Федеральное государственное бюджетное образовательное учреждение высшего образования
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПРАВОСУДИЯ»

Рабочая программа дисциплины
Системное программирование в среде Linux
(наименование дисциплины в соответствии с учебным планом)

Специальность: 09.02.07 Информационные системы и программирование
(код и наименование)

Рабочая программа разработана в соответствии с требованиями ФГОС.

Разработчик (-и): Новиков И.А.
(ФИО, ученая степень, ученое звание)

Зав. кафедрой _____ _____
(ФИО, ученая степень, ученое звание) (подпись)

Оглавление

Аннотация рабочей программы дисциплины «Системное программирование в среде Linux»

Разработчик: Новиков И.А.

Цель изучения дисциплины	Целью изучения дисциплины «Системное программирование в среде Linux» является формирование у студентов знаний о структуре и специфике системного программирования в среде Linux, а также навыков их эффективного использования для решения профессиональных задач в области проектирования и управления базами данных. Дисциплина способствует развитию профессиональных компетенций и личностного роста, что соответствует общим целям образовательной программы СПО
Место дисциплины в структуре ООП	Дисциплина «Системное программирование в среде Linux» относится к вариативной части профессионального цикла образовательной программы (ООП)
Компетенции, формируемые в результате освоения дисциплины	ПК 1.2: разрабатывать программные модули в соответствии с техническим заданием. ПК 1.3: выполнять отладку программных модулей с использованием специализированных программных средств.
Содержание дисциплины	Перечень тем: Тема 1. Введение в мировые информационные ресурсы: классификация и структура. Тема 2. Методы поиска и анализа информации в глобальной сети. Тема 3. Основные источники и базы данных для профессиональной деятельности. Тема 4. Анализ функциональной области для проектирования баз данных. Тема 5. Информационные ресурсы и их роль в проектировании баз данных. Тема 6. Технологии управления информационными ресурсами. Тема 7. Практическое использование информационных ресурсов в принятии решений баз данных. Тема 8. Перспективы и тенденции развития мировых информационных ресурсов.
Общая трудоемкость дисциплины	Общая трудоемкость дисциплины составляет 106 часов.
Форма промежуточной аттестации	Промежуточная аттестация проводится в форме КЗ, зачета.

1.Цели и планируемые результаты изучения дисциплины

Целью изучения дисциплины "Системное программирование в среде Linux" является формирование у студентов теоретических знаний и практических навыков в области разработки системного программного обеспечения для операционной системы Linux. В рамках курса студенты изучат основные концепции и методы системного

программирования, освоят инструменты и технологии разработки, отладки и оптимизации системного ПО, а также научатся эффективно использовать возможности операционной системы Linux для решения прикладных и системных задач.

В совокупности с другими дисциплинами ООП, данная дисциплина обеспечивает формирование следующих компетенций:

Таблица 1

№ п/п	Код компетенции	Название
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием.
2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

2. Место дисциплины в структуре ООП

Дисциплина «Системное программирование в среде Linux» является частью вариативного цикла образовательной программы среднего профессионального образования (СПО) и включена в обязательную часть данного цикла.

Она направлена на формирование профессиональных и общекультурных компетенций, необходимых для эффективного выполнения задач в области информационных технологий и проектирования баз данных.

В профессиональном цикле дисциплина «Системное программирование в среде Linux» играет ключевую роль в подготовке студентов направленных на формирование профессиональных компетенций, необходимых для разработки и сопровождения системного программного обеспечения..

Связь с другими дисциплинами ООП

Дисциплина "Системное программирование в среде Linux" тесно связана с другими дисциплинами учебного плана, обеспечивая комплексное и целостное освоение знаний и навыков, необходимых для профессиональной деятельности в области информационных технологий. такими как:

Операционные системы и среды: дисциплина предоставляет фундаментальные знания о принципах работы операционных систем, что является основой для изучения системного программирования в Linux. Эти взаимосвязи обеспечивают комплексное освоение студентами навыков и знаний, необходимых для эффективной работы с информационными ресурсами в их будущей профессиональной деятельности.

Архитектура аппаратных средств: понимание архитектуры компьютеров и их компонентов важно для эффективного использования возможностей аппаратного обеспечения при разработке системного ПО

Основы алгоритмизации и программирования: языки программирования являются основными для системного программирования в среде Linux.

Компьютерные сети: Знания о сетевых технологиях и протоколах важны для разработки сетевых приложений и системного ПО, взаимодействующего с сетью.

Технология разработки и защиты баз данных: умение работать с базами данных может быть полезно при разработке системных приложений, которые требуют взаимодействия с системами хранения данных.

Значение дисциплины в образовательной программе

Изучение дисциплины "Системное программирование в среде Linux" позволяет студентам:

1. Получить глубокие знания о принципах и методах разработки системного программного обеспечения.
2. Освоить навыки работы с операционной системой Linux, что является важным для многих современных ИТ-проектов.
3. Приобрести компетенции, необходимые для эффективного использования возможностей операционной системы при разработке прикладных и системных решений.
4. Подготовиться к решению практических задач, связанных с разработкой, отладкой и оптимизацией системного ПО, что существенно повышает их конкурентоспособность на рынке труда.

Дисциплина способствует укреплению междисциплинарных связей и интеграции знаний, полученных в других дисциплинах, обеспечивая целостное понимание роли и значения информационных ресурсов в профессиональной деятельности

3.Объем дисциплины (модуля) и виды учебной работы

Таблица 2.1.
Очная форма обучения

Вид учебной работы	**Трудоемкость**			
	зач. ед.	**час.**	**по семестрам**	
			6	7
Общая трудоемкость дисциплины по учебному плану	-	106	76	30
Контактная работа	-	100	72	28
Самостоятельная работа под контролем преподавателя, НИРС	-	6	4	2
Занятия лекционного типа	-	32	18	14
Занятия семинарского типа	-	68	54	14
в том числе с практической подготовкой (при наличии)	-			
Форма промежуточной аттестации	-		КЗ	зачет

4. Содержание дисциплины (модуля)

4.1. Текст рабочей программы по темам

Тема 1. Введение в системное программирование в среде Linux. Инструменты разработки для системного программирования

Понятие и цели системного программирования. Особенности операционной системы Linux. История и развитие Linux. Обзор возможностей и преимуществ Linux для системного программирования. Основные инструменты: GCC, GDB, Makefile. Использование текстовых редакторов и IDE (Vim, Emacs, Visual Studio Code). Управление версиями кода с использованием Git.

Тема 2. Введение в системные вызовы. Работа с файловой системой.

Понятие системных вызовов. Основные системные вызовы для работы с файлами: open, read, write, close. Управление процессами: fork, exec, wait. Обработка ошибок и код возврата. Структура файловой системы Linux. Работа с директориями: opendir, readdir, closedir. Управление правами доступа: chmod, chown, umask. Жесткие и символические ссылки: ln, unlink.

Тема 3. Управление процессами. Многопоточность в Linux.

Понятие процессов и потоков. Создание и управление процессами: fork, exec, wait. Межпроцессное взаимодействие (IPC): pipes, FIFOs, signals. Планирование процессов и управление приоритетами. Основы многопоточности. Создание и управление потоками: pthread_create, pthread_join. Синхронизация потоков: mutexes, condition variables, semaphores. Обработка гонок и мертвых блокировок.

Тема 4. Ввод/вывод и буферизация. Сетевое программирование в Linux

Системные вызовы для ввода/вывода. Буферизация ввода/вывода. Работа с терминалом и псевдотерминалами. Асинхронный ввод/вывод. Основы сетевого программирования. Сокеты: создание, настройка и использование. TCP и UDP: отличия и примеры использования. Разработка клиент-серверных приложений.

Тема 5. Модульное программирование и динамические библиотеки. Разработка и отладка модулей ядра

Преимущества модульного программирования. Создание и использование статических и динамических библиотек. Инструменты для работы с библиотеками: ar, nm, ldd. Управление зависимостями. Введение в разработку модулей ядра. Структура и компоненты модуля ядра. Компиляция и загрузка модулей: insmod, rmmod, modprobe. Отладка модулей ядра: printk, /proc, /sys.

Тема 6. Управление памятью. Безопасность системного программного обеспечения.

Модель памяти в Linux. Управление динамической памятью: malloc, free, realloc. Работа с разделяемой памятью: mmap, shmget, shmat. Оптимизация использования памяти. Основы безопасности программного обеспечения. Механизмы защиты памяти: ASLR, NX bit, stack canaries. Управление правами и доступом. Внедрение механизмов аутентификации и авторизации.

Тема 7. Средства мониторинга и профилирования. Автоматизация и скриптовые языки

Инструменты мониторинга: top, htop, vmstat, iostat. Профилирование производительности: gprof, perf. Анализ и оптимизация производительности. Основы написания скриптов на Bash. Автоматизация задач с использованием скриптов. Взаимодействие скриптов с системным ПО. Примеры использования Perl и Python для системного программирования.

Тема 8. Практические проекты и кейсы. Перспективы и направления развития системного программирования в Linux

Обзор реальных проектов системного программирования. Разработка и презентация собственных проектов. Обсуждение и анализ кейсов из практики. Современные тенденции в системном программировании. Новые технологии и инструменты. Возможности и вызовы для будущих разработчиков системного ПО.

4.2. Разделы и темы дисциплины, виды занятий (тематический план)

Таблица 3.1.

Тематический план

Очная форма обучения

№	Раздел дисциплины, тема	Код компетенции	Общая трудоёмкость дисциплины	в том числе					Наименование оценочного средства
				Контактная работа	Самостоятельная Работа под	Занятия лекционного типа	Занятия семинарского типа	Практическая подготовка	
			час.	час.	час.	час.	час.	час.	
1	Тема 1. Введение в системное программирование в среде Linux. Инструменты разработки для системного программирования	ПК 1.2 ПК 1.3	12	12		4	8		Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, деловая игра, составление проектов документов, вопросы к зачету
2	Тема 2. Введение в системные вызовы. Работа с файловой системой.	ПК 1.2 ПК 1.3	24	24		8	16		Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения,

									тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
3	Тема 3. Управление процессами. Многопоточность в Linux.	ПК 1.2 ПК 1.3	24	24		8	16		Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
4	Тема 4. Ввод/вывод и буферизация. Сетевое программирование в Linux	ПК 1.2 ПК 1.3	24	24		8	16		Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
5	Тема 5. Модульное программирование и динамические библиотеки.	ПК 1.2 ПК 1.3	20	16	4	4	12		Вопросы для семинаров (для проведения контрольного

	Разработка и отладка модулей ядра								опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
6	Тема 6. Управление памятью. Безопасность системного программного обеспечения.	ПК 1.2 ПК 1.3	20	20		4	16		Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
7	Тема 7. Средства мониторинга и профилирования. Автоматизация и скриптовые языки	ПК 1.2 ПК 1.3	14	14		4	10		Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление

									проектов документов, вопросы к зачету
8	Тема 8. Практические проекты и кейсы. Перспективы и направления развития системного программирования в Linux	ПК 1.2 ПК 1.3	16	14	2	4	10		Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
ВСЕГО			106	100	6	32	68	-	

Таблица 3.2.

4.3. Самостоятельное изучение обучающимися разделов дисциплины

Таблица 4.1.
Очная форма обучения

№ раздела (темы) дисциплины	Вопросы, выносимые на самостоятельное изучение	Кол-во часов
Тема 5. Модульное программирование и динамические библиотеки. Разработка и отладка модулей ядра	«Разрешительные» (permissive) лицензии: BSD, MIT, Apache. «Копилефтные» лицензии: GPL, AGPL. Общественное достояние (Public Domain): CC0, Unlicense, WTFPL. Сообщества Stack Overflow, GitHub.	4
Тема 8. Практические проекты и кейсы. Перспективы и направления развития системного программирования в Linux	Внесерверная обработка данных (FaaS). Расширение облачных вычислений. Всплеск периферийных вычислений. Квантовые вычисления: новый рубеж	2
Всего		**6**

4.4. Темы курсового проекта (курсовой работы)

Курсовой проект (курсовая работа) учебным планом не предусмотрен(а).

5. Учебно-методическое и информационное обеспечение дисциплины (модуля)

5.1. Учебно-методические рекомендации по изучению дисциплины (модуля)

Учебно-методические рекомендации для обучающихся по видам учебных занятий

Общие положения

Занятия со студентами проводятся в форме лекций, семинаров, практических занятий, лабораторных практикумов. Применяются теоретические знания на практике, выполняются проекты, связанные с интеграцией данных из различных источников. Участие в практических занятиях и кейсах, позволяют использовать информационные ресурсы для создания баз данных.

От студентов требуется посещение лекций и иных форм занятий. Особо ценится активная работа на семинарских и практических занятиях. Для успешной работы на семинарском и практическом занятии студент должен прочесть рекомендованную настоящим учебно-методическим комплексом литературу, нормативные правовые акты и активно участвовать в дискуссии, уметь изложить основные идеи прочитанных источников и дать им аргументированную оценку.

Занятия лекционного типа

Лекционные занятия – это устное систематическое и последовательное изложение учебного материала по темам дисциплины. Они являются организующим и ориентирующим началом для изучения дисциплины. В ходе лекций раскрываются основные положения онтологии данных, обращается внимание студентов на сложные теоретические и технологические вопросы, показывается их практическая значимость.

Лекционные занятия проводятся в форме информационной лекции (преподавателем сообщаются сведения, предназначенные для запоминания) и лекции-дискуссии (преподавателем ставятся на обсуждение студентов проблемные вопросы теории и технологической практики).

В ходе лекционных занятий используется аудиторный фонд Университета.

Занятия семинарского типа

Семинарские занятия проводятся с целью усвоения лекционного теоретического материала, углубления и расширения знаний студентов. На семинарах студенты учатся рассуждать, делать собственные выводы, аргументировано отстаивать свою точку зрения. На семинарских занятиях применяются следующие методы контрольного опроса, дискуссии.

