ЦЕНТРАЛЬНЫЙ УНИВЕРСИТЕТ

УТВЕРЖДЕНА

Решением Ученого совета
АНО ВО «Центральный университет»
«25» мая 2026 г.
Протокол №2

Рабочая программа дисциплины (модуля)
«Архитектура компьютера и операционные системы. Часть 1»

Направление подготовки: *02.03.01 Математика и компьютерные науки*

Направленность (профиль) подготовки: *Математика и компьютерные науки*

Квалификация (степень) выпускника: *бакалавр*

Форма обучения: *очная*

Срок освоения программы: *4 года*

Год набора: *2026*

ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ

Сертификат: 0215B0C000D1B3FEB44A37EE631E0C27D4
Владелец: Ивашкевич Евгений Васильевич
Действителен: с 13.01.2026 до 13.04.2027

Электронный документ

Москва
2026

Содержание

1. **Краткая характеристика дисциплины (модуля)** 3
2. **Перечень планируемых результатов обучения** 5
3. **Тематический план** 7
4. **Содержание дисциплины (модуля)** 8
5. **Учебно-методическое обеспечение** 9
6. **Материально-техническое обеспечение** 9
7. **Методические и оценочные материалы** 11

1. Краткая характеристика дисциплины (модуля)

Рабочая программа дисциплины (модуля) «Архитектура компьютера и операционные системы. Часть 1» составлена в соответствии с федеральным государственным образовательным стандартом высшего образования – бакалавриат по специальности 02.03.01 «Математика и компьютерные науки», профиль «Математика и компьютерные науки», утвержденный приказом Министерства науки и высшего образования Российской Федерации № 807 от 23.08.2017 года.

Изучение дисциплины (модуля) «Архитектура компьютера и операционные системы. Часть 1» позволяет студентам позволяет понять, как функционируют компьютерные системы, включая взаимодействие аппаратного и программного обеспечения, что критично для оптимизации производительности и разработки эффективных приложений. Это знание поможет в профессиональной деятельности эффективно управлять ресурсами, обеспечивая стабильность и безопасность вычислительных систем.

Место дисциплины (модуля) в структуре образовательной программы

Настоящая дисциплина (модуль) включена в учебный план по программе подготовки бакалавриата по направлению 02.03.01 «Математика и компьютерные науки», профиль «Математика и компьютерные науки» и входит в обязательную часть Блока 1, как дисциплина по выбору.

Дисциплина (модуль) изучается на 2 курсе в 3 семестре. Доступна к изучению после успешного освоения одной из дисциплин (модулей): «Основы промышленной разработки», «Язык программирования C++ 1» или «Язык программирования Go 1».

Цель изучения дисциплины (модуля): заключается в формировании у студентов фундаментальных знаний об устройстве работы компьютера и операционных систем, которые необходимы для эффективной разработки программного обеспечения, а также для понимания более сложных аспектов компьютерных наук, таких как распределенные системы, виртуализация и кибербезопасность.

Задачи изучения дисциплины (модуля):

— изучить основные компоненты архитектуры компьютера, включая процессоры, память и устройства ввода-вывода, их взаимодействие и принципы работы;

— освоить фундаментальные концепции операционных систем, такие как управление процессами, памятью и файловыми системами;

— развить навыки анализа и оптимизации производительности вычислительных систем, включая программирование на уровне ассемблера и работу с низкоуровневыми интерфейсами ОС.

В результате освоения дисциплины (модуля) обучающийся должен:

знать:

— синтаксис и семантику языка Си в части управления памятью;

— этапы компиляции программы и роль каждого артефакта;

— принципы организации статических и динамических библиотек, заголовочных файлов и системы пакетов в Linux;

— базовые логические блоки процессора и побитовые операции, способы представления целых и вещественных чисел, текстовых и бинарных данных, кодировок;

— концепцию архитектур набора команд (ISA), отличия архитектур (RISC, CISC); основы кросс-компиляции;

— структуру и семантику инструкций ассемблера: работу с памятью, условные переходы, циклы, стек вызовов и соглашение о вызовах;

— оптимизации процессора и векторных инструкций (SIMD);

— механизм системных вызовов как интерфейс между прикладными программами и ядром ОС;

— принципы работы с динамическими библиотеками: загрузка, символы, связывание на этапе выполнения.

