1, З-ОПК-7, У-ОПК-7, В-ОПК-7
2	Программы с множественными единицами трансляции в языке С	9-15	14/0/14	к.р-14 (25)	25	КИ-15	З-ОПК-1, У-ОПК-1, В-ОПК-1, З-ОПК-7, У-ОПК-7, В-ОПК-7
	Итого за 2 Семестр		30/0/30		50		
	Контрольные мероприятия за 2 Семестр				50	З	З-ОПК-1, У-ОПК-1, В-ОПК-1, З-ОПК-7, У-ОПК-7, В-ОПК-7

* – сокращенное наименование формы контроля

** – сумма максимальных баллов должна быть равна 100 за семестр, включая зачет и (или) экзамен

Сокращение наименований форм текущего контроля и аттестации разделов:

Обозначение	**Полное наименование**
Отч	Отчет
к.р	Контрольная работа
КИ	Контроль по итогам
З	Зачет

КАЛЕНДАРНЫЙ ПЛАН

Недели	**Темы занятий / Содержание**	**Лек., час.**	**Пр./сем., час.**	**Лаб., час.**
	2 Семестр	30	0	30
1-8	**Введение в язык программирования С**	16	0	16
1 - 2	**Тема 1. История развития языков программирования. Синтаксис языка программирования С.** Символы и лексемы языка. Операции в языке С: классификация, приоритет выполнения операций. Функции форматированного ввода-вывода. Машинный язык. Машинно-ориентированные язык. Единица трансляции. Программы-трансляторы. Основные этапы трансляции программы.	Всего аудиторных часов 4 Онлайн 0	0	4 0
3 - 4	**Тема 2. Классификация операторов в языке С.** Выражения, операторы и блоки. Условная операция, операторы условия и множественного выбора. Операторы цикла.	Всего аудиторных часов 4 Онлайн 0	0	4 0

	Другие операторы передачи управления.			
5	**Тема 3. Массивы и указатели в языке С.** Объявление и использование массивов в языке С. Тип Указатель. Доступ к элементам массива с помощью указателей. Способы передачи параметров разных типов в функцию.	Всего аудиторных часов		
		2	0	2
		Онлайн		
		0	0	0
6	**Тема 4. Работа с символами и строками в языке С.** Представление символов и строк в языке С. Допустимые операции. Стандартные функции для работы с символами и строками.	Всего аудиторных часов		
		2	0	2
		Онлайн		
		0	0	0
7	**Тема 5. Структуры и объединения в языке С.** Описание пользовательских типов данных в программе. Структуры и объединения – объявление и использование в программах, допустимые операции.	Всего аудиторных часов		
		2	0	2
		Онлайн		
		0	0	0
8	**Тема 6. Файловый ввод-вывод в языке С.** Стандартные потоки. Работа с файлами (тип FILE). Функции языка, используемые для работы с консолью и файлами.	Всего аудиторных часов		
		2	0	2
		Онлайн		
		0	0	0
9-15	**Программы с множественными единицами трансляции в языке С**	14	0	14
9	**Тема 7. Модель памяти.** Динамическая память. Управление памятью.	Всего аудиторных часов		
		2	0	2
		Онлайн		
		0	0	0
10 - 12	**Тема 8. Компоновщик. Отладчик.** Программы из множества единиц трансляции. Компоновка программы. Язык сценариев компоновщика. Объектные файлы. Таблица символов. Отладчик. Работа программы под отладчиком.	Всего аудиторных часов		
		6	0	6
		Онлайн		
		0	0	0
13	**Тема 9. Библиотеки.** Статические и разделяемые библиотеки. Динамическое подключение библиотек.	Всего аудиторных часов		
		2	0	2
		Онлайн		
		0	0	0
14 - 15	**Тема 10. Режимы ввода. Буферизация. Время.** Неканонический режим ввода. Структуры и функции для работы со временем. Управление терминалом.	Всего аудиторных часов		
		4	0	4
		Онлайн		
		0	0	0

Сокращенные наименования онлайн опций:

Обозначение	**Полное наименование**
ЭК	Электронный курс
ПМ	Полнотекстовый материал
ПЛ	Полнотекстовые лекции
ВМ	Видео-материалы
АМ	Аудио-материалы
Прз	Презентации
Т	Тесты
ЭСМ	Электронные справочные материалы
ИС	Интерактивный сайт