Основная форма работы на семинаре – обсуждение всеми студентами вопросов, указанных в плане текущего семинарского занятия.

Практическая подготовка

Практические занятия проводятся с целью овладения студентами навыками применения полученных теоретических знаний для решения задач, возникающих в профессиональной деятельности. Применяются теоретические знания на практике, выполняются проекты, связанные с интеграцией данных из различных источников. Участие в практических занятиях и кейсах, позволяет использовать информационные ресурсы для создания баз данных.

Учебно-методические рекомендации по выполнению различных форм самостоятельной работы

Общие положения

К формам самостоятельной работы студентов относятся:

- выполнение самостоятельного поиска в выбранных базах данных по конкретной профессиональной теме.;
- конспектирование текстов дополнительной литературы;
- работа со словарями и справочниками;
- работа со справочными правовыми системами и ресурсами сети Интернет;
- подбор судебной практики по теме семинарского или практического занятия;
- решение ситуационных задач;
- составление проектов документов;
- участие в круглых столах, научно-практических конференциях;
- подготовка к сдаче зачета.

Учебно-методические рекомендации по изучению обучающимися вопросов, выносимых на самостоятельное изучение

Семинарские и практические занятия по дисциплине "Системное программирование в среде Linux" направлены на:

Обсуждение и более глубокое усвоение студентами наиболее сложных вопросов системного программирования.

Подробное изучение инструментов и технологий, используемых в разработке системного программного обеспечения.

Анализ специфики и проведение сравнения различных системных вызовов и функций операционной системы Linux.

Анализ и обобщение кейсов и примеров из практики системного программирования.

Для того чтобы знания, приобретенные при изучении учебной литературы и источников по системному программированию, стали глубже, необходимо ознакомиться с рекомендациями по изучению каждого раздела.

Рекомендации по изучению тем:

Тема 1: Введение в системное программирование в среде Linux. Инструменты разработки для системного программирования

Ознакомьтесь с базовыми понятиями и целями системного программирования. Рекомендуется использовать учебные ресурсы и онлайн-документацию Linux для изучения возможностей операционной системы. Важно создать конспект, охватывающий ключевые понятия и преимущества использования Linux для системного программирования. Практикуйтесь в использовании основных инструментов разработки, таких как GCC, GDB, Makefile. Необходимо сравнивать различные текстовые редакторы и IDE для выбора наиболее удобного инструмента. Рекомендуется освоить базовые команды Git для управления версиями кода.

Тема 2: Введение в системные вызовы. Работа с файловой системой.

Изучите основные системные вызовы для работы с файлами и процессами. Практикуйте написание простых программ, использующих системные вызовы open, read, write, close, fork, exec, wait. Важно понимать обработку ошибок и использование кодов возврата. Ознакомьтесь со структурой файловой системы Linux и основными командами для работы с директориями и файлами. Практикуйтесь в использовании системных вызовов для управления файлами и правами доступа. Рекомендуется изучить создание и управление жесткими и символическими ссылками.

Тема 3: Управление процессами. Многопоточность в Linux.

Изучите концепции процессов и потоков. Практикуйте создание и управление процессами с использованием fork и exec. Ознакомьтесь с методами межпроцессного взаимодействия (IPC), такими как pipes и signals. Важно понимать принципы планирования процессов и управления приоритетами. Ознакомьтесь с основами многопоточности и синхронизации потоков. Практикуйтесь в создании и управлении потоками с использованием pthread. Изучите методы синхронизации, такие как mutexes и semaphores,

и научитесь избегать гонок и мертвых блокировок.

Тема 4: Ввод/вывод и буферизация. Сетевое программирование в Linux

Изучите системные вызовы для ввода/вывода и принципы буферизации. Практикуйтесь в работе с терминалом и псевдотерминалами. Рекомендуется освоить асинхронный ввод/вывод для повышения эффективности программ. Ознакомьтесь с основами сетевого программирования и созданием сокетов. Практикуйтесь в разработке клиент-серверных приложений с использованием TCP и UDP. Важно понимать принципы работы с сетевыми протоколами и их реализацию в Linux.

Тема 5: Модульное программирование и динамические библиотеки. Разработка и отладка модулей ядра

Изучите преимущества модульного программирования и создание библиотек. Практикуйтесь в создании и использовании статических и динамических библиотек. Рекомендуется освоить инструменты для работы с библиотеками, такие как ar, nm, ldd. Ознакомьтесь с основами разработки модулей ядра и их структурой. Практикуйтесь в компиляции и загрузке модулей ядра. Изучите методы отладки модулей, такие как printk и использование файловых систем /proc и /sys.

Тема 6: Управление памятью. Безопасность системного программного обеспечения.

Изучите модель памяти в Linux и методы управления динамической памятью. Практикуйтесь в использовании системных вызовов для работы с разделяемой памятью. Важно понимать оптимизацию использования памяти для повышения производительности программ. Ознакомьтесь с основами безопасности программного обеспечения и методами защиты памяти. Изучите механизмы управления правами и доступом. Практикуйтесь во внедрении механизмов аутентификации и авторизации.

Тема 7: Средства мониторинга и профилирования. Автоматизация и скриптовые языки.

Изучите инструменты мониторинга системы и профилирования производительности. Практикуйтесь в использовании gprof и perf для анализа производительности программ. Важно понимать методы оптимизации производительности. Ознакомьтесь с основами написания скриптов на Bash и автоматизации задач. Практикуйтесь во взаимодействии скриптов с системным ПО. Рекомендуется изучить примеры использования Perl и Python для системного программирования.

Тема 8: Практические проекты и кейсы. Перспективы и направления развития системного программирования в Linux

Изучите реальные проекты системного программирования. Практикуйтесь в разработке и презентации собственных проектов. Важно анализировать и обсуждать кейсы из практики. Изучите современные тенденции в системном программировании. Ознакомьтесь с новыми технологиями и инструментами. Рекомендуется анализировать возможности и вызовы для будущих разработчиков системного ПО.

Учебно-методические рекомендации по выполнению отдельных форм самостоятельной работы

В ходе зачета обучающийся должен показать глубокое знание предмета, умение связывать теоретические знания с практикой. Обучающийся должен знать определения всех базовых понятий дисциплины, уметь характеризовать теоретические понятия в их практическом применении, уметь формулировать, обосновывать и излагать собственную точку зрения по дискуссионным вопросам, уметь проводить системные связи между понятиями и категориями, давать содержательно структурированный и последовательный ответ на поставленные в билете вопросы. Студент должен обладать навыками разработки системного ПО для различных задач.

Ответы на задания должны быть:

даны с использованием надежных источников информации, такие как научные

статьи, профессиональные базы данных и интернет-ресурсы;

даны на все поставленные вопросы;

развернутыми, аргументированными.

Работа обучающегося по подготовке к промежуточной аттестации должна быть направлена на тщательную проработку и усвоение лекционного материала, основной учебной и дополнительной литературы в соответствии с предлагаемым тематическим планом.

Учебно-методические рекомендации для обучающихся с ограниченными возможностями здоровья и инвалидами по освоению дисциплины

Под специальными условиями для получения среднего професионального образования по образовательным программам обучающимися с ограниченными возможностями здоровья понимаются условия обучения таких обучающихся, включающие в себя использование специальных образовательных программ и методов обучения и воспитания, специальных учебников, учебных пособий и дидактических материалов, специальных технических средств обучения коллективного и индивидуального пользования, предоставление услуг ассистента (помощника), оказывающего обучающимся необходимую техническую помощь, проведение групповых и индивидуальных коррекционных занятий, обеспечение доступа в здания организаций и другие условия, без которых невозможно или затруднено освоение образовательных программ обучающимися с ограниченными возможностями здоровья.

В освоении дисциплины инвалидами и лицами с ограниченными возможностями здоровья большое значение имеет индивидуальная работа. Под индивидуальной работой подразумевается две формы деятельности: самостоятельная работа по освоению и закреплению материала; индивидуальная учебная работа в контактной форме, предполагающая взаимодействие с преподавателем (в частности, консультации), т.е. дополнительное разъяснение учебного материала и углубленное изучение материала. Индивидуальные консультации по предмету являются важным фактором, способствующим индивидуализации обучения и установлению воспитательного контакта между преподавателем и обучающимся.

В целях освоения учебной программы дисциплины инвалидами и лицами с ограниченными возможностями здоровья возможно:

- использование специальных технических и иных средств индивидуального пользования, рекомендованных врачом-специалистом;

- присутствие ассистента, оказывающего обучающемуся необходимую помощь.

На лекционном занятии рекомендуется использовать звукозаписывающие устройства и компьютеры, как способ конспектирования.

Для освоения дисциплины (в т.ч. подготовки к занятиям, при самостоятельной работе) лицами с ограниченными возможностями здоровья предоставляется возможность использования учебной литературы в виде электронного документа в электронно-библиотечной системе Book.ru имеющей специальную версию для слабовидящих; обеспечивается доступ к учебно-методическим материалам посредством СЭО «Фемида»; доступ к информационным и библиографическим ресурсам посредством сети «Интернет».

5.2. Перечень нормативных правовых актов, актов высших судебных органов, материалов судебной практики

Не предусмотрено.

5.3. Информационное обеспечение изучения дисциплины (модуля)

Информационные, в том числе электронные ресурсы Университета, а также иные электронные ресурсы, необходимые для изучения дисциплины (модуля):

№ п./п.	Наименование	Адрес в сети Интернет
1.	ZNANIUM.COM	http://znanium.com Основная коллекция Коллекция издательства Статут Znanium.com. Discovery для аспирантов
2.	ЭБС ЮРАЙТ	www.biblio-online.ru
3.	ЭБС «BOOK.ru»	www.book.ru коллекция издательства Проспект Юридическая литература ; коллекции издательства Кнорус Право, Экономика и Менеджмент
4.	EastViewInformationServices	www.ebiblioteka.ru Универсальная база данных периодики (электронные журналы)
5.	НЦР РУКОНТ	http://rucont.ru/ Раздел Ваша коллекция - РГУП-периодика (электронные журналы)
6.	OxfordBibliographies	www.oxfordbibliographies.com **модуль Management** –аспирантура Экономика и **модуль InternationalLaw**-аспирантура Юриспруденция
7.	Информационно-образовательный портал РГУП	www.op.raj.ru электронные версии учебных, научных и научно-практических изданий РГУП
8.	Система электронного обучения «Фемида»	www.femida.raj.ru Учебно-методические комплексы, Рабочие программы по направлению подготовки
9.	Правовые системы	Гарант, Консультант

Основная и дополнительная литература указана в Карте обеспеченности литературой.

6. Материально-техническое обеспечение

Для материально-технического обеспечения дисциплины используются специальные помещения. Специальные помещения представляют собой учебные аудитории для проведения занятий лекционного типа, занятий семинарского типа, курсового проектирования (выполнения курсовых работ), групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, а также помещения для самостоятельной работы и помещения для хранения и профилактического обслуживания учебного оборудования.

Для проведения занятий лекционного типа предлагаются наборы демонстрационного оборудования и учебно-наглядных пособий, обеспечивающие тематические иллюстрации, соответствующие рабочим программам дисциплин. Демонстрационное оборудование представлено в виде мультимедийных средств. Учебно-наглядные пособия представлены в виде экранно-звуковых средств, печатных пособий, слайд-презентаций, видеофильмов, макетов и т.д., которые применяются по необходимости в соответствии с темами (разделами) дисциплины.

Для самостоятельной работы обучающихся помещения оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду университета.

Предусмотрены помещения для хранения и профилактического обслуживания учебного оборудования.

Перечень специальных помещений ежегодно обновляется и отражается в справке о материально-техническом обеспечении основной образовательной программы.

Состав необходимого комплекта лицензионного программного обеспечения ежегодно обновляется, утверждается и отражается в справке о материально-техническом обеспечении основной образовательной программы.

№ п/п	Наименование дисциплины (модуля) в соответствии с учебным планом	Наименование специальных помещений и помещений для самостоятельной работы
1.	Системное программирование в системе Linux	Студия разработки дизайна веб-приложений. (ИЛК-1 (помещение 1001)-9 (330))

7. Карта обеспеченности литературой

Кафедра информационного права, информатики и математики
Специальность: 09.02.07 Информационные системы и программирование
Дисциплина: Системное программирование в среде Linux
Курс:2

Полное библиографическое описание
Основная литература
Гунько А.В. Системное программирование в среде Linux : учебное пособие / Гунько А.В.. — Новосибирск : Новосибирский государственный технический университет, 2020. — 235 c. — ISBN 978-5-7782-4160-2. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/98735.html (дата обращения: 25.06.2024). — Режим доступа: для авторизир. пользователей
Токманцев, Т. Б. Алгоритмические языки и программирование : учебное пособие для СПО / Т. Б. Токманцев ; под редакцией В. Б. Костоусова. — 3-е изд. — Саратов, Екатеринбург : Профобразование, Уральский федеральный университет, 2024. — 102 c. — ISBN 978-5-4488-0510-3, 978-5-7996-2899-4. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139513.html (дата обращения: 23.05.2024). — Режим доступа: для авторизир. пользователей
Дополнительная литература
Губарь, Ю. В. Введение в математическое программирование : учебное пособие для СПО / Ю. В. Губарь. — 2-е изд. — Саратов : Профобразование, 2024. — 225 c. — ISBN 978-5-4488-0992-7. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139745.html (дата обращения: 27.05.2024). — Режим доступа: для авторизир. пользователей
Моренкова, О. И. Операционные системы. Linux : учебное пособие для СПО / О. И. Моренкова, А. Ю. Голошубов. — Саратов : Профобразование, 2024. — 108 c. — ISBN 978-5-4488-1864-6. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139041.html (дата обращения: 07.05.2024). — Режим доступа: для авторизир. пользователей

Зав. библиотекой _____О.В. Астраханцева Зав. кафедрой___ _____

8. Фонд оценочных средств

8.1. Паспорт фонда оценочных средств по дисциплине

№ п/п	Раздел дисциплины, тема	Код компетенции	Наименование оценочного средства
1.	Тема 1. Введение в системное программирование в среде Linux. Инструменты разработки для системного программирования	ПК 1.2 ПК 1.3	Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
2.	Тема 2. Введение в системные вызовы. Работа с файловой системой.	ПК 1.2 ПК 1.3	Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
3.	Тема 3. Управление процессами. Многопоточность в Linux.	ПК 1.2 ПК 1.3	Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
4.	Тема 4. Ввод/вывод и буферизация. Сетевое программирование в Linux	ПК 1.2 ПК 1.3	Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания

			для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
5.	Тема 5. Модульное программирование и динамические библиотеки. Разработка и отладка модулей ядра	ПК 1.2 ПК 1.3	Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
6.	Тема 6. Управление памятью. Безопасность системного программного обеспечения.	ПК 1.2 ПК 1.3	Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
7.	Тема 7. Средства мониторинга и профилирования. Автоматизация и скриптовые языки	ПК 1.2 ПК 1.3	Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету
8.	Тема 8. Практические проекты и кейсы. Перспективы и направления развития системного	ПК 1.2 ПК 1.3	Вопросы для семинаров (для проведения контрольного опроса и дискуссий), задания

	программирования в Linux		для контрольной работы для очно-заочной формы обучения, тестовые задания, темы докладов, составление проектов документов, вопросы к зачету

8.2. Оценочные средства

Деловая (ролевая) игра для изучения дисциплины "Системное программирование в среде Linux"

Код компетенции

№ п/п	Код компетенции	Наименование
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием
2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

1. Тема (проблема): «Разработка и отладка системного программного обеспечения для управления ресурсами сервера»

Современные компании используют серверные решения для управления различными ресурсами и услугами. Участникам игры предлагается разработать и отладить системное программное обеспечение, которое будет эффективно управлять ресурсами сервера, такими как процессорное время, память, сетевые соединения и файловая система.