уметь:

— писать программы на языке Си с ручным управлением памятью: выделять и освобождать память, избегать утечек и неопределённого поведения;

— читать и интерпретировать вывод компилятора (предупреждения, ошибки);

— применять побитовые операции для кодирования, маскирования и упаковки данных в память;

— читать и писать простые фрагменты на ассемблере: реализовывать условия, циклы, вызовы функций с соблюдением соглашения о вызовах;

— производить системные вызовы напрямую;

— подключать и использовать динамические библиотеки;

— объяснять, почему вещественная арифметика неточна, и применять правила работы с IEEE 754 при отладке численных ошибок.

владеть:

— навыками разработки программ через системные вызовы;

— практикой отладки низкоуровневого кода;

— навыками разработки программ на языке Си;

— навыками дизассеблирования программ и анализа их поведения;

— практикой безопасной работы с памятью при разработке низкоуровневых программ.

2. Перечень планируемых результатов обучения

Компетенции, формируемые в результате освоения дисциплины (модуля) при проведении учебных занятий в форме контактной работы обучающихся с педагогическими работниками Университета и в форме самостоятельной работы обучающихся:

Компетенция	Содержание компетенции	Индикатор компетенции	Перечень планируемых результатов обучения по дисциплине (модулю)
УК-1.	Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	УК-1.1.	Знает методы поиска и анализа информации в области разработки, основные принципы критической оценки источников информации и их релевантности.
		УК-1.2.	Умеет критически оценивать источники информации и синтезировать данные из различных источников для решения задач, применять системный подход к анализу и решению комплексных проблем.
		УК-1.3.	Имеет практический опыт работы с современными инструментами и технологиями для обработки информации, формулировании и структурировании задач на основе полученной информации.
УК-2.	Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1.	Знает действующие правовые нормы, регулирующие деятельность в области решения задач, основные методы и подходы к определению круга задач.
		УК-2.2.	Умеет определять круг задач в рамках поставленной цели, выбирать оптимальные способы решения задач, учитывая имеющиеся ресурсы и ограничения.
		УК-2.3.	Имеет практический опыт применения знаний о правовых нормах и ресурсах в реальных ситуациях, разработки и реализации решений в соответствии с установленными ограничениями.
ОПК-4.	Способен находить, анализировать, реализовывать программно и использовать на практике математические алгоритмы, в том числе с применением современных вычислительных систем	ОПК-4.1.	Знает базовые основы современного математического аппарата, связанного с проектированием, разработкой, реализацией и оценкой качества программных продуктов и программных комплексов в различных областях человеческой деятельности.
		ОПК-4.2.	Умеет использовать этот математический аппарат в профессиональной деятельности.

		ОПК-4.3.	Имеет практический опыт применения современного математического аппарата, связанного с проектированием, разработкой, реализацией и оценкой качества программных продуктов и программных комплексов в различных областях человеческой деятельности.
ОПК-5.	Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности	ОПК-5.1.	Обладает базовыми знаниями, полученными в области математических и (или) естественных наук, программирования и информационных технологий
		ОПК-5.2.	Умеет пользоваться технологиями прикладного программирования, включая среды высокоуровневого программирования.
		ОПК-5.3.	Имеет практический опыт использования технологий прикладного программирования.
ПК-3	Способен использовать методы математического и алгоритмического моделирования при решении теоретических и прикладных задач	ПК-3.1.	Знает основные методы математического и алгоритмического моделирования, а также их применение для решения теоретических и прикладных задач
		ПК-3.2.	Умеет разрабатывать и применять математические модели и алгоритмы для решения различных задач, анализируя полученные результаты
		ПК-3.3.	Имеет практический опыт использования методов математического и алгоритмического моделирования в реальных проектах или исследованиях