ТЕМЫ ЛАБОРАТОРНЫХ РАБОТ

Недели	Темы занятий / Содержание
	2 Семестр
1 - 2	**Лабораторная работа №1** Управляющие структуры и функции языка Си.
3 - 4	**Лабораторная работа №2** Функции языка Си.
5 - 6	**Лабораторная работа №3** Указатели и ссылки. Передача параметров в функцию по указателю. Сортировка одномерного массива.
7 - 8	**Лабораторная работа №4** Обработка символов и строк в языке Си. Работа со структурами и файлами в языке Си.
9 - 12	**Лабораторная работа №5** Динамические структуры данных, связные списки.
13	**Лабораторная работа №6** Разделяемая библиотека функций управления связными списками.
14 - 15	**Лабораторная работа №7** Интерактивная работа со связным списком.

6. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ

В процессе преподавания курса используются следующие формы:

- лекции;

- лабораторные работы, в рамках которых выполняются практические задания, обсуждаются вопросы лекций;

- экспресс-диагностика и контрольная работа по отдельным темам дисциплины;

- самостоятельная работа студентов, включающая изучение теоретического материала, подготовку к защите лабораторных работ, к текущему контролю знаний и к промежуточной аттестации;

- консультирование студентов по сложным вопросам учебного материала и лабораторным работам.

7. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

Фонд оценочных средств по дисциплине обеспечивает проверку освоения планируемых результатов обучения (компетенций и их индикаторов) посредством мероприятий текущего, рубежного и промежуточного контроля по дисциплине.

Связь между формируемыми компетенциями и формами контроля их освоения представлена в следующей таблице:

Компетенция	Индикаторы освоения	Аттестационное мероприятие (КП 1)
ОПК-1	З-ОПК-1	З, КИ-8, КИ-15, Отч-8, к.р-14
	У-ОПК-1	З, КИ-8, КИ-15, Отч-8, к.р-14
	В-ОПК-1	З, КИ-8, КИ-15, Отч-8, к.р-14
ОПК-7	З-ОПК-7	З, КИ-8, КИ-15, Отч-8, к.р-14
	У-ОПК-7	З, КИ-8, КИ-15, Отч-8, к.р-14

	В-ОПК-7	З, КИ-8, КИ-15, Отч-8, к.р-14

Шкалы оценки образовательных достижений

Шкала каждого контрольного мероприятия лежит в пределах от 0 до установленного максимального балла включительно. Итоговая аттестация по дисциплине оценивается по 100-балльной шкале и представляет собой сумму баллов, заработанных студентом при выполнении заданий в рамках текущего и промежуточного контроля.

Итоговая оценка выставляется в соответствии со следующей шкалой:

Сумма баллов	Оценка по 4-х балльной шкале	Отметка о зачете	Оценка ECTS
90-100	5 – *«отлично»*	*«зачтено»*	A
85-89	4 – «*хорошо*»		B
75-84			C
70-74			D
65-69	3 – «*удовлетворительно*»		
60-64			E
ниже 60	2 – «*неудовлетворительно*»	*«не зачтено»*	F

Оценка «отлично» соответствует глубокому и прочному освоению материала программы обучающимся, который последовательно, четко и логически стройно излагает свои ответы, умеет тесно увязывать теорию с практикой, использует в ответах материалы монографической литературы.

Оценка «хорошо» соответствует твердым знаниям материала обучающимся, который грамотно и, по существу, излагает свои ответы, не допуская существенных неточностей.

Оценка «удовлетворительно» соответствует базовому уровню освоения материала обучающимся, при котором освоен основной материал, но не усвоены его детали, в ответах присутствуют неточности, недостаточно правильные формулировки, нарушения логической последовательности.

Отметка «зачтено» соответствует, как минимум, базовому уровню освоения материала программы, при котором обучающийся владеет необходимыми знаниями, умениями и навыками, умеет применять теоретические положения для решения типовых практических задач.

Оценку «неудовлетворительно» / отметку «не зачтено» получает обучающийся, который не знает значительной части материала программы, допускает в ответах существенные ошибки, не выполнил все обязательные задания, предусмотренные программой. Как правило, такие обучающиеся не могут продолжить обучение без дополнительных занятий.

8. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

ОСНОВНАЯ ЛИТЕРАТУРА:

1. ЭИ Т 76 Алгоритмизация и программирование : учебник для вузов, Павловская Т. А., Трофимов В. В., Москва: Юрайт, 2023

2. ЭИ С 60 Практическое введение в язык программирования Си : учебное пособие, Попов И. В., Солдатенко И. С., Санкт-Петербург: Лань, 2022

3. ЭИ Р 27 Программирование на языке Си : учебное пособие для вузов, Рацеев С. М., Санкт-Петербург: Лань, 2023

4. 004 К36 Язык программирования С : , Ритчи Д., Керниган Б., Москва [и др.]: Вильямс, 2015

5. 004 П70 Язык программирования С++. Лекции и упражнения : , Прата С., Москва [и др.]: Вильямс, 2017

ДОПОЛНИТЕЛЬНАЯ ЛИТЕРАТУРА:

1. ЭИ П 12 С/С++. Программирование на языке высокого уровня: Учебник для вузов : , Павловская Т. А., Санкт-Петербург: Питер, 2013

2. ЭИ П 12 С/С++. Процедурное и объектно-ориентированное программирование. Учебник для вузов. Стандарт 3-го поколения : , Павловская Т.А., Санкт-Петербург: Питер, 2015

3. 004 Д30 Основы алгоритмизации и программирования. Язык Си : учеб. пособие, Демидович Е.М., Санкт-Петербург: БХВ-Петербург, 2006

4. ЭИ О-66 Теория и практика языков программирования. Учебник для вузов. Стандарт 3-го поколения : , Орлов С., Санкт-Петербург: Питер, 2014

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ:

1. ОС GNU/Linux (К-822)

LMS И ИНТЕРНЕТ-РЕСУРСЫ:

1. Научная библиотека МИФИ (library.mephi.ru)

2. Ресурсы по методологии и программным продуктам ARIS (http://www.ariscommunity.com/aris-express/tutorials -)

3. Общероссийский математический портал Math-Net.Ru (http://www.mathnet.ru)

4. Образовательный портал GeekBrains (https://gb.ru)

5. Основы программирования на языках Си и С++ (http://cppstudio.com/)

6. Обучающие статьи о Computer Science и использование классических алгоритмов и структур данных в реше (https://tproger.ru/tag/algorithms/)

https://online.mephi.ru/

http://library.mephi.ru/

9. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

Специальное материально-техническое обеспечение не требуется

10. УЧЕБНО-МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ДЛЯ СТУДЕНТОВ

Студентам важно с самого начала выработать привычку к осознанному программированию: не просто реализовывать требуемую функциональность, а постоянно контролировать корректность каждого действия — от обработки результатов операций до управления ресурсами. Такой подход помогает замечать потенциальные проблемы еще на ранних этапах разработки.

Полезно регулярно проверять поведение кода в нетривиальных ситуациях: на граничных и экстремальных значениях, при отсутствии данных или при сбоях во внешних операциях. Простая проверка крайних случаев нередко позволяет выявить скрытые уязвимости и сделать программу надежнее.

Для лучшего понимания того, как код превращается в исполняемую программу, стоит изучать промежуточные этапы обработки — это дает представление о внутренней механике системы и упрощает поиск ошибок. При работе со структурами данных и способами хранения информации важно уделять особое внимание деталям: учитывать особенности представления и ограничения используемых механизмов, заранее продумывать возможные риски.

Для освоения этапов трансляции полезно вручную прогонять код через отдельные стадии: смотреть вывод препроцессора, изучать ассемблерный листинг и объектные файлы. Это формирует понимание того, как исходный текст превращается в исполняемую программу, и сильно упрощает дальнейшую отладку.

При разработке сложных программ полезно придерживаться четкой структуры: разделять функционал на логически независимые части, ясно оформлять интерфейсы и минимизировать элементы с широкой областью видимости. Такой подход облегчает поддержку кода и его дальнейшее развитие.

При сборке многомодульных программ важно с самого начала придерживаться четкой структуры проекта: выносить объявления в заголовочные файлы, следить за областью видимости символов и избегать дублирования. Практика отдельной компиляции каждого модуля в объектный файл и последующей компоновки помогает лучше понять роль линковщика и таблицы символов.

При создании и подключении библиотек стоит не просто следовать готовым инструкциям, а разбираться в механике: понимать, чем отличается статическая линковка от динамической, как система ищет разделяемые библиотеки и как работает динамическая загрузка. Это знание пригодится при переносе программ между системами и при диагностике ошибок запуска.

Освоение инструментов отладки и анализа стоит вести параллельно с написанием кода: регулярное применение этих средств помогает превратить их в привычный рабочий инструмент, а не прибегать к ним только в критических ситуациях. Это существенно ускоряет выявление и исправление ошибок.

Работа с внешними компонентами и механизмами взаимодействия требует не формального следования инструкциям, а понимания лежащих в их основе принципов.