2. Концепция игры

Игра предполагает работу в командах, каждая из которых отвечает за разработку и отладку определенных модулей системного ПО. Основная задача участников – разработать программные модули в соответствии с техническим заданием и выполнить их отладку с использованием специализированных инструментов.

Фаза 1: Определение требований – Команды получают техническое задание и анализируют требования к программным модулям. **Фаза 2: Разработка модулей** – Команды разрабатывают программные модули на языке C/C++. **Фаза 3: Отладка модулей** – Команды выполняют отладку разработанных модулей с использованием инструментов, таких как GDB и Valgrind. **Фаза 4: Презентация и обсуждение** – Команды представляют свои модули, объясняют выбранные подходы и решения. Обсуждаются сильные и слабые стороны предложенных решений.

3. Роли

1. **Менеджер проекта**:
   - Обязанности: координация работы команды и связь с другими командами, осуществляет контроль за выполнением задач и соблюдением сроков.
   - Советы: постоянно проверяйте прогресс команды и корректируйте план работы при необходимости, организуйте регулярные встречи для обсуждения статуса проекта и решения возникающих проблем.
2. **Аналитик требований**:
   - Обязанности: сбор и анализ требований к программным модулям, разработка схем и диаграмм данных, оценка качества и релевантности данных.

- Советы: используйте интервью и опросы для сбора требований от различных отделов, применяйте техники визуализации для лучшего понимания структуры данных.

3. **Разработчик**:
   - Обязанности: разработка программных модулей в соответствии с техническим заданием, определение структуры кода и его взаимосвязей.
   - Советы: используйте специализированные инструменты для разработки (например, Visual Studio Code), убедитесь, что код соответствует собранным требованиям.
4. **Тестировщик**:
   - Обязанности: выполнение отладки программных модулей, тестирование и оценка удобства использования разработанных модулей.
   - Советы: используйте инструменты для отладки, такие как GDB и Valgrind, представляйте себя в роли конечного пользователя системы и определите, какие функции будут наиболее полезными.

4. Ожидаемый(е) результат(ы)

1. **Разработанные модули** – каждая команда должна представить свои программные модули с описанием их функциональности.
2. **Документ требований** – включает все собранные требования и обоснование выбора архитектуры и решений.
3. **Презентация проекта** – команды представляют свои модули и объясняют, как они учитывают потребности всех отделов и обеспечивают эффективное управление ресурсами сервера.
4. **Оценка модулей** – команды оценивают сильные и слабые стороны предложенных решений, обсуждают возможные улучшения и интеграцию.

5. Методические материалы по проведению

1. Инструкция по проведению деловой игры

Описание целей и задач игры: разработать программные модули для управления ресурсами сервера, учитывая потребности различных отделов. Задачи игры: собрать и проанализировать требования к программным модулям, создать и отладить программные модули, презентовать разработанные модули и обосновать их выбор, оценить и обсудить модули, предложенные другими командами.

Введение в тему и проблему:

Игровой сценарий предполагает, что участники являются сотрудниками компании, занимающейся разработкой серверного ПО. Компания расширяется, и ей необходимо новое программное обеспечение для эффективного управления ресурсами сервера.

Пошаговое руководство по каждому этапу игры:

Фаза 1: Определение требований

Команды собирают требования от всех отделов компании.

Участники обсуждают, какие функции необходимы для управления ресурсами сервера и как они будут использоваться.

Фаза 2: Разработка модулей

Команды разрабатывают программные модули на языке C/C++.

Определяются основные функции и взаимосвязи между модулями.

Фаза 3: Отладка модулей

Команды выполняют отладку разработанных модулей с использованием GDB и Valgrind.

Исправляются найденные ошибки и оптимизируется производительность.

Фаза 4: Презентация и обсуждение

Команды представляют свои модули и объясняют выбранные решения.

Проводится обсуждение сильных и слабых сторон каждого модуля.

Фаза 5: Оценка и выводы

Участники оценивают предложенные модули и обсуждают возможные улучшения.

Подведение итогов игры и обсуждение полученных уроков.

2. Материалы для подготовки

Шаблоны для сбора и анализа требований:

Шаблон требований: название модуля, описание функций, частота обновления данных, требования к доступу и безопасности данных, особые требования (например, интеграция с другими модулями).

Примеры концептуальных моделей программных модулей:

- Пример модуля 1: Модуль управления памятью (функции: выделение, освобождение, мониторинг).
- Пример модуля 2: Модуль управления процессами (функции: создание, завершение, планирование).

Описание основных функций и компонентов программных модулей:

Функции: Основные действия, выполняемые программными модулями.

Интерфейсы: Способы взаимодействия между модулями.

Ошибки: Обработка и обработка ошибок.

3. Руководство для ролей

Менеджер проекта:

Обязанности: координация работы команды и поддержание связи с другими командами, обеспечение соблюдения сроков и качества выполнения задач, руководство процессом презентации и обсуждения программных модулей.

Советы: постоянно проверяйте прогресс команды и корректируйте план работы при необходимости, организуйте регулярные встречи для обсуждения статуса проекта и решения возникающих проблем.

Аналитик требований:

Обязанности: сбор и анализ требований к программным модулям, разработка схем и диаграмм данных, оценка качества и релевантности данных.

Советы: используйте интервью и опросы для сбора требований от различных отделов, применяйте техники визуализации для лучшего понимания структуры данных.

Разработчик:

Обязанности: разработка программных модулей в соответствии с техническим заданием, определение структуры кода и его взаимосвязей.

Советы: используйте специализированные инструменты для разработки (например, Visual Studio Code), убедитесь, что код соответствует собранным требованиям.

Тестировщик:

Обязанности: выполнение отладки программных модулей, тестирование и оценка удобства использования разработанных модулей.

Советы: используйте инструменты для отладки, такие как GDB и Valgrind, представляйте себя в роли конечного пользователя системы и определите, какие функции будут наиболее полезными.

4. Презентационные материалы

Шаблоны для подготовки и представления программных модулей:

Шаблон слайдов:

- Титульный слайд: название проекта, команда, дата.
- Введение: обзор темы и цели проекта.
- Анализ требований: основные функции и потребности.
- Модуль программного обеспечения: схема и описание функций.
- Отладка: используемые инструменты и результаты.
- Заключение: итоги проекта и предложения по улучшению.
- Вопросы и ответы: время для обсуждения и вопросов от аудитории.

5. Оценочные критерии

Чек-листы для оценки программных модулей:

Чек-лист для оценки модуля:

- Соответствие модуля собранным требованиям.
- Логическая целостность и взаимосвязь функций.
- Оптимальность структуры кода для выполнения задач.
- Удобство использования и масштабируемость модуля.
- Соответствие стандартам безопасности и защиты данных.

Критерии оценки презентации и обсуждения:

Чек-лист для оценки презентации:

- Четкость и логичность изложения материала.
- Качество визуальных материалов и их соответствие содержанию.
- Уверенность и ясность ораторского искусства.
- Способность отвечать на вопросы и участвовать в обсуждении.
- Вклад каждого участника команды в презентацию.

Опросники для обратной связи от участников игры:

Вопросы для обратной связи:

- Как вы оцениваете полезность и интересность игры?
- Какие аспекты игры были наиболее сложными для вас?
- Что бы вы предложили улучшить в организации и проведении игры?
- Оцените свою команду и её вклад в успех проекта.
- Как игра помогла вам лучше понять процесс разработки и отладки программных модулей?

6. Технические ресурсы

Доступ к программному обеспечению для разработки и отладки программных модулей, таким как GCC, GDB, Valgrind.

7. Критерии оценивания:

Критерии	Баллы
Обучающийся выполняет разработку и отладку модуля полностью и без ошибок	2
Обучающийся выполняет разработку и отладку модуля с небольшими ошибками, которые успешно исправляет	1.5
Обучающийся выполняет разработку и отладку модуля с существенными ошибками	1
Обучающийся не выполняет разработку и отладку модуля	0

Вопросы для занятий семинарского типа (семинаров, коллоквиумов) для изучения дисциплины «Системное программирование в среде Linux»

Код компетенции

№ п/п	Код компетенции	Наименование
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием
2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

Тема семинара: Введение в системное программирование в среде Linux

№ п/п	Вопросы	Код компетенции (части) компетенции
1	Понятие и цели системного программирования в Linux.	ПК 1.2
2	История и развитие операционной системы Linux.	ПК 1.2
3	Основные компоненты и архитектура Linux.	ПК 1.2
4	Важность системного программирования для управления ресурсами и эффективного использования ОС.	ПК 1.2
5	Примеры системного программного обеспечения и его применение.	ПК 1.2

Тема семинара: Инструменты разработки для системного программирования

№ п/п	Вопросы	Код компетенции (части) компетенции
1	Основные инструменты для разработки в Linux: GCC, GDB, Makefile.	ПК 1.2
2	Использование текстовых редакторов и IDE (Vim, Emacs, Visual Studio Code).	ПК 1.2
3	Управление версиями кода с использованием Git.	ПК 1.2
4	Принципы и примеры использования автоматизации сборки проектов.	ПК 1.2
5	Сравнение различных инструментов разработки и отладки.	ПК 1.2

Тема семинара: Введение в системные вызовы

№ п/п	Вопросы	Код компетенции (части) компетенции
1	Понятие и виды системных вызовов в Linux.	ПК 1.2
2	Основные системные вызовы для работы с файлами: open, read, write, close.	ПК 1.2
3	Управление процессами с использованием системных вызовов: fork, exec, wait.	ПК 1.2
4	Обработка ошибок и код возврата в системных вызовах.	ПК 1.2
5	Примеры программ, использующих системные вызовы, и их анализ.	ПК 1.2

Тема семинара: Работа с файловой системой

№ п/п	Вопросы	Код компетенции (части) компетенции
1	Структура файловой системы Linux и её основные компоненты.	ПК 1.2

2	Системные вызовы для работы с файлами и директориями.	ПК 1.2
3	Управление правами доступа к файлам и директориям.	ПК 1.2
4	Создание и управление жесткими и символическими ссылками.	ПК 1.2
5	Примеры программ для работы с файловой системой и их анализ.	ПК 1.2

Тема семинара: Управление процессами

№ п/п	**Вопросы**	**Код компетенции (части) компетенции**
1	Понятие процессов и потоков в Linux.	ПК 1.2, ПК 1.3
2	Системные вызовы для управления процессами: fork, exec, wait.	ПК 1.2, ПК 1.3
3	Межпроцессное взаимодействие (IPC): pipes, FIFOs, signals.	ПК 1.2, ПК 1.3
4	Планирование процессов и управление приоритетами.	ПК 1.2, ПК 1.3
5	Примеры программ для управления процессами и их анализ.	ПК 1.2, ПК 1.3

Тема семинара: Многопоточность в Linux

№ п/п	**Вопросы**	**Код компетенции (части) компетенции**
1	Основы многопоточности и её преимущества.	ПК 1.2, ПК 1.3
2	Создание и управление потоками: pthread_create, pthread_join.	ПК 1.2, ПК 1.3
3	Синхронизация потоков: mutexes, condition variables, semaphores.	ПК 1.2, ПК 1.3
4	Обработка гонок и мертвых блокировок в многопоточных приложениях.	ПК 1.2, ПК 1.3
5	Примеры многопоточных программ и их анализ.	ПК 1.2, ПК 1.3

Тема семинара: Ввод/вывод и буферизация

№ п/п	**Вопросы**	**Код компетенции (части) компетенции**
1	Системные вызовы для ввода/вывода и их особенности.	ПК 1.2
2	Принципы буферизации ввода/вывода.	ПК 1.2
3	Работа с терминалом и псевдотерминалами.	ПК 1.2
4	Асинхронный ввод/вывод и его применение.	ПК 1.2
5	Примеры программ для ввода/вывода и их анализ.	ПК 1.2

Тема семинара: Сетевое программирование в Linux

№ п/п	**Вопросы**	**Код компетенции (части) компетенции**
1	Основы сетевого программирования и создание сокетов.	ПК 1.2
2	Разработка клиент-серверных приложений: TCP и UDP.	ПК 1.2
3	Сетевые протоколы и их реализация в Linux.	ПК 1.2
4	Примеры сетевых программ и их анализ.	ПК 1.2