3. Тематический план

№п/п	Наименование раздела дисциплины (модуля)	Трудоемкость, академические часы				ТКУ (текущий контроль успеваемости)
		Очная форма				
		Аудиторная работа		Контроль	Самостоятельная работа	
		Лекции	Семинары			
1	Основы низкоуровненого программирования в Linux	10	16	4	32	Домашние задания, Контрольная работа
2	Машинное представление данных	6	12		20	Домашние задания
3	Архитектуры команд процессоров и ассемблер	10	12	8	32	Домашние задания, Контрольная работа
4	Взаимодействие с операционной системой	4	8		12	Домашние задания
	Экзамен			4		
	Итого	***30***	***48***	***16***	***96***	
Объем дисциплины (модуля) (в ак. ч.)		***190***				
Объем дисциплины (модуля) (в зач. ед.)		***5***				

4. Содержание дисциплины (модуля)

№п/п	Наименование раздела дисциплины (модуля)	Содержание дисциплины (модуля) по темам
1	Основы низкоуровненого программирования в Linux	Введенние в язык Си и инструменты разработки Процесс и артефакты компиляции. Препроцессор Си Арифметика указателей, строки и преобразования строк Структуры данных и работа с данными на куче Библиотеки, заголовочные файлы и пакеты в Linux
2	Машинное представление данных	Базовые логические блоки процессора и целочисленная арифметика Текстовые и бинарные данные, кодировки и работа с файлами Вещественная арифметика и квантизация
3	Архитектуры команд процессоров и ассемблер	Трансляция программ с высокоуровневых языков на линейный код Архитектуры команд процессора, кросс-компиляция и основы ассемблера Ассемблер: работа с памятью и реализация условий и циклов Ассемблер: стек вызовов и реализация функций Оптимизации процессора и векторные инструкции
4	Взаимодействие с операционной системой	Системные вызовы Динамические библиотеки и их использование

5. Учебно-методическое обеспечение

Университет располагает полным набором лицензионного и свободно распространяемого программного обеспечения, включая продукты отечественного производства.

Каждый студент в течение всего периода обучения получает индивидуальный неограниченный доступ к электронно-библиотечной системе и электронной информационно-образовательной среде университета. Эти системы предоставляют возможность доступа к ресурсам из любой точки, где есть подключение к сети Интернет, как на территории университета, так и за его пределами.

Студентам обеспечен удаленный доступ к современным профессиональным базам данных и информационным справочным системам.

Основная литература:

1. Таненбаум, Э. С. Компьютерные сети / Э. С. Таненбаум, Д. Уэзеролл. - 5-е изд. - Санкт-Петербург : Питер, 2021. - 960 с. - ISBN 978-5-4461-1248-7. - Текст : электронный. - URL: https://znanium.ru/catalog/product/2141427.

2. Кузин, А. В. Программирование на языке Си : учебное пособие / А.В. Кузин, Е.В. Чумакова. — Москва : ФОРУМ : ИНФРА-М, 2024. — 143 с. — (Среднее профессиональное образование). - ISBN 978-5-00091-556-1. - Текст : электронный. - URL: https://znanium.ru/catalog/product/2137197.

3. Рабчевский, А. Н. Компьютерные сети и системы связи. Вводный курс : учебное пособие для вузов / А. Н. Рабчевский. — Москва : Издательство Юрайт, 2025. — 207 с. — (Высшее образование). — ISBN 978-5-534-21489-5. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/572633.

Дополнительная литература:

1. Амини Камран. Экстремальный Си. Параллелизм, ООП и продвинутые возможности. — СПб.: Питер, 2021. — 752 с. — ISBN 978-5-4461-1694-2

2. Керриск Майкл. Linux API. Исчерпывающее руководство. — СПб.: Питер, 2019. — 1248 с. — ISBN 978-5-4461-0985-2

3. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. — СПб.: Питер, 2022. — 1120 с. — ISBN 978-5-4461-1155-8

6. Материально-техническое обеспечение

Университет располагает материально-технической базой, соответствующей действующим противопожарным правилам и нормам и обеспечивающей проведение всех видов дисциплинарной и междисциплинарной подготовки, практической и научно-исследовательской работ обучающихся, предусмотренных учебным планом.