Осознанное отношение к подключению библиотек, управлению ресурсами и взаимодействию с окружением делает решения более устойчивыми и переносимыми.

В проектной деятельности стоит заранее продумывать общую архитектуру и способы проверки корректности работы: наличие простых сценариев проверки и понятных пояснений к ключевым решениям заметно повышает качество результата.

Оценка работы ориентирована не только на то, что программа выполняет требуемые действия, но и на то, насколько предсказуемо и надежно она ведет себя в разных условиях. Поэтому полезно регулярно возвращаться к уже написанному коду, анализировать его с позиции возможных уязвимостей и постепенно улучшать — это формирует профессиональный взгляд на разработку.

11. УЧЕБНО-МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ДЛЯ ПРЕПОДАВАТЕЛЕЙ

Преподавателям курса стоит выстраивать обучение так, чтобы у студентов формировалось не формальное знание синтаксиса, а инженерное понимание механизмов работы языка. Эффективнее всего подавать материал через связку концепции с практическими примерами и разбором типичных ошибок: такой подход помогает учащимся видеть не только как писать код, но и почему отдельные конструкции ведут себя определенным образом и какие риски за ними скрываются.

Важную роль играет демонстрация внутренней механики преобразований — от исходного текста до исполняемой программы: это делает процессы более предсказуемыми для обучающихся. Визуализация структур и схем размещения данных, а также разбор поведения кода в инструментах отладки позволяют упростить восприятие абстрактных понятий и быстрее выработать устойчивые навыки поиска проблем.

При организации лабораторных работ стоит ориентироваться на постепенное наращивание сложности и последовательное освоение ключевых рисков, характерных для языка С. Задания следует формулировать так, чтобы обязательной частью их выполнения была проверка поведения программы в нетривиальных и потенциально опасных ситуациях — например, при некорректном вводе, отсутствии ресурсов или выходе за допустимые границы структур данных. Это приучает студентов к ответственному подходу и закладывает основы надежной разработки.

Полезно включать в лабораторные задания элементы проектирования: пусть студенты самостоятельно продумывают интерфейсы функций, определяют допустимые диапазоны входных данных и описывают способы обработки ошибочных ситуаций. Такой подход развивает не только навыки кодирования, но и умение заранее учитывать возможные проблемы и прорабатывать способы их предотвращения.

Особое внимание в лабораторных работах стоит уделять инструментам анализа и отладки: студенты должны регулярно использовать отладчики, средства проверки памяти и утилиты для изучения промежуточных результатов трансляции. Эффективной практикой является требование сопровождать решение задачи кратким описанием того, как именно проверялась корректность работы — например, какие тесты запускались, какие инструменты применялись и какие проблемы удалось выявить и устранить.

Целесообразно выстраивать оценку лабораторных работ не только по критерию «код работает», но и по тому, насколько осознанно и устойчиво реализовано решение: учитывается прозрачность управления ресурсами, предсказуемость поведения при сбоях, четкость архитектуры и корректность сборки. Для этого в задания полезно включать требования к

структуре проекта, оформлению интерфейсов, ведению версий и подготовке сценариев тестирования.

Хорошим приемом считается включение в лабораторные работы фрагментов заведомо ошибочного или уязвимого кода, который студенты должны проанализировать, объяснить проблему и предложить способы ее устранения. Такие задания развивают критическое мышление и помогают лучше усвоить типичные ошибки, характерные для низкоуровневого программирования.

Оценка результатов должна отражать не только факт работоспособности программы, но и степень осознанности принятых решений. Поэтому в контрольные мероприятия уместно включать задания на анализ чужого кода, объяснение логики работы и выявление потенциальных уязвимостей — это лучше проверяет глубину понимания, чем простое написание решений.

Ценной оказывается и регулярная обратная связь, ориентированная на развитие: важно не просто указывать на ошибки, а показывать, где скрыты риски и как сделать решение устойчивее, подкрепляя замечания конкретными примерами.

Методическая поддержка со стороны преподавателя помогает студентам сосредоточиться на сути, а не на организационных деталях: наличие готовых шаблонов структуры проекта, единых требований к стилю кода и понятных чек-листов заметно экономит время и делает процесс обучения более прозрачным. При этом сохранение единой терминологии и последовательности подачи материала на всех видах занятий снижает путаницу и помогает учащимся увереннее ориентироваться в разнородных источниках и инструментах.

Автор(ы):

Курасов Юрий Викторович