5	Отладка и тестирование сетевых приложений.	ПК 1.3

Тема семинара: Модульное программирование и динамические библиотеки

№ п/п	**Вопросы**	**Код компетенции (части) компетенции**
1	Преимущества модульного программирования в Linux.	ПК 1.2
2	Создание и использование статических и динамических библиотек.	ПК 1.2
3	Инструменты для работы с библиотеками: ar, nm, ldd.	ПК 1.2
4	Управление зависимостями в программных проектах.	ПК 1.2
5	Примеры использования библиотек и их анализ.	ПК 1.2

Тема семинара: Разработка и отладка модулей ядра

№ п/п	**Вопросы**	**Код компетенции (части) компетенции**
1	Основы разработки модулей ядра и их структура.	ПК 1.2, ПК 1.3
2	Компиляция и загрузка модулей ядра: insmod, rmmod, modprobe.	ПК 1.2, ПК 1.3
3	Отладка модулей ядра: printk, /proc, /sys.	ПК 1.3
4	Примеры модулей ядра и их анализ.	ПК 1.2, ПК 1.3
5	Важность тестирования и отладки модулей ядра.	ПК 1.3

Тема семинара: Управление памятью

№ п/п	**Вопросы**	**Код компетенции (части) компетенции**
1	Модель памяти в Linux и её компоненты.	ПК 1.2
2	Управление динамической памятью: malloc, free, realloc.	ПК 1.2
3	Работа с разделяемой памятью: mmap, shmget, shmat.	ПК 1.2
4	Оптимизация использования памяти в приложениях.	ПК 1.2
5	Примеры программ для управления памятью и их анализ.	ПК 1.2

Тема семинара: Безопасность системного программного обеспечения

№ п/п	**Вопросы**	**Код компетенции (части) компетенции**
1	Основы безопасности программного обеспечения в Linux.	ПК 1.2
2	Механизмы защиты памяти: ASLR, NX bit, stack canaries.	ПК 1.2
3	Управление правами и доступом к системным ресурсам.	ПК 1.2
4	Внедрение механизмов аутентификации и авторизации.	ПК 1.2
5	Примеры программ для обеспечения безопасности и их анализ.	ПК 1.2

Тема семинара: Средства мониторинга и профилирования

№ п/п	Вопросы	Код компетенции (части) компетенции
1	Инструменты мониторинга системы: top, htop, vmstat, iostat.	ПК 1.3
2	Профилирование производительности программ с помощью gprof и perf.	ПК 1.3
3	Анализ и оптимизация производительности программ.	ПК 1.3
4	Примеры использования инструментов мониторинга и профилирования.	ПК 1.3
5	Важность мониторинга и профилирования в системном программировании.	ПК 1.3

Тема семинара: Автоматизация и скриптовые языки

№ п/п	Вопросы	Код компетенции (части) компетенции
1	Основы написания скриптов на Bash и их использование.	ПК 1.2
2	Автоматизация задач с использованием скриптовых языков.	ПК 1.2
3	Взаимодействие скриптов с системным ПО.	ПК 1.2
4	Примеры использования Perl и Python для системного программирования.	ПК 1.2
5	Примеры скриптов и их анализ.	ПК 1.2

Тема семинара: Практические проекты и кейсы

№ п/п	Вопросы	Код компетенции (части) компетенции
1	Обзор реальных проектов системного программирования.	ПК 1.2, ПК 1.3
2	Разработка и презентация собственных проектов.	ПК 1.2, ПК 1.3
3	Анализ кейсов из практики системного программирования.	ПК 1.2, ПК 1.3
4	Обсуждение решений и подходов к разработке системного ПО.	ПК 1.2, ПК 1.3
5	Оценка и улучшение предложенных проектов.	ПК 1.2, ПК 1.3

Тема семинара: Перспективы и направления развития системного программирования в Linux

№ п/п	Вопросы	Код компетенции (части) компетенции
1	Современные тенденции в системном программировании.	ПК 1.2
2	Новые технологии и инструменты для системного программирования.	ПК 1.2
3	Возможности и вызовы для будущих разработчиков системного ПО.	ПК 1.2
4	Влияние открытого исходного кода на развитие системного программирования.	ПК 1.2
5	Перспективы развития Linux и его влияние на системное программирование.	ПК 1.2

Критерии оценивания:

Критерии	Баллы

Ответ дан полностью, ответ верный, ответ дан с использованием актуальных знаний и практических примеров, обучающийся точно использует профессиональную терминологию, ответ изложен последовательно, развернутый, аргументированный.	2
Ответ дан полностью, ответ верный, обучающийся точно использует профессиональную терминологию, но ответ изложен непоследовательно, ответ односложный, неаргументированный, не в полной мере использована терминология.	1.5
Ответ дан не полностью либо ответ частично верный, изложен непоследовательно, ответ односложный, неаргументированный, не использована терминология.	1
Ответ неверный либо ответ отсутствует.	0

Комплект задач репродуктивного уровня

Код компетенции

№ п/п	Код компетенции	Наименование
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием
2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

Задачи репродуктивного уровня

Задача 1

Ситуация: Компания "СофтЛинукс" разрабатывает новое программное обеспечение для мониторинга системных ресурсов сервера. Одной из задач является создание модуля для мониторинга использования процессора и памяти.

Вопросы:

1. Какой системный вызов можно использовать для получения информации об использовании процессора и памяти?
2. Напишите пример кода на C, который использует данный системный вызов для получения информации о текущем использовании процессора.
3. Какие дополнительные функции могут быть включены в модуль для расширения его функциональности?

Код компетенции (части) компетенции: ПК 1.2, ПК 1.3

Задача 2

Ситуация: Студенту Ивану необходимо разработать модуль для управления файловой системой в Linux, который будет создавать, удалять и перемещать файлы и директории.

Вопросы:

1. Какие системные вызовы необходимо использовать для создания, удаления и перемещения файлов и директорий?
2. Напишите пример кода на C, который создает новый файл и записывает в него данные.
3. Как отлаживать и тестировать разработанный модуль для управления файловой системой?

Код компетенции (части) компетенции: ПК 1.2, ПК 1.3

Задача 3

Ситуация: Организация "ТехПроект" разрабатывает систему для управления процессами на сервере. Им необходимо создать модуль, который будет запускать новые процессы и отслеживать их состояние.

Вопросы:

1. Какие системные вызовы используются для создания и отслеживания процессов в Linux?
2. Напишите пример кода на C, который создает новый процесс с использованием fork() и exec().
3. Как можно реализовать отслеживание состояния процессов и их завершение?

Код компетенции (части) компетенции: ПК 1.2, ПК 1.3

Задача 4

Ситуация: Компания "Сетевые Решения" разрабатывает сетевое приложение для обмена сообщениями между серверами. Необходимо создать модуль для работы с сокетами.

Вопросы:

1. Какие системные вызовы используются для работы с сокетами в Linux?
2. Напишите пример кода на C, который создает серверный сокет и принимает соединения от клиентов.

3. Как можно отладить и протестировать сетевой модуль для обеспечения его надежности?

Код компетенции (части) компетенции: ПК 1.2, ПК 1.3

Задача 5

Ситуация: Группа студентов разрабатывает приложение для управления многозадачностью в Linux. Им необходимо создать модуль, который будет управлять потоками.

Вопросы:

1. Какие библиотеки и системные вызовы используются для управления потоками в Linux?
2. Напишите пример кода на C, который создает и управляет потоками с использованием pthread.
3. Какие методы можно использовать для синхронизации потоков и предотвращения гонок?

Код компетенции (части) компетенции: ПК 1.2, ПК 1.3

Задача 6

Ситуация: Библиотека университета хочет создать программу для резервного копирования файлов. Необходимо разработать модуль, который будет выполнять резервное копирование и восстановление данных.

Вопросы:

1. Какие системные вызовы можно использовать для копирования файлов и директорий в Linux?
2. Напишите пример кода на C, который выполняет резервное копирование файла.
3. Как можно обеспечить целостность данных при выполнении резервного копирования?

Код компетенции (части) компетенции: ПК 1.2, ПК 1.3

Задача 7

Ситуация: Технологическая компания разрабатывает систему для автоматического сбора и анализа логов. Им необходимо создать модуль для работы с файлами логов.

Вопросы:

1. Какие системные вызовы и библиотеки можно использовать для работы с файлами логов в Linux?
2. Напишите пример кода на C, который читает данные из файла лога и выводит их на экран.
3. Как можно отладить и оптимизировать модуль для работы с большими объемами данных?

Код компетенции (части) компетенции: ПК 1.2, ПК 1.3

Задача 8

Ситуация: Некоммерческая организация "ЭкоПроект" создает приложение для мониторинга экологических данных. Необходимо разработать модуль для работы с сенсорными данными.

Вопросы:

1. Какие системные вызовы можно использовать для чтения данных с сенсоров в Linux?
2. Напишите пример кода на C, который читает данные с сенсора и сохраняет их в файл.
3. Какие методы можно использовать для отладки и тестирования модуля для работы с сенсорами?

Код компетенции (части) компетенции: ПК 1.2, ПК 1.3

Критерии оценивания:

Критерии	**Баллы**
Умение не сформировано	0
Умение сформировано частично	1,0

Умение сформировано, но имеет несущественные недостатки	1,5
Умение сформировано полностью	2

Инструкция и/или методические рекомендации по выполнению

Решение задач репродуктивного уровня направлено на развитие у студентов навыков практического применения знаний в области системного программирования для решения конкретных задач. Для успешного выполнения задачи студенту следует внимательно ознакомиться с её текстом, понять суть представленной проблемы и определить ключевые аспекты, которые требуют анализа и решения. Важно рассмотреть три основных направления: информационное, аналитическое и технологическое.

1. Начните с внимательного прочтения текста задачи. Определите ключевые элементы задачи: какие системные вызовы задействованы, каковы цели и контекст их использования. Проговорите для себя ситуацию, обдумайте, какие системные вызовы и технологии обсуждаются и какие проблемы необходимо решить.
2. Первое направление, которое нужно рассмотреть, это информационное. Определите, какие системные вызовы рассматриваются в задаче. Определите их типы и функции, такие как работа с файлами, процессами, потоками и сетями. Проанализируйте, насколько актуальны и надежны эти системные вызовы для решения представленной задачи. Убедитесь, что используемые системные вызовы соответствуют потребностям ситуации и цели их применения.
3. Следующее направление — аналитическое. Выявите основную проблему, описанную в задаче, и определите её ключевые аспекты. Проанализируйте, какие данные и информация необходимы для решения этой проблемы. Рассмотрите, как можно анализировать и интерпретировать доступные данные для понимания проблемы. Используйте подходящие методы анализа данных, такие как логический анализ или отладка кода, для получения полезных инсайтов.
4. Третье направление — технологическое. Определите, какие технологии и инструменты могут быть использованы для управления и обработки системных вызовов в контексте задачи. Рассмотрите, как можно применить эти технологии для эффективного решения проблемы. Проанализируйте, как можно интегрировать различные системные вызовы и инструменты для достижения наилучших результатов. Оцените, какие технологические подходы будут наиболее эффективными для решения поставленной задачи.

После анализа по трём направлениям перейдите к применению знаний и формулированию ответа. Определите необходимые информационные и технологические решения, которые подходят для данной задачи. Примените свои знания о системных вызовах и методах программирования для решения представленной проблемы. Сформулируйте чёткий и обоснованный ответ на поставленную задачу. Объясните, почему выбраны те или иные системные вызовы и технологические подходы. Укажите, как предложенное решение поможет эффективно справиться с поставленной проблемой.

Для оценки выполнения задач будут учитываться следующие критерии: насколько точно студент понимает суть проблемы и ключевые аспекты, требующие решения; способность студента оценить релевантность и надёжность используемых системных вызовов; насколько эффективно студент применяет методы анализа данных для решения задачи; способность предложить и обосновать использование подходящих технологических решений; ясность, логичность и обоснованность предложенного решения.

Темы докладов

Код компетенции

№ п/п	Код компетенции	Наименование
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием
2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

Перечень тем докладов:

№ п/п	Тема	Код компетенции (части) компетенции
1	История развития операционной системы Linux	ПК 1.2
2	Основные компоненты и архитектура Linux	ПК 1.2
3	Системные вызовы в Linux: основные понятия и применение	ПК 1.2, ПК 1.3
4	Разработка и отладка модулей ядра в Linux	ПК 1.2, ПК 1.3
5	Управление процессами в Linux: создание и завершение процессов	ПК 1.2, ПК 1.3
6	Многопоточность в Linux: использование pthreads	ПК 1.2, ПК 1.3
7	Работа с файловой системой в Linux: основные команды и системные вызовы	ПК 1.2, ПК 1.3
8	Управление памятью в Linux: malloc, free, mmap	ПК 1.2
9	Сетевое программирование в Linux: создание и управление сокетами	ПК 1.2
10	Инструменты разработки в Linux: GCC, GDB, Makefile	ПК 1.2
11	Автоматизация сборки проектов в Linux: использование Makefile	ПК 1.2
12	Обеспечение безопасности системного программного обеспечения в Linux	ПК 1.2, ПК 1.3
13	Профилирование и мониторинг производительности программ в Linux	ПК 1.3
14	Работа с межпроцессным взаимодействием (IPC) в Linux	ПК 1.2, ПК 1.3
15	Использование динамических библиотек в Linux: создание и подключение	ПК 1.2
16	Отладка и оптимизация многопоточных приложений в Linux	ПК 1.3
17	Асинхронный ввод/вывод в Linux: концепции и применение	ПК 1.2, ПК 1.3
18	Интеграция C и C++ программ в среде Linux	ПК 1.2
19	Использование системных журналов для отладки программ в Linux	ПК 1.3
20	Управление устройствами и драйверами в Linux	ПК 1.2
21	Разработка системных утилит для Linux	ПК 1.2, ПК 1.3
22	Инструменты для работы с исходным кодом в Linux: Git и другие	ПК 1.2
23	Разработка и отладка сетевых служб в Linux	ПК 1.2, ПК 1.3
24	Обеспечение совместимости и переносимости кода в разных дистрибутивах Linux	ПК 1.2

Критерии оценивания:

Критерии	Баллы
Доклад соответствует заявленной теме, целям и задачам; материал представлен и освещен логично и последовательно; тема раскрыта полно; источники достоверны и релевантны, автор использует широкий круг источников и фактического материала; выводы автора обоснованы; презентация соответствует рекомендациям (в т.ч. оформление).	1,5
Доклад соответствует заявленной теме, целям и задачам; материал представлен и освещен логично и последовательно, но есть отдельные отступления; тема раскрыта практически полно; источники достоверны и релевантны, но есть неточности; автор использует достаточный круг источников и фактического материала; выводы автора обоснованы; презентация соответствует рекомендациям (в т.ч. оформление).	1,0
Доклад в целом соответствует заявленной теме, целям и задачам; материал представлен и освещен не совсем логично и последовательно; тема раскрыта неполно; источники достоверны и релевантны, но есть ошибки; автор использует узкий круг источников и фактического материала; выводы автора недостаточно обоснованы; презентация соответствует рекомендациям (в т.ч. оформление) только в части.	0,5
Доклад не соответствует заявленной теме, целям и задачам; материал представлен и освещен нелогично и непоследовательно; тема не раскрыта; источники недостоверны и нерелевантны; автор использует только два-три источника, не анализирует и фактический материал; выводы автора не обоснованы; презентация не соответствует рекомендациям (в т.ч. оформление).	0

Методические рекомендации по подготовке доклада

Доклад – это устное выступление на заданную тему. В учебных заведениях время доклада, как правило, составляет 7-10 минут.