Помещения, которые представляют собой учебные аудитории для проведения занятий лекционного типа, занятий семинарского (практического) типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, а также помещения для самостоятельной работы и помещения для хранения и профилактического обслуживания учебного оборудования. Помещения укомплектованы специализированной мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории.

Изучение дисциплины (модуля) обеспечивается в учебных аудиториях, оснащенных:

- столами и стульями;
- компьютерной техникой;
- механическими калькуляторами;
- специализированным оборудованием, включая демонстрационное оборудование.

Помещения для самостоятельной работы обучающихся, в том числе приспособленные для использования инвалидами и лицами с ограниченными возможностями здоровья, оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду Университета.

Обучающимся предоставляется доступ (в том числе удаленный) к ресурсам информационно-телекоммуникационной сети «Интернет», электронным ресурсам (в том числе электронным библиотечным системам, современным профессиональным базам данных и информационным справочным системам):

№	Наименование портала (издания, курса, документа)	Ссылка
1.	Научная электронная библиотека elibrary.ru библиотека	https://elibrary.ru/defaultx.asp
2.	База данных для IT-специалистов	https://habr.com
3.	База данных ScienceDirect	https://www.sciencedirect.com
4.	Официальный сайт Министерства науки и высшего образования Российской Федерации	https://minobrnauki.gov.ru/
5.	Федеральный портал «Российское образование"	https://www.edu.ru/
6.	Информационная система "Единое окно доступа к образовательным ресурсам"	http://window.edu.ru/
7.	Единая коллекция цифровых образовательных ресурсов	http://school-collection.edu.ru/
8.	Федеральный центр информационно - образовательных ресурсов	http://fcior.edu.ru/

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), в том числе комплект лицензионного программного обеспечения, современные профессиональные базы данных и информационные справочные системы:

Наименование ПО	Производство	Лицензионное / свободно распространяемое
Операционные системы:		
Microsoft Imagine (Windows Client, Server)	зарубежное	лицензионное
Браузеры:		
Яндекс.Браузер	отечественное	свободно распространяемое
Google Chrome	зарубежное	свободно распространяемое
Офисные приложения:		
Microsoft Imagine (Visio, OneNote)	зарубежное	лицензионное
TeXstudio	зарубежное	свободно распространяемое
Adobe Acrobat Reader	зарубежное	свободно распространяемое
Программное обеспечение для планирования и учета времени:		
Toggle app	зарубежное	свободно распространяемое
Системы управления проектами:		
Microsoft Imagine (Project)	зарубежное	лицензионное
Системы управления базами данных:		
Microsoft Imagine (SQL Server)	зарубежное	лицензионное
Системы резервного копирования (backup):		
Acronis Backup Advanced for HyperV	зарубежное	лицензионное
Справочно-правовые системы:		
КонсультантПлюс: справочно-правовая система	отечественное	лицензионное
Средства антивирусной защиты:		

Kaspersky Endpoint Security для бизнеса Стандартный Russian Edition	отечественное	лицензионное
Среды разработки:		
Visual Studio Code	зарубежное	свободно распространяемое
Bash (Unix shell)	зарубежное	свободно распространяемое
Anaconda	зарубежное	свободно распространяемое
Robotic Operating System	зарубежное	свободно распространяемое
CopelliaSim	зарубежное	свободно распространяемое
Google Colaboratory	зарубежное	свободно распространяемое
Пакеты программных средств и библиотек:		
AutoPsy	зарубежное	свободно распространяемое
Interactive Disassembler (IDA)	зарубежное	свободно распространяемое
Системы управления библиографической информацией:		
Zotero	зарубежное	свободно распространяемое
Сервисы и службы:		
Bind	зарубежное	свободно распространяемое
Docker	зарубежное	свободно распространяемое

7. Методические и оценочные материалы

Методические указания для обучающихся по освоению дисциплины (модуля)

В процессе изучения дисциплины (модуля) «Архитектура компьютера и операционные системы. Часть 1» в рамках текущего контроля успеваемости используются такие виды учебной работы, как лекции, семинары, домашние задания и контрольные работы, а также различные виды самостоятельной работы обучающихся по заданию преподавателя, направленные на развитие навыков профессиональной лексики, закрепление практических профессиональных компетенций, поощрение инициатив.

Лекция – систематическое, последовательное, монологическое изложение преподавателем учебного материала, как правило, теоретического характера.

В процессе лекций рекомендуется вести конспект лекций: кратко и схематично фиксировать основные идеи, выводы и обобщения лекции; выделять важные мысли, ключевые слова и термины. Необходимо отметить вопросы или материалы, которые вызывают затруднения, и попытаться найти ответы в рекомендованной литературе. Если разобраться в материале не удается, следует сформулировать вопрос и задать его преподавателю на консультации или во время семинарского (практического) занятия.

Семинар – это форма учебной деятельности, проводимая в учебном заведении под руководством преподавателя, где студенты активно участвуют в обсуждениях, практических заданиях и других формах взаимодействия.

Для успешной подготовки к семинару рекомендуется заранее ознакомиться с темой занятия и основными материалами, чтобы иметь возможность активно участвовать в обсуждении. Также полезно подготовить вопросы и идеи для обсуждения, что поможет глубже понять материал и продемонстрировать заинтересованность.

Аудиторная работа – активная работа студента на семинаре, его ответы на вопросы преподавателя и участие в дискуссии.

Для успешного участия в семинаре студентам рекомендуется заранее ознакомиться с темой обсуждения, прочитать необходимые материалы и подготовить вопросы. Важно активно слушать и вовлекаться в дискуссию, высказывая свои мнения и аргументируя их. При ответах на вопросы преподавателя стоит быть уверенным, четким и логичным, опираясь на изученный материал. Также полезно поддерживать диалог с однокурсниками, чтобы обогатить обсуждение и расширить свои знания.

Домашнее задание – набор задач по темам недели.

При работе над домашними заданиями важно внимательно ознакомиться с требованиями и сроками выполнения. Рекомендуется разбивать задания на этапы, чтобы избежать перегрузки и лучше усвоить материал. Использовать различные источники информации, включая учебники и онлайн-ресурсы, для более глубокого понимания темы.

Контрольная работа – письменная работа с набором задач, которые нужно решить за ограниченное время.

Цель контрольной работы - получить специальные знания по одной или нескольким темам дисциплины (модуля) и продемонстрировать навыки их практического применения.

Самостоятельная работа – работа студентов, направленная на углубленное изучение отдельных тем и вопросов учебной дисциплины (модуля).

В процессе самостоятельной работы студенты взаимодействуют с рекомендованными материалами при минимальном участии преподавателя. Задачи студента включают работу с конспектами лекций (обработка текста), повторное изучение учебных материалов планов и тезисов ответов, изучение дополнительных тем, выполнение учебно-исследовательских заданий и другое.

Система оценивания результатов обучения по дисциплине (модулю)

Критерии получения уровня и оценивания сформированности компетенций по дисциплине (модулю) «Архитектура компьютера и операционные системы. Часть 1»

Оценивание уровня учебных достижений, обучающихся по дисциплине (модулю), осуществляется в виде текущего контроля успеваемости и промежуточной аттестации.

Промежуточная аттестация по дисциплине (модулю) осуществляется в форме ***экзамена***, при этом проводится оценка компетенций, сформированных по дисциплине.