Цели доклада

1. Научиться убедительно и кратко излагать свои мысли в устной форме (эффективно продавать свой интеллектуальный продукт).
2. Донести информацию до слушателя, установить контакт с аудиторией и получить обратную связь.

План и содержание доклада

Важно при подготовке доклада учитывать три его фазы: мотивацию, убеждение, побуждение.

Фаза 1: Мотивация

- Риторические вопросы.
- Актуальные события.
- Цитаты.
- Неожиданное для слушателей начало доклада.

Главная цель фазы открытия (мотивации) – привлечь внимание слушателей к докладчику, поэтому длительность её минимальна.

Фаза 2: Убеждение Ядром хорошего доклада является информация. Она должна быть новой и понятной. Важно в процессе доклада не только сообщить информацию, но и убедить слушателей в правильности своей точки зрения. Для убеждения следует использовать:

- Сообщение о себе: кто?
- Обоснование необходимости доклада: почему?
- Доказательство: кто? когда? где? сколько?
- Пример: берем пример с…
- Сравнение: это так же, как…

- Проблемы: что мешает?

Фаза 3: Побуждение Третья фаза доклада должна способствовать положительной реакции слушателей. В заключении могут быть использованы:

- Обобщение.
- Прогноз.
- Цитата.
- Пожелания.
- Объявление о продолжении дискуссии.
- Просьба о предложениях по улучшению.
- Благодарность за внимание.

Обратная связь

При общении следует помнить о правильной реакции (реплике) на задаваемые вам вопросы.

Правильная реакция на вопрос:

- Да.
- Хорошо.
- Спасибо, что вы мне сказали.
- Это является совсем новой точкой зрения.
- Это можно реализовать.
- Именно это я имею в виду.
- Прекрасная идея.
- Это можно делать и так.
- Вы правы.
- Спасибо за ваши комментарии.
- Именно это и является основным вопросом проблемы.

Составляющие воздействия докладчика на слушателей

Составляющие воздействия	**Средства достижения воздействия**
Язык доклада	Короткие предложения. Выделение главных предложений. Выбор слов. Образность языка.
Голос	Выразительность. Вариации громкости. Темп речи.
Внешнее общение	Зрительный контакт. Обратная связь. Доверительность. Жестикуляция.

Для занятий, проводимым в дистанционном формате (видеоконференции)

1. Примерный перечень учебно-методических материалов по занятиям, проводимым в дистанционном формате (видеоконференции)

Наименование	Длительность	Формат	Примечание
Конспект лекции (обязательно)	До 10 – 12 стр	.pdf где Формат названия файла: «Лекция №__»	На каждое занятие лекционного типа. Не более 10 Мб
Презентация по каждой лекции (обязательно)		.ppt, .pptx где Формат названия файла: «Презентация к лекции №__»	Шаблон по Приказу 65 от 06.04.23 Не более 10 Мб
Учебник/учебное пособие/монография/любые публикации		.pdf	Не более 10 Мб
Ссылки на законодательные акты			Необходимо создать ссылку на ресурс
Задание к семинарскому (практическому) занятию (обязательно)		.pdf Формат названия файла: «Задание к семинару №__»	На каждое семинарское (практическое) занятие. Не более 10 Мб
Задание для контрольной работы (обязательно)		.pdf Формат названия файла: «Контрольная работа»	Не более 10 Мб

2. Примерный план семинарских занятий, проводимых в дистанционном формате (видеоконференции)

Этап	Длительность	Содержание	Примечание
Вступление	5-10 минут	Преподаватель формулирует тему, цели занятия (компетенции), сообщает критерии оценивания	
Блиц опрос	5-10 минут	Краткий опрос по основному материалу пройденных тем, основным терминам	Рекомендательный характер
Основной этап	40-65 минут	Опрос по вопросам, заданным к семинарскому занятия Проверка выполнения практических заданий, обсуждение решений Заслушивание и обсуждение докладов	
Заключительный этап	5-10 минут	Подведением итогов занятия, выводы	

ПРИМЕРНЫЕ ЗАДАНИЯ К СЕМИНАРСКИМ (ПРАКТИЧЕСКИМ) ЗАНЯТИЯМ

Тема 1. Введение в системное программирование в среде Linux. Инструменты разработки для системного программирования
Перечень компетенций (части компетенции), проверяемых оценочными средствами на семинаре:

№ п/п	Код компетенции	Наименование
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием
2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

Примерные вопросы к семинарскому (практическому) занятию:

1. Что такое системное программирование и каковы его основные цели и задачи в контексте операционной системы Linux?
2. Какие инструменты разработки используются для системного программирования в Linux? Опишите их основные функции.
3. Каковы ключевые особенности и преимущества операционной системы Linux для системного программирования?
4. Как использование инструментов таких как GCC, GDB и Makefile способствует эффективной разработке и отладке программ?
5. Приведите примеры реальных проектов, в которых системное программирование в Linux сыграло ключевую роль.

Пример практического задания:
Задание: Студенты должны разработать простой модуль для мониторинга системных ресурсов в Linux, используя инструменты GCC и GDB. Модуль должен собирать данные о загрузке процессора и использовании памяти и выводить их в удобном формате.
Вопросы к задаче:

1. Какие шаги необходимо выполнить для разработки и компиляции модуля с использованием GCC?
2. Как можно использовать GDB для отладки модуля и выявления ошибок?
3. Какие ключевые функции следует реализовать в модуле для сбора и отображения данных о системных ресурсах?
4. Объясните, как правильно организовать структуру проекта и написать Makefile для автоматизации сборки.
5. Как можно расширить функциональность модуля для мониторинга других системных параметров?

СПИСОК ЛИТЕРАТУРЫ, НОРМАТИВНЫХ ПРАВОВЫХ АКТОВ, АКТЫ СУДЕБНОЙ ПРАКТИКИ, ССЫЛКИ НА ИНТЕРНЕТ-РЕСУРСЫ:

Основная литература

1. Гунько А.В. Системное программирование в среде Linux : учебное пособие / Гунько А.В.. — Новосибирск : Новосибирский государственный технический университет, 2020. — 235 с. — ISBN 978-5-7782-4160-2. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/98735.html (дата обращения: 25.06.2024). — Режим доступа: для авторизир. пользователей
2. Токманцев, Т. Б. Алгоритмические языки и программирование : учебное пособие для СПО / Т. Б. Токманцев ; под редакцией В. Б. Костоусова. — 3-е изд. — Саратов,

Екатеринбург : Профобразование, Уральский федеральный университет, 2024. — 102 с. — ISBN 978-5-4488-0510-3, 978-5-7996-2899-4. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139513.html (дата обращения: 23.05.2024). — Режим доступа: для авторизир. Пользователей

Дополнительная литература

1. Губарь, Ю. В. Введение в математическое программирование : учебное пособие для СПО / Ю. В. Губарь. — 2-е изд. — Саратов : Профобразование, 2024. — 225 с. — ISBN 978-5-4488-0992-7. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139745.html (дата обращения: 27.05.2024). — Режим доступа: для авторизир. пользователей
2. Моренкова, О. И. Операционные системы. Linux : учебное пособие для СПО / О. И. Моренкова, А. Ю. Голошубов. — Саратов : Профобразование, 2024. — 108 с. — ISBN 978-5-4488-1864-6. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139041.html (дата обращения: 07.05.2024). — Режим доступа: для авторизир. пользователей

БАЛЛЫ ЗА СЕМИНАРСКОЕ (ПРАКТИЧЕСКОЕ) ЗАНЯТИЕ:

Всего баллов за семинар, в том числе:	**До 5**
Вопросы к семинарскому (практическому) занятию	от 0 до 2
Практические задания	от 0 до 3

Тема 2. Введение в системные вызовы. Работа с файловой системой

Перечень компетенций (части компетенции), проверяемых оценочными средствами на семинаре:

№ п/п	**Код компетенции**	**Наименование**
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием
2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

Примерные вопросы к семинарскому (практическому) занятию:

1. Что такое системные вызовы и какую роль они играют в операционной системе Linux?
2. Опишите основные системные вызовы для работы с файлами и директориями. Приведите примеры их использования.
3. Как осуществляется обработка ошибок при использовании системных вызовов?
4. Какие права доступа к файлам и директориям существуют в Linux и как они управляются?
5. Каковы особенности работы с символическими и жесткими ссылками в файловой системе Linux?

Пример практического задания:

Задание: Разработайте программу на языке C, которая создает новый файл, записывает в него данные, а затем читает и выводит содержимое файла. В программе должны быть реализованы проверки ошибок.

Вопросы к задаче:

1. Какие системные вызовы необходимо использовать для создания, записи и чтения файлов?

2. Как можно обработать и логировать ошибки, возникающие при выполнении системных вызовов?
3. Какие дополнительные функции можно реализовать для управления файлами и директориями (например, изменение прав доступа)?
4. Объясните, как программа должна обрабатывать различные исключительные ситуации (например, попытка открытия несуществующего файла).
5. Какие методы отладки можно использовать для проверки корректности работы программы?

СПИСОК ЛИТЕРАТУРЫ, НОРМАТИВНЫХ ПРАВОВЫХ АКТОВ, АКТЫ СУДЕБНОЙ ПРАКТИКИ, ССЫЛКИ НА ИНТЕРНЕТ-РЕСУРСЫ:

Основная литература

1. Гунько А.В. Системное программирование в среде Linux : учебное пособие / Гунько А.В.. — Новосибирск : Новосибирский государственный технический университет, 2020. — 235 с. — ISBN 978-5-7782-4160-2. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/98735.html (дата обращения: 25.06.2024). — Режим доступа: для авторизир. пользователей
2. Токманцев, Т. Б. Алгоритмические языки и программирование : учебное пособие для СПО / Т. Б. Токманцев ; под редакцией В. Б. Костоусова. — 3-е изд. — Саратов, Екатеринбург : Профобразование, Уральский федеральный университет, 2024. — 102 с. — ISBN 978-5-4488-0510-3, 978-5-7996-2899-4. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139513.html (дата обращения: 23.05.2024). — Режим доступа: для авторизир. Пользователей

Дополнительная литература

1. Губарь, Ю. В. Введение в математическое программирование : учебное пособие для СПО / Ю. В. Губарь. — 2-е изд. — Саратов : Профобразование, 2024. — 225 с. — ISBN 978-5-4488-0992-7. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139745.html (дата обращения: 27.05.2024). — Режим доступа: для авторизир. пользователей
2. Моренкова, О. И. Операционные системы. Linux : учебное пособие для СПО / О. И. Моренкова, А. Ю. Голошубов. — Саратов : Профобразование, 2024. — 108 с. — ISBN 978-5-4488-1864-6. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139041.html (дата обращения: 07.05.2024). — Режим доступа: для авторизир. пользователей

БАЛЛЫ ЗА СЕМИНАРСКОЕ (ПРАКТИЧЕСКОЕ) ЗАНЯТИЕ:

Всего баллов за семинар, в том числе:	**До 5**
Вопросы к семинарскому (практическому) занятию	от 0 до 2
Практические задания	от 0 до 3

Тема 3. Управление процессами. Многопоточность в Linux
Перечень компетенций (части компетенции), проверяемых оценочными средствами на семинаре:

№ п/п	**Код компетенции**	**Наименование**
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием

2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

Примерные вопросы к семинарскому (практическому) занятию:

1. Как осуществляется создание и управление процессами в Linux с использованием системных вызовов?
2. Какие системные вызовы используются для создания и управления потоками в Linux?
3. Как реализуется межпроцессное взаимодействие (IPC) в Linux? Приведите примеры.
4. Какие механизмы синхронизации потоков существуют в Linux и как их правильно использовать?
5. Какие проблемы могут возникнуть при многопоточном программировании и как их можно предотвратить?

Пример практического задания:

Задание: Разработайте многопоточную программу на языке C, которая создает несколько потоков, каждый из которых выполняет вычисления и выводит результат. Реализуйте синхронизацию потоков для предотвращения гонок.

Вопросы к задаче:

1. Какие функции библиотеки pthread используются для создания и управления потоками?
2. Как можно реализовать синхронизацию потоков с использованием mutex и condition variables?
3. Какие проблемы могут возникнуть при некорректной синхронизации потоков и как их избежать?
4. Как можно отладить многопоточную программу и выявить ошибки синхронизации?
5. Какие дополнительные функции можно реализовать для улучшения производительности многопоточной программы?

СПИСОК ЛИТЕРАТУРЫ, НОРМАТИВНЫХ ПРАВОВЫХ АКТОВ, АКТЫ СУДЕБНОЙ ПРАКТИКИ, ССЫЛКИ НА ИНТЕРНЕТ-РЕСУРСЫ:

Основная литература

1. Гунько А.В. Системное программирование в среде Linux : учебное пособие / Гунько А.В.. — Новосибирск : Новосибирский государственный технический университет, 2020. — 235 c. — ISBN 978-5-7782-4160-2. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/98735.html (дата обращения: 25.06.2024). — Режим доступа: для авторизир. пользователей
2. Токманцев, Т. Б. Алгоритмические языки и программирование : учебное пособие для СПО / Т. Б. Токманцев ; под редакцией В. Б. Костоусова. — 3-е изд. — Саратов, Екатеринбург : Профобразование, Уральский федеральный университет, 2024. — 102 c. — ISBN 978-5-4488-0510-3, 978-5-7996-2899-4. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139513.html (дата обращения: 23.05.2024). — Режим доступа: для авторизир. Пользователей

Дополнительная литература

1. Губарь, Ю. В. Введение в математическое программирование : учебное пособие для СПО / Ю. В. Губарь. — 2-е изд. — Саратов : Профобразование, 2024. — 225 c. — ISBN 978-5-4488-0992-7. — Текст : электронный // Цифровой образовательный

ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139745.html (дата обращения: 27.05.2024). — Режим доступа: для авторизир. пользователей

2. Моренкова, О. И. Операционные системы. Linux : учебное пособие для СПО / О. И. Моренкова, А. Ю. Голошубов. — Саратов : Профобразование, 2024. — 108 с. — ISBN 978-5-4488-1864-6. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139041.html (дата обращения: 07.05.2024). — Режим доступа: для авторизир. пользователей

БАЛЛЫ ЗА СЕМИНАРСКОЕ (ПРАКТИЧЕСКОЕ) ЗАНЯТИЕ:

Всего баллов за семинар, в том числе:	**До 5**
Вопросы к семинарскому (практическому) занятию	от 0 до 2
Практические задания	от 0 до 3

Тема 4. Ввод/вывод и буферизация. Сетевое программирование в Linux

Перечень компетенций (части компетенции), проверяемых оценочными средствами на семинаре:

№ п/п	Код компетенции	Наименование
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием
2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

Примерные вопросы к семинарскому (практическому) занятию:

1. Какие системные вызовы используются для выполнения операций ввода/вывода в Linux?
2. Как осуществляется буферизация ввода/вывода и почему она важна?
3. Как реализовать асинхронный ввод/вывод в Linux?
4. Какие функции используются для создания и управления сетевыми сокетами?
5. Приведите примеры клиент-серверных приложений, реализованных с использованием сетевых сокетов.

Пример практического задания:

Задание: Напишите программу на языке C, которая реализует простой сетевой сервер, принимающий подключения от клиентов и обрабатывающий запросы в многопоточном режиме.

Вопросы к задаче:

1. Какие системные вызовы используются для создания серверного сокета и приема соединений?
2. Как можно реализовать обработку запросов клиентов в многопоточном режиме?
3. Какие методы буферизации и асинхронного ввода/вывода можно использовать для повышения производительности сервера?
4. Как можно отладить сетевую программу и обеспечить ее корректную работу при высоких нагрузках?
5. Какие меры безопасности необходимо предусмотреть при разработке сетевых приложений?

СПИСОК ЛИТЕРАТУРЫ, НОРМАТИВНЫХ ПРАВОВЫХ АКТОВ, АКТЫ СУДЕБНОЙ ПРАКТИКИ, ССЫЛКИ НА ИНТЕРНЕТ-РЕСУРСЫ:

Основная литература

1. Гунько А.В. Системное программирование в среде Linux : учебное пособие / Гунько А.В.. — Новосибирск : Новосибирский государственный технический университет, 2020. — 235 с. — ISBN 978-5-7782-4160-2. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/98735.html (дата обращения: 25.06.2024). — Режим доступа: для авторизир. пользователей
2. Токманцев, Т. Б. Алгоритмические языки и программирование : учебное пособие для СПО / Т. Б. Токманцев ; под редакцией В. Б. Костоусова. — 3-е изд. — Саратов, Екатеринбург : Профобразование, Уральский федеральный университет, 2024. — 102 с. — ISBN 978-5-4488-0510-3, 978-5-7996-2899-4. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139513.html (дата обращения: 23.05.2024). — Режим доступа: для авторизир. Пользователей

Дополнительная литература

1. Губарь, Ю. В. Введение в математическое программирование : учебное пособие для СПО / Ю. В. Губарь. — 2-е изд. — Саратов : Профобразование, 2024. — 225 с. — ISBN 978-5-4488-0992-7. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139745.html (дата обращения: 27.05.2024). — Режим доступа: для авторизир. пользователей
2. Моренкова, О. И. Операционные системы. Linux : учебное пособие для СПО / О. И. Моренкова, А. Ю. Голошубов. — Саратов : Профобразование, 2024. — 108 с. — ISBN 978-5-4488-1864-6. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139041.html (дата обращения: 07.05.2024). — Режим доступа: для авторизир. пользователей

БАЛЛЫ ЗА СЕМИНАРСКОЕ (ПРАКТИЧЕСКОЕ) ЗАНЯТИЕ:

Всего баллов за семинар, в том числе:	**До 5**
Вопросы к семинарскому (практическому) занятию	от 0 до 2
Практические задания	от 0 до 3

Тема 5. Модульное программирование и динамические библиотеки. Разработка и отладка модулей ядра

Перечень компетенций (части компетенции), проверяемых оценочными средствами на семинаре:

№ п/п	Код компетенции	Наименование
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием
2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

Примерные вопросы к семинарскому (практическому) занятию:

1. Каковы основные принципы модульного программирования в Linux?

2. Как создать и использовать статические и динамические библиотеки в Linux?
3. Какие инструменты используются для разработки и отладки модулей ядра?
4. Каковы основные этапы разработки и компиляции модулей ядра?
5. Как осуществлять отладку и тестирование модулей ядра в Linux?

Пример практического задания:

Задание: Разработайте и отладьте модуль ядра, который добавляет новую системную функцию в Linux. Модуль должен быть загружаемым и выгружаемым.

Вопросы к задаче:

1. Какие шаги необходимо выполнить для создания и компиляции модуля ядра?
2. Как можно загрузить и выгрузить модуль ядра с помощью команд insmod и rmmod?
3. Какие инструменты можно использовать для отладки модуля ядра и выявления ошибок?
4. Как можно реализовать и протестировать новую системную функцию в модуле ядра?
5. Какие меры предосторожности необходимо учитывать при разработке и тестировании модулей ядра?

СПИСОК ЛИТЕРАТУРЫ, НОРМАТИВНЫХ ПРАВОВЫХ АКТОВ, АКТЫ СУДЕБНОЙ ПРАКТИКИ, ССЫЛКИ НА ИНТЕРНЕТ-РЕСУРСЫ:

Основная литература

1. Гунько А.В. Системное программирование в среде Linux : учебное пособие / Гунько А.В.. — Новосибирск : Новосибирский государственный технический университет, 2020. — 235 c. — ISBN 978-5-7782-4160-2. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/98735.html (дата обращения: 25.06.2024). — Режим доступа: для авторизир. пользователей
2. Токманцев, Т. Б. Алгоритмические языки и программирование : учебное пособие для СПО / Т. Б. Токманцев ; под редакцией В. Б. Костоусова. — 3-е изд. — Саратов, Екатеринбург : Профобразование, Уральский федеральный университет, 2024. — 102 c. — ISBN 978-5-4488-0510-3, 978-5-7996-2899-4. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139513.html (дата обращения: 23.05.2024). — Режим доступа: для авторизир. Пользователей

Дополнительная литература

1. Губарь, Ю. В. Введение в математическое программирование : учебное пособие для СПО / Ю. В. Губарь. — 2-е изд. — Саратов : Профобразование, 2024. — 225 c. — ISBN 978-5-4488-0992-7. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139745.html (дата обращения: 27.05.2024). — Режим доступа: для авторизир. пользователей
2. Моренкова, О. И. Операционные системы. Linux : учебное пособие для СПО / О. И. Моренкова, А. Ю. Голошубов. — Саратов : Профобразование, 2024. — 108 c. — ISBN 978-5-4488-1864-6. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139041.html (дата обращения: 07.05.2024). — Режим доступа: для авторизир. пользователей

БАЛЛЫ ЗА СЕМИНАРСКОЕ (ПРАКТИЧЕСКОЕ) ЗАНЯТИЕ:

Всего баллов за семинар, в том числе:	**До 5**
Вопросы к семинарскому (практическому) занятию	от 0 до 2
Практические задания	от 0 до 3

Тема 6. Управление памятью. Безопасность системного программного обеспечения

Перечень компетенций (части компетенции), проверяемых оценочными средствами на семинаре:

№ п/п	Код компетенции	Наименование
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием
2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

Примерные вопросы к семинарскому (практическому) занятию:

1. Какие механизмы управления памятью существуют в Linux?
2. Как использовать системные вызовы malloc, free и mmap для управления памятью?
3. Какие методы обеспечения безопасности программного обеспечения используются в Linux?
4. Как предотвратить переполнение буфера и другие уязвимости в программном коде?
5. Какие инструменты и методы можно использовать для анализа безопасности кода?

Пример практического задания:

Задание: Напишите программу на языке C, которая выделяет динамическую память для массива данных, заполняет его и выводит содержимое. Реализуйте проверку переполнения буфера.

Вопросы к задаче:

1. Как использовать функции malloc и free для динамического выделения и освобождения памяти?
2. Какие методы можно использовать для предотвращения переполнения буфера?
3. Как отладить программу с динамическим выделением памяти и выявить ошибки управления памятью?
4. Какие дополнительные меры безопасности можно внедрить в программу?
5. Как можно использовать инструменты статического и динамического анализа для проверки безопасности кода?

СПИСОК ЛИТЕРАТУРЫ, НОРМАТИВНЫХ ПРАВОВЫХ АКТОВ, АКТЫ СУДЕБНОЙ ПРАКТИКИ, ССЫЛКИ НА ИНТЕРНЕТ-РЕСУРСЫ:

Основная литература

1. Гунько А.В. Системное программирование в среде Linux : учебное пособие / Гунько А.В.. — Новосибирск : Новосибирский государственный технический университет,

2020. — 235 с. — ISBN 978-5-7782-4160-2. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/98735.html (дата обращения: 25.06.2024). — Режим доступа: для авторизир. пользователей

2. Токманцев, Т. Б. Алгоритмические языки и программирование : учебное пособие для СПО / Т. Б. Токманцев ; под редакцией В. Б. Костоусова. — 3-е изд. — Саратов, Екатеринбург : Профобразование, Уральский федеральный университет, 2024. — 102 с. — ISBN 978-5-4488-0510-3, 978-5-7996-2899-4. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139513.html (дата обращения: 23.05.2024). — Режим доступа: для авторизир. Пользователей

Дополнительная литература

1. Губарь, Ю. В. Введение в математическое программирование : учебное пособие для СПО / Ю. В. Губарь. — 2-е изд. — Саратов : Профобразование, 2024. — 225 с. — ISBN 978-5-4488-0992-7. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139745.html (дата обращения: 27.05.2024). — Режим доступа: для авторизир. пользователей
2. Моренкова, О. И. Операционные системы. Linux : учебное пособие для СПО / О. И. Моренкова, А. Ю. Голошубов. — Саратов : Профобразование, 2024. — 108 с. — ISBN 978-5-4488-1864-6. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139041.html (дата обращения: 07.05.2024). — Режим доступа: для авторизир. пользователей

БАЛЛЫ ЗА СЕМИНАРСКОЕ (ПРАКТИЧЕСКОЕ) ЗАНЯТИЕ:

Всего баллов за семинар, в том числе:	**До 5**
Вопросы к семинарскому (практическому) занятию	от 0 до 2
Практические задания	от 0 до 3

Тема 7. Средства мониторинга и профилирования. Автоматизация и скриптовые языки

Перечень компетенций (части компетенции), проверяемых оценочными средствами на семинаре:

№ п/п	Код компетенции	Наименование
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием
2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

Примерные вопросы к семинарскому (практическому) занятию:

1. Какие инструменты используются для мониторинга и профилирования системных ресурсов в Linux?
2. Как использовать профилировщики, такие как gprof и perf, для анализа производительности программ?
3. Какие возможности предоставляют скриптовые языки (Bash, Python) для автоматизации задач системного программирования?
4. Как написать и отладить скрипты для автоматизации повседневных задач администратора системы?
5. Приведите примеры автоматизации задач с использованием скриптовых языков.

Пример практического задания:

Задание: Напишите скрипт на Bash, который автоматически собирает информацию о системных ресурсах (загрузка процессора, использование памяти) и сохраняет её в лог-файл. Реализуйте автоматическое удаление старых лог-файлов.

Вопросы к задаче:

1. Какие команды и утилиты Bash можно использовать для сбора информации о системных ресурсах?
2. Как можно реализовать управление лог-файлами с помощью скриптов?
3. Какие методы можно использовать для отладки и тестирования скриптов?
4. Как можно расширить функциональность скрипта для выполнения дополнительных задач (например, отправка уведомлений)?
5. Какие меры безопасности необходимо учитывать при написании скриптов?