Для оценивания текущего контроля успеваемости и промежуточной аттестации используется десятибалльная шкала оценивания, которая соотносится с традиционной пятибалльной шкалой следующим образом, если иное не указано в силлабусе дисциплины:

Десятибалльная оценка	**Пятибалльная оценка**	**Общая характеристика результата обучения по дисциплине (модулю)**
10	Отлично	Студент полностью владеет знаниями, изложенными в рабочей программе, и глубоко осмысляет дисциплину. Он самостоятельно и логически последовательно отвечает на все вопросы, акцентируя внимание на наиболее важном. Умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя ключевые моменты и устанавливая причинно-следственные связи. Четко формулирует ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные задачи. Студент хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты дисциплины (модуля) с практическими задачами.
9	Отлично	
8	Отлично	
7	Хорошо	Студент обладает знаниями предмета почти в полном объеме рабочей программы и самостоятельно, логически последовательно и всесторонне отвечает на все вопросы,
6	Хорошо	

Десятибалльная оценка	Пятибалльная оценка	Общая характеристика результата обучения по дисциплине (модулю)
		акцентируя внимание на наиболее значимых моментах. Он умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя его ключевые аспекты и устанавливая причинно-следственные связи. Формулирует свои ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные ситуационные задачи. Студент хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты предмета с практическими задачами.
5	Удовлетворительно	Студент обладает базовыми знаниями по дисциплине (модулю), но испытывает трудности при самостоятельных ответах и использует неточные формулировки. В ходе ответов он допускает ошибки, касающиеся сути вопросов. Студент способен решать только самые простые задачи и владеет лишь минимальным набором методов исследования.
4	Удовлетворительно	
3	Не сдан	Студент не овладел обязательным минимумом знаний по предмету и не может ответить на вопросы, даже если преподаватель задает дополнительные наводящие вопросы.
2	Не сдан	
1	Не сдан	

Дисциплина (модуль) «Архитектура компьютера и операционные системы. Часть 1» оценивается следующим образом:

Активность	Вес	Количество	Описание
Домашнее задание	20%	11	Набор задач по темам недели
Контрольная работа	30%	3	Письменная работа с набором задач, которые нужно решить за ограниченное время
Аудиторная работа	10%	1	Активная работа студента на занятии
Экзамен	40%	1	Письменная или устная работа над заданием, направленным на проверку полученных знаний и навыков по курсу

Формула расчёта итоговой оценки по дисциплине (модулю) «Архитектура компьютера и операционные системы. Часть 1»: «0,2 × среднее за домашние задания + 0,3 × среднее за контрольные работы + 0,1 × аудиторная работа + 0,4 × экзамен».

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные домашние задания

Домашнее задание 1.

1. Напишите простую программу на языке Си, которая выводит "Hello, World!" на экран. Объясните, как работает каждая часть программы.
2. Составьте список и кратко опишите основные инструменты разработки для языка Си (например, компиляторы, IDE, отладчики).

3. Реализуйте программу, которая запрашивает у пользователя два числа и выводит их сумму. Используйте комментарии для пояснения кода.

4. Объясните, что такое переменные и типы данных в языке Си. Приведите примеры использования различных типов данных.

5. Напишите программу, которая использует условные операторы для определения, является ли введенное пользователем число четным или нечетным.

Домашнее задание 2.

1. Напишите программу на C, которая выводит побитовое представление целого числа (например, int x = -5) с помощью побитовых операций. Используйте цикл и маску 1 << i, чтобы вывести 32 бита. Объясните, как представляются отрицательные числа (дополнительный код).

2. Напишите функцию hex_to_int(const char *hex), которая преобразует строку в шестнадцатеричном формате (например, "1A") в целое число. Реализуйте обработку символов 'A'–'F' и 'a'–'f' вручную (без strtol). Протестируйте на нескольких значениях.

3. Создайте программу, которая читает текстовый файл с числами (по одному на строку) и записывает их в бинарный файл с помощью fwrite. Затем напишите вторую программу, которая читает этот бинарный файл и выводит значения. Сравните размеры текстового и бинарного файлов.

4. Напишите программу, демонстрирующую неточность вещественной арифметики. Выполните цикл for (float f = 0.0; f != 1.0; f += 0.1) и объясните, почему он может стать бесконечным. Исправьте сравнение с использованием эпсилон.

5. Напишите функцию, которая определяет порядок байт (endianness) на текущей системе. Используйте объединение (union) или указатель на int, чтобы проверить, в каком порядке записываются байты. Выведите результат: "little-endian" или "big-endian".