СПИСОК ЛИТЕРАТУРЫ, НОРМАТИВНЫХ ПРАВОВЫХ АКТОВ, АКТЫ СУДЕБНОЙ ПРАКТИКИ, ССЫЛКИ НА ИНТЕРНЕТ-РЕСУРСЫ:

Основная литература

1. Гунько А.В. Системное программирование в среде Linux : учебное пособие / Гунько А.В.. — Новосибирск : Новосибирский государственный технический университет, 2020. — 235 c. — ISBN 978-5-7782-4160-2. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/98735.html (дата обращения: 25.06.2024). — Режим доступа: для авторизир. пользователей
2. Токманцев, Т. Б. Алгоритмические языки и программирование : учебное пособие для СПО / Т. Б. Токманцев ; под редакцией В. Б. Костоусова. — 3-е изд. — Саратов, Екатеринбург : Профобразование, Уральский федеральный университет, 2024. — 102 c. — ISBN 978-5-4488-0510-3, 978-5-7996-2899-4. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139513.html (дата обращения: 23.05.2024). — Режим доступа: для авторизир. Пользователей

Дополнительная литература

1. Губарь, Ю. В. Введение в математическое программирование : учебное пособие для СПО / Ю. В. Губарь. — 2-е изд. — Саратов : Профобразование, 2024. — 225 c. — ISBN 978-5-4488-0992-7. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139745.html (дата обращения: 27.05.2024). — Режим доступа: для авторизир. пользователей
2. Моренкова, О. И. Операционные системы. Linux : учебное пособие для СПО / О. И. Моренкова, А. Ю. Голошубов. — Саратов : Профобразование, 2024. — 108 c. — ISBN 978-5-4488-1864-6. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139041.html (дата обращения: 07.05.2024). — Режим доступа: для авторизир. Пользователей

БАЛЛЫ ЗА СЕМИНАРСКОЕ (ПРАКТИЧЕСКОЕ) ЗАНЯТИЕ:

Всего баллов за семинар, в том числе:	**До 5**
Вопросы к семинарскому (практическому) занятию	от 0 до 2
Практические задания	от 0 до 3

Тема 8. Практические проекты и кейсы. Перспективы и направления развития системного программирования в Linux
Перечень компетенций (части компетенции), проверяемых оценочными средствами на семинаре:

№ п/п	Код компетенции	Наименование
1	ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием
2	ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств

Примерные вопросы к семинарскому (практическому) занятию:

1. Какие актуальные проблемы решаются с помощью системного программирования в Linux?
2. Приведите примеры успешных проектов, реализованных с использованием системного программирования в Linux.
3. Какие направления развития системного программирования в Linux являются наиболее перспективными?
4. Как современные технологии (контейнеризация, виртуализация) влияют на системное программирование в Linux?
5. Какие вызовы и возможности открываются перед разработчиками системного ПО в Linux?

Пример практического задания:
Задание: Студенты должны разработать мини-проект, включающий создание и отладку модуля ядра или системной утилиты. Проект должен быть основан на реальном кейсе и включать все этапы разработки: анализ требований, разработка, тестирование, отладка и документация.
Вопросы к задаче:

1. Какие этапы необходимо пройти при разработке мини-проекта на основе системного программирования в Linux?
2. Как правильно организовать работу команды для эффективного выполнения проекта?
3. Какие инструменты и методы использовать для отладки и тестирования разрабатываемого программного обеспечения?
4. Как документировать процесс разработки и результаты проекта?
5. Какие дополнительные функции можно внедрить для улучшения функциональности разрабатываемого ПО?

СПИСОК ЛИТЕРАТУРЫ, НОРМАТИВНЫХ ПРАВОВЫХ АКТОВ, АКТЫ СУДЕБНОЙ ПРАКТИКИ, ССЫЛКИ НА ИНТЕРНЕТ-РЕСУРСЫ:

Основная литература

1. Гунько А.В. Системное программирование в среде Linux : учебное пособие / Гунько А.В.. — Новосибирск : Новосибирский государственный технический университет, 2020. — 235 с. — ISBN 978-5-7782-4160-2. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/98735.html (дата обращения: 25.06.2024). — Режим доступа: для авторизир. пользователей

2. Токманцев, Т. Б. Алгоритмические языки и программирование : учебное пособие для СПО / Т. Б. Токманцев ; под редакцией В. Б. Костоусова. — 3-е изд. — Саратов, Екатеринбург : Профобразование, Уральский федеральный университет, 2024. — 102 с. — ISBN 978-5-4488-0510-3, 978-5-7996-2899-4. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139513.html (дата обращения: 23.05.2024). — Режим доступа: для авторизир. Пользователей

Дополнительная литература

1. Губарь, Ю. В. Введение в математическое программирование : учебное пособие для СПО / Ю. В. Губарь. — 2-е изд. — Саратов : Профобразование, 2024. — 225 с. — ISBN 978-5-4488-0992-7. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139745.html (дата обращения: 27.05.2024). — Режим доступа: для авторизир. пользователей
2. Моренкова, О. И. Операционные системы. Linux : учебное пособие для СПО / О. И. Моренкова, А. Ю. Голошубов. — Саратов : Профобразование, 2024. — 108 с. — ISBN 978-5-4488-1864-6. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/139041.html (дата обращения: 07.05.2024). — Режим доступа: для авторизир. Пользователей

БАЛЛЫ ЗА СЕМИНАРСКОЕ (ПРАКТИЧЕСКОЕ) ЗАНЯТИЕ:

Всего баллов за семинар, в том числе:	**До 5**
Вопросы к семинарскому (практическому) занятию	от 0 до 2
Практические задания	от 0 до 3

Тестовые задания

Содержание банка тестовых заданий

V1: Системное программирование в среде Linux

ПК 1.2: Разрабатывать программные модули в соответствии с техническим заданием

Пример теста

I:
S: Какой системный вызов используется для создания нового процесса в Linux?
+: fork()
-: create_process()
-: new_process()
-: process_start()

I:
S: Какой компилятор используется для компиляции программ на языке C/C++ в Linux?
+: GCC
-: GDB
-: Makefile
-: CLang

I:
S: Какой системный вызов используется для открытия файла в Linux?
+: open()
-: fopen()
-: file_open()
-: file()

I:
S: Какую функцию используют для динамического выделения памяти в языке C?
+: malloc()
-: new()
-: alloc()
-: calloc()

ПК 1.3: Выполнять отладку программных модулей с использованием специализированных программных средств

Пример теста

I:
S: Какой инструмент используется для отладки программ в Linux?
+: GDB
-: GCC
-: Makefile
-: Valgrind

I:
S: Какой ключ GDB используется для установки точки останова (breakpoint)?
+: break

-: stop
-: hold
-: pause

I:
S: Какой инструмент помогает выявлять утечки памяти в приложениях на языке C/C++?
+: Valgrind
-: GDB
-: GCC
-: Makefile

I:
S: Какой флаг используется с компилятором GCC для включения отладочной информации в исполняемый файл?
+: -g
-: -o
-: -O2
-: -Wall

Примеры дополнительных тестов
ПК 1.2

I:
S: Какой системный вызов используется для чтения данных из файла в Linux?
+: read()
-: fread()
-: get()
-: retrieve()

I:
S: Какой системный вызов используется для записи данных в файл в Linux?
+: write()
-: fwrite()
-: put()
-: send()

I:
S: Какой системный вызов используется для закрытия файла в Linux?
+: close()
-: fclose()
-: end()
-: terminate()

ПК 1.3

I:
S: Какой ключ GDB используется для продолжения выполнения программы после остановки?
+: continue
-: go
-: run

-: resume

I:

S: Какой флаг компиляции используется в GCC для включения всех предупреждений?

+: -Wall

-: -Werror

-: -Wextra

-: -Wpedantic

I:

S: Какой инструмент используется для профилирования производительности программ в Linux?

+: perf

-: top

-: htop

-: gprof

<u>Форма заданий для зачета в дистанционном формате</u>

09.03.01 Информатика и вычислительная техника
V1: Системное программирование в среде Linux.
V2: Задание 1 (Далее указываются все варианты вопросов, которые идут в перечне вопросов к зачету, и они будут включены в экзаменационные билеты).
I:1.1
S: Как анализировать и оптимизировать использование процессора и памяти?.
I:1.2
S: Как осуществляется работа с межпроцессным взаимодействием (IPC) с использованием pipe и FIFO?

Ответы на билеты представляются в виде эссе, со свободным вводом текста.

Количество заданий в билете для дифференцированного зачета в рамках промежуточной аттестации – 2.

Форма тестового задания для зачета и дифференцированного зачета в дистанционном формате

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПРАВОСУДИЯ»

F1: Дисциплина: «Системное программирование в среде Linux»
F2: Кафедра информационного права, информатики и математики
V1: Системное программирование в среде Linux
ПК 1.2: Разрабатывать программные модули в соответствии с техническим заданием
Пример теста

I:
S: Какой системный вызов используется для создания нового процесса в Linux?
+: fork()
-: create_process()
-: new_process()
-: process_start()

I:
S: Какой компилятор используется для компиляции программ на языке C/C++ в Linux?
+: GCC
-: GDB
-: Makefile
-: CLang

I:
S: Какой системный вызов используется для открытия файла в Linux?
+: open()
-: fopen()
-: file_open()
-: file()

I:
S: Какую функцию используют для динамического выделения памяти в языке C?
+: malloc()
-: new()
-: alloc()
-: calloc()

ПК 1.3: Выполнять отладку программных модулей с использованием специализированных программных средств
Пример теста

I:
S: Какой инструмент используется для отладки программ в Linux?
+: GDB
-: GCC
-: Makefile
-: Valgrind

I:
S: Какой ключ GDB используется для установки точки останова (breakpoint)?
+: break
-: stop
-: hold
-: pause

I:
S: Какой инструмент помогает выявлять утечки памяти в приложениях на языке C/C++?
+: Valgrind
-: GDB
-: GCC
-: Makefile

I:
S: Какой флаг используется с компилятором GCC для включения отладочной информации в исполняемый файл?
+: -g
-: -o
-: -O2
-: -Wall

Примеры дополнительных тестов
ПК 1.2

I:
S: Какой системный вызов используется для чтения данных из файла в Linux?
+: read()
-: fread()
-: get()
-: retrieve()

I:
S: Какой системный вызов используется для записи данных в файл в Linux?
+: write()
-: fwrite()
-: put()
-: send()

I:
S: Какой системный вызов используется для закрытия файла в Linux?
+: close()
-: fclose()
-: end()
-: terminate()

ПК 1.3

I:
S: Какой ключ GDB используется для продолжения выполнения программы после остановки?
+: continue

-: go
-: run
-: resume

I:
S: Какой флаг компиляции используется в GCC для включения всех предупреждений?
+: -Wall
-: -Werror
-: -Wextra
-: -Wpedantic

I:
S: Какой инструмент используется для профилирования производительности программ в Linux?
+: perf
-: top
-: htop
-: gprof

Форма разметки экзаменационных билетов в дистанционном формате

V1: Системное программирование в среде Linux

I:1 S: Билет № 1

1. Что такое системное программирование и какие задачи оно решает?
2. Какие основные инструменты используются для разработки системного программного обеспечения в Linux?

I:2 S: Билет № 2

1. Какие системные вызовы используются для работы с файлами в Linux?
2. Как осуществляется управление правами доступа к файлам и директориям в Linux?

I:3 S: Билет № 3

1. Как используется компилятор GCC для компиляции программ на языке C/C++?
2. Какие возможности предоставляет отладчик GDB?

I:4 S: Билет № 4

1. Что такое многопоточность и какие преимущества она предоставляет?
2. Какие функции библиотеки pthread используются для создания и управления потоками?

I:5 S: Билет № 5

1. Какие системные вызовы используются для создания и управления процессами в Linux?
2. Как осуществляется межпроцессное взаимодействие (IPC) в Linux?

I:6 S: Билет № 6

1. Какие функции используются для создания и управления сетевыми сокетами?
2. Опишите процесс создания клиент-серверного приложения в Linux.

I:7 S: Билет № 7

1. Какие механизмы управления памятью существуют в Linux?
2. Как использовать системные вызовы malloc, free и mmap для управления памятью?

I:8 S: Билет № 8

1. Какие инструменты используются для мониторинга и профилирования системных ресурсов в Linux?
2. Как профилировать производительность программы с помощью gprof и perf?

I:9 S: Билет № 9

1. Что такое динамические библиотеки и как они используются в Linux?

2. Как создаются и подключаются статические и динамические библиотеки?

I:10 S: Билет № 10

1. Какие методы обеспечения безопасности программного обеспечения используются в Linux?
2. Как анализировать безопасность кода и выявлять уязвимости?

I:11 S: Билет № 11

1. Какие основные системные вызовы для работы с файловой системой вы знаете?
2. Каковы особенности работы с символическими и жесткими ссылками в файловой системе Linux?

I:12 S: Билет № 12

1. Какие существуют методы синхронизации потоков в Linux?
2. Какие проблемы могут возникнуть при некорректной синхронизации потоков?

I:13 S: Билет № 13

1. Какие системные вызовы используются для выполнения операций ввода/вывода в Linux?
2. Как осуществляется буферизация ввода/вывода и почему она важна?

I:14 S: Билет № 14

1. Какие функции и утилиты используются для мониторинга системных ресурсов в Linux?
2. Какие возможности предоставляют скриптовые языки (Bash, Python) для автоматизации задач?

I:15 S: Билет № 15

1. Как осуществляется управление устройствами и драйверами в Linux?
2. Какие права доступа и модели безопасности существуют для системных ресурсов?

I:16 S: Билет № 16

1. Какие инструменты используются для разработки и отладки модулей ядра в Linux?
2. Опишите процесс создания и компиляции модуля ядра.

I:17 S: Билет № 17

1. Как осуществить асинхронный ввод/вывод в Linux?
2. Какие системные вызовы используются для создания и управления потоками?

I:18 S: Билет № 18

1. Как использовать утилиту strace для отладки системных вызовов?
2. Какие методы можно использовать для предотвращения переполнения буфера?

I:19 S: Билет № 19

1. Какие инструменты используются для работы с исходным кодом в Linux?
2. Как Git помогает в управлении версиями кода?

I:20 S: Билет № 20

1. Какие меры безопасности необходимы при разработке сетевых приложений?
2. Какие инструменты используются для анализа производительности и отладки программ в Linux?

I:21 S: Билет № 21

1. Какие системные вызовы используются для работы с процессами и потоками?
2. Как осуществляется синхронизация данных между процессами и потоками?

I:22 S: Билет № 22

1. Какие функции используются для управления разделяемой памятью?
2. Как использовать инструмент Valgrind для выявления утечек памяти?

I:23 S: Билет № 23

1. Какие меры предосторожности необходимы при работе с динамическим выделением памяти?
2. Как осуществляется обработка сигналов в Linux?

I:24 S: Билет № 24

1. Какие функции используются для создания и управления очередями сообщений?
2. Как реализовать безопасность и защиту данных при системном программировании?

I:25 S: Билет № 25

1. Какие инструменты используются для тестирования и верификации системного программного обеспечения?
2. Как интегрировать различные системы и данные в рамках одного программного проекта в Linux?

I:26 S: Билет № 26

1. Какие современные технологии влияют на развитие системного программирования в Linux?
2. Как контейнеризация и виртуализация используются в системном программировании?

I:27 S: Билет № 27

1. Какие вызовы стоят перед разработчиками системного программного обеспечения в Linux?
2. Как осуществляется автоматизация сборки проектов с использованием Makefile?

I:28 S: Билет № 28

1. Какие основные функции и особенности операционной системы Linux делают её удобной для системного программирования?
2. Какие права доступа к файлам и директориям существуют в Linux и как они управляются?

I:29 S: Билет № 29

1. Как осуществляется профилирование и мониторинг многопоточных приложений?
2. Какие системные вызовы используются для работы с сетевыми соединениями?

I:30 S: Билет № 30

1. Как осуществляется автоматизация и скриптование задач с использованием Bash?
2. Какие инструменты можно использовать для тестирования и отладки скриптов?

I:31 S: Билет № 31

1. Как осуществляется управление буферами ввода/вывода?
2. Какие методы используются для анализа и оптимизации производительности программ?

I:32 S: Билет № 32

1. Как профилировать и отлаживать сетевые приложения в Linux?
2. Какие особенности следует учитывать при разработке многопоточных приложений?

I:33 S: Билет № 33

1. Какие системные вызовы используются для управления файловой системой?
2. Как осуществляется работа с символическими и жесткими ссылками?

I:34 S: Билет № 34

1. Какие механизмы и инструменты используются для обеспечения безопасности в системах Linux?
2. Как осуществить управление доступом и правами пользователей?

I:35 S: Билет № 35

1. Как осуществляется отладка программ с использованием gdb?
2. Какие методы используются для предотвращения утечек памяти?

I:36 S: Билет № 36

1. Как осуществляется межпроцессное взаимодействие с использованием очередей сообщений?
2. Какие функции используются для управления потоками в многопоточном программировании?

I:37 S: Билет № 37

1. Как используются системные журналы для мониторинга и отладки?
2. Какие функции и утилиты можно использовать для мониторинга производительности системы?

I:38 S: Билет № 38

1. Какие инструменты используются для анализа и профилирования сетевых подключений?
2. Как обеспечить безопасность сетевых приложений?

I:39 S: Билет № 39

1. Какие методы и технологии используются для разработки модулей ядра?
2. Как отладить и протестировать модуль ядра?

I:40 S: Билет № 40

1. Как организовать автоматизацию задач с использованием скриптов Python?
2. Какие функции и библиотеки можно использовать для управления системными ресурсами через скрипты?

I:41 S: Билет № 41

1. Как используется система контроля версий Git в процессе разработки?
2. Какие стратегии используются для управления ветвлением и слиянием в Git?

I:42 S: Билет № 42

1. Какие методы и инструменты используются для обеспечения отказоустойчивости системных приложений?
2. Как провести нагрузочное тестирование сетевых сервисов?

I:43 S: Билет № 43

1. Какие методы и инструменты используются для мониторинга системных ресурсов в реальном времени?
2. Как анализировать и оптимизировать использование процессора и памяти?

I:44 S: Билет № 44

1. Как осуществляется работа с межпроцессным взаимодействием (IPC) с использованием pipe и FIFO?
2. Какие методы используются для синхронизации процессов?

I:45 S: Билет № 45

1. Какие функции используются для управления виртуальной памятью?
2. Как отладить программы с использованием инструментов Valgrind и gdb?

I:46 S: Билет № 46

1. Какие системные вызовы используются для управления приоритетами процессов?
2. Как реализовать управление многозадачностью в приложениях Linux?

I:47 S: Билет № 47

1. Как осуществляется интеграция динамических библиотек в приложения?
2. Какие методы используются для управления конфигурацией приложений?

I:48 S: Билет № 48

1. Какие методы и инструменты используются для тестирования безопасности программного обеспечения?
2. Как анализировать и предотвращать уязвимости в коде?

I:49 S: Билет № 49

1. Как осуществляется управление входом/выходом в сетевых приложениях?
2. Какие функции используются для управления соединениями и их состояниями?

I:50 S: Билет № 50

1. Как используется strace для анализа работы системных вызовов?
2. Какие методы используются для отладки и анализа производительности системных приложений?

Федеральное государственное бюджетное образовательное учреждение высшего образования
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПРАВОСУДИЯ»

Вопросы, выносимые на зачет по дисциплине
Мировые информационные ресурсы
(наименование дисциплины)

1. Что такое системное программирование и какие задачи оно решает?
2. Какие особенности операционной системы Linux делают её удобной для системного программирования?
3. Какие основные инструменты используются для разработки системного программного обеспечения в Linux?
4. Как используется компилятор GCC для компиляции программ на языке C/C++?
5. Какие возможности предоставляет отладчик GDB?
6. Что такое Makefile и как он помогает автоматизировать сборку проектов?
7. Опишите процесс создания и управления процессами в Linux.
8. Какие системные вызовы используются для работы с файлами в Linux?
9. Как выполняется межпроцессное взаимодействие (IPC) в Linux?
10. Что такое системные вызовы и как они работают?
11. Какие основные системные вызовы для работы с файловой системой вы знаете?
12. Как осуществляется управление правами доступа к файлам и директориям в Linux?
13. Что такое символические и жесткие ссылки и как они используются?
14. Как осуществляется обработка ошибок при использовании системных вызовов?
15. Какие существуют методы синхронизации потоков в Linux?
16. Что такое многопоточность и какие преимущества она предоставляет?
17. Как используются библиотеки pthread для создания и управления потоками?
18. Какие проблемы могут возникнуть при некорректной синхронизации потоков?
19. Как реализуется асинхронный ввод/вывод в Linux?
20. Какие функции используются для создания и управления сетевыми сокетами?
21. Опишите процесс создания клиент-серверного приложения в Linux.
22. Какие основные системные вызовы используются для работы с сетевыми соединениями?
23. Что такое буферизация ввода/вывода и почему она важна?
24. Какие функции и утилиты используются для мониторинга системных ресурсов в Linux?
25. Как профилировать производительность программы с помощью gprof и perf?
26. Какие возможности предоставляют скриптовые языки (Bash, Python) для автоматизации задач?
27. Как написать и отладить скрипт на Bash для автоматизации системных задач?
28. Что такое динамические библиотеки и как они используются в Linux?
29. Как создаются и подключаются статические и динамические библиотеки?
30. Какие инструменты используются для разработки и отладки модулей ядра в Linux?
31. Опишите процесс создания и компиляции модуля ядра.
32. Какие меры предосторожности необходимо учитывать при разработке модулей ядра?
33. Как осуществляется управление памятью в Linux?
34. Какие системные вызовы используются для динамического выделения и освобождения памяти?
35. Что такое переполнение буфера и как его предотвратить?
36. Какие методы и инструменты используются для обеспечения безопасности программного обеспечения в Linux?

37. Как анализировать безопасность кода и выявлять уязвимости?
38. Какие современные технологии влияют на развитие системного программирования в Linux?
39. Как контейнеризация и виртуализация используются в системном программировании?
40. Какие вызовы стоят перед разработчиками системного программного обеспечения в Linux?
41. Как осуществляется управление устройствами и драйверами в Linux?
42. Какие права доступа и модели безопасности существуют для системных ресурсов?
43. Как использовать утилиту strace для отладки системных вызовов?
44. Что такое системные журналы и как их использовать для отладки программ?
45. Какие инструменты используются для работы с исходным кодом в Linux?
46. Как Git помогает в управлении версиями кода?
47. Какие меры безопасности необходимы при разработке сетевых приложений?
48. Как осуществляется автоматизация сборки проектов с использованием Makefile?
49. Какие инструменты используются для анализа производительности и отладки программ в Linux?
50. Как осуществляется профилирование и мониторинг многопоточных приложений?
51. Какие системные вызовы используются для работы с процессами и потоками?
52. Как осуществляется синхронизация данных между процессами и потоками?
53. Какие функции используются для управления разделяемой памятью?
54. Как использовать инструмент Valgrind для выявления утечек памяти?
55. Какие меры предосторожности необходимы при работе с динамическим выделением памяти?
56. Как осуществляется обработка сигналов в Linux?
57. Какие функции используются для создания и управления очередями сообщений?
58. Как реализовать безопасность и защиту данных при системном программировании?
59. Какие инструменты используются для тестирования и верификации системного программного обеспечения?
60. Как интегрировать различные системы и данные в рамках одного программного проекта в Linux?

Заведующий кафедрой _____/ _____
(подпись) (ФИО)

Форма билета для зачета

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПРАВОСУДИЯ»

Образовательная программа 09.02.07 Информационные системы и программирование
(код и наименование программы)

Дисциплина Системное программирование в среде Linux
(наименование дисциплины)

1. Как используется система контроля версий Git в процессе разработки?
2. Как использовать инструмент Valgrind для выявления утечек памяти?

Заведующий кафедрой _____ / _____
(подпись) (ФИО)

Критерии оценивания зачета:

Оценка успеваемости обучающихся на очной форме обучения проводится в соответствии с Положением «О рейтинговой системе оценки успеваемости обучающихся» (утверждено приказом Ректора № 111 от 02.03.2023 года).

ИЗВЛЕЧЕНИЕ

Порядок начисления баллов и оценки успеваемости обучающихся по образовательным программам бакалавриата по очной формы обучения

6.1. Максимальная сумма баллов, набираемая обучающимся по каждой дисциплине, включая промежуточную аттестацию, равна 100 баллам, из них:

- до 40 баллов - по результатам текущего контроля успеваемости;
- до 60 баллов - по результатам промежуточной аттестации.

6.1.1. По результатам текущего контроля успеваемости обучающийся может получить максимально 40 баллов, из них:

- если дисциплина изучается в течение одного семестра'.

• до 14 баллов -за посещаемость учебных занятий;

• до 26 баллов - по результатам учебных занятий и научной работы

6.1.2. Прохождение промежуточной аттестации является обязательным.

Если устный или письменный ответ обучающегося на промежуточной аттестации оценен менее, чем в 16 баллов - баллы, полученные на промежуточной аттестации не суммируются с баллами, набранными обучающимся по результатам текущего контроля успеваемости по данной дисциплине.

В ведомости промежуточной аттестации и в аттестационной ведомости в графе «Результат промежуточной аттестации» проставляется прочерк, в графе «Итого» - сумма баллов по результатам текущего контроля, а в графе «Оценка» - «неудовлетворительно» или «не зачтено».

При успешном прохождении промежуточной аттестации, если устный или письменный ответ обучающегося оценен в 16 и более баллов - баллы суммируются с баллами, набранными обучающимся по результатам текущего контроля успеваемости по данной дисциплине, и переводятся преподавателем в пятибалльную шкалу оценок:

- для дисциплин, по которым предусмотрена промежуточная аттестация в форме зачета:

• менее 37 баллов - неудовлетворительно;

• от 37 до 58 - удовлетворительно;

• от 59 до 79 - хорошо;

• от 80 до 100 – отлично.

Порядок начисления баллов и оценки успеваемости обучающихся по образовательным программам магистратуры по очно-заочной и заочной формам обучения

7.1. Максимальная сумма баллов, набираемая обучающимся по каждой дисциплине, включая промежуточную аттестацию, равна 100 баллам, из них:

- до 40 баллов - по результатам текущего контроля успеваемости;
- до 60 баллов - по результатам промежуточной аттестации.

7.1.1. если дисциплина изучается в течение одного семестра и учебным планом предусмотрено выполнение контрольной работы:

• до 10 баллов -за посещаемость учебных занятий;

• до 10 баллов - по результатам учебных занятий и научной работы;

• до 10 баллов -за контрольную работу.

Выполнение контрольной работы является обязательным.

Баллы, полученные за контрольную работу, соответствуют оценкам:

- если дисциплина изучалась в течение одного семестра:

• менее 4 баллов - не зачтено;

• от 4 до 10 баллов -зачтено

7.2. Прохождение промежуточной аттестации является обязательным.
По результатам промежуточной аттестации обучающийся может получить максимально до 60 баллов.
Если устный или письменный ответ обучающегося на промежуточной аттестации оценен менее, чем в 16 баллов - баллы, полученные на промежуточной аттестации не суммируются с баллами, набранными обучающимся по результатам текущего контроля успеваемости по данной дисциплине.
В ведомости промежуточной аттестации и в аттестационной ведомости в графе «Результат промежуточной аттестации» проставляется прочерк, в графе «Итого» - сумма баллов по результатам текущего контроля, а в графе «Оценка» — «неудовлетворительно» или «не зачтено».
При успешном прохождении промежуточной аттестации, если устный или письменный ответ обучающегося оценен в 16 и более баллов - баллы суммируются с баллами, набранными обучающимся по результатам текущего контроля успеваемости по данной дисциплине, и переводятся преподавателем в пятибалльную шкалу оценок:
для дисциплин, по которым предусмотрена промежуточная аттестация в форме зачета:
• менее 37 баллов - неудовлетворительно;
• от 37 до 58 - удовлетворительно;
• от 59 до 79 - хорошо;
• от 80 до 100 – отлично.