Домашнее задание 3.

1. Скомпилируйте простую функцию на C (например, int add(int a, int b) { return a + b; }) в ассемблер с помощью gcc -S. Изучите полученный код (AT&T или Intel синтаксис). Объясните, какие инструкции отвечают за вход в функцию, сложение и возврат значения.

2. Напишите на ассемблере (встроенный ассемблер GCC или отдельный .s файл) функцию max(int a, int b), которая возвращает большее из двух чисел, используя условный переход. Свяжите её с C-программой и протестируйте.

3. Напишите программу на C, которая вызывает функцию, а затем изучите стек вызовов с помощью отладчика (gdb). Покажите, где находятся: возвращаемый адрес, аргументы, локальные переменные. Объясните, как работает стек при вложенных вызовах.

4. Реализуйте цикл на ассемблере (например, сумму чисел от 1 до 10) с использованием меток и инструкций cmp, jl, jmp. Вставьте его во встроенный ассемблер (asm) в C-программе и выведите результат.

5. Напишите кросс-компиляторную команду для сборки C-программы под архитектуру ARM (например, с использованием arm-linux-gnueabi-gcc). Объясните, зачем нужна кросс-компиляция и в каких случаях она применяется.

Примерные задания для контрольной работы
Контрольная работа 1.

1. Напишите программу на языке Си, которая выводит информацию о системе (например, имя хоста, версия ОС). Объясните, как вы используете системные вызовы для получения этой информации.

2. Опишите процесс создания динамической библиотеки в Linux. Напишите пример кода, который показывает, как создать и использовать динамическую библиотеку.

3. Создайте программу, которая использует динамическую библиотеку для выполнения математических операций (например, сложение и вычитание). Объясните, как вы подключаете библиотеку и вызываете функции из неё.

4. Исследуйте зависимости между библиотеками. Напишите о том, как можно проверить зависимости между библиотеками с помощью утилит Linux (например, ldd).

5. Создайте Makefile для проекта, который использует динамические библиотеки. Объясните, как он организует сборку проекта и какие команды вы используете.

6. Опишите, как пакеты в Linux управляют установкой и удалением библиотек. Приведите примеры менеджеров пакетов (например, apt, yum) и их команд.

7. Напишите программу, которая демонстрирует использование нескольких динамических библиотек, каждая из которых выполняет разные функции (например, математические и строковые операции). Объясните, как вы организовали код и структуру проекта.

Контрольная работа 2.

1. Объясните, что такое трансляция программы в машинный код. Опишите, как высокоуровневый код на языке Си преобразуется в последовательность инструкций ассемблера и машинный код. Приведите пример простой функции и её представления на ассемблере.

2. Напишите функцию на языке Си, которая складывает два числа, и скомпилируйте её в ассемблер с помощью gcc -S. Изучите полученный код и укажите инструкции, отвечающие за загрузку аргументов, выполнение сложения и возврат результата. Объясните назначение используемых регистров.

3. Исследуйте, как в ассемблере реализуются условия. Напишите фрагмент кода (на встроенном ассемблере GCC или в .s файле), который сравнивает два значения и переходит по метке, если первое меньше второго. Объясните, как работают инструкции cmp, jl, jmp.

4. Создайте цикл на ассемблере, который суммирует числа от 1 до 10. Используйте метки, инструкции cmp и jle. Вставьте код во встроенный ассемблер в С-программе и выведите результат. Объясните, как организован счётчик цикла и условие выхода.

5. Опишите, как устроена работа со стеком вызовов в ассемблере. Напишите простую функцию на C, вызовите её, а затем с помощью gdb или анализа ассемблерного кода покажите, как формируется кадр стека: где хранятся аргументы, возвращаемый адрес и локальные переменные.

6. Напишите на ассемблере фрагмент кода, который использует битовые операции (сдвиги, AND, OR) для реализации умножения на степень двойки. Объясните, как такие операции используются в оптимизации и почему они быстрее арифметических инструкций.

7. Исследуйте, как в ассемблере реализуется вызов функции и возврат из неё. Напишите простую функцию multiply(a, b) на C, посмотрите её ассемблерный код и объясните, как передаются аргументы, как работает call и ret, и как соблюдается соглашение о вызовах (например, cdecl или System V ABI).

Примерные вопросы для подготовки к занятию

1. Как целые числа представляются в памяти компьютера? Объясните разницу между беззнаковыми и знаковыми целыми, а также принцип дополнительного кода.

2. Что такое битовые операции (AND, OR, XOR, NOT, сдвиги) и как они используются в низкоуровневом программировании? Приведите пример использования сдвигов для умножения на степень двойки.

3. В чём разница между текстовыми и бинарными файлами? Как данные записываются и считываются в каждом из форматов в языке Си?

4. Как устроено представление чисел с плавающей запятой по стандарту IEEE 754? Объясните структуру (знак, экспонента, мантисса) и назовите основные источники погрешности при вычислениях.

5. Что такое кодировки символов (ASCII, UTF-8, UTF-16)? Как кодировка влияет на хранение и обработку текстовых данных в программе?

Задания для промежуточной аттестации по дисциплине (модулю)

№ п/п	Задание	Ответ	Компетенция
1.	Укажите этап компиляции, на котором выполняются директивы вида **#include** и **#define**.	препроцессинг / preprocessing	УК-1
2.	Как называется принцип проектирования, при котором поведение системы анализируется как взаимодействие компонентов: процессор, память, компилятор, ОС?	системный подход	УК-1
3.	Укажите, какой артефакт компиляции содержит машинный код, но ещё не может быть выполнен напрямую из-за неразрешённых ссылок.	объектный файл / object file / .o файл	УК-1
4.	Как называется процесс преобразования высокоуровневого кода в последовательность машинных инструкций?	трансляция / компиляция	УК-1
5.	Укажите команду в терминале Linux для компиляции C-файла с помощью GCC без этапа линковки.	gcc -c / gcc -c file.c	УК-2
6.	Какой флаг GCC используется для указания имени выходного файла при компиляции?	-o	УК-2
7.	Укажите, какой ресурс может быть ограничен при выполнении программы, использующей большое количество рекурсивных вызовов.	стек / stack / память стека	УК-2
8.	Какой тип лицензии чаще всего используется для системных библиотек в Linux (например, glibc)?	LGPL / GPL / open source / свободная лицензия	УК-2
9.	Укажите, какое логическое устройство выполняет операцию сложения двух битов с учётом переноса.	сумматор / full adder	ОПК-4
10.	Как называется способ представления отрицательных целых чисел в памяти, используемый в большинстве современных процессоров?	дополнительный код / two's complement	ОПК-4
11.	Укажите, какое количество бит используется для хранения значения типа **float** по стандарту IEEE 754.	32	ОПК-4
12.	Как называется ошибка, возникающая при сложении двух очень малых вещественных чисел, которые не изменяют большее число из-за ограниченной точности?	потеря значимости / underflow / округление	ОПК-4
13.	Укажите команду языка C, используемую для выделения памяти на куче.	malloc	ОПК-5
14.	Какой регистр в архитектуре x86-64 используется для хранения адреса следующей инструкции?	RIP / EIP	ОПК-5
15.	Укажите, какой системный вызов используется в Linux для создания нового процесса.	fork	ОПК-5
16.	Как называется файл, содержащий объявления функций, используемых в C-программе?	заголовочный файл /	ОПК-5

		header file / .h файл	
17.	Укажите тип архитектуры команд, в которой набор инструкций прост и фиксирован по длине (например, ARM, RISC-V).	RISC / reduced instruction set computer	ПК-3
18.	Как называется инструкция в ассемблере, которая передаёт управление функции и сохраняет адрес возврата?	call	ПК-3
19.	Укажите, как называется область памяти, где хранятся локальные переменные и адреса возврата при вызове функций.	стек / stack	ПК-3
20.	Какой тип векторных инструкций используется в современных процессорах для параллельной обработки нескольких данных в одном регистре?	SIMD / SSE / AVX	ПК-3