ГОУ ВПО Российско-Армянский (Славянский) университет

Утверждено
Директор Института математики и информатики
Арамян Р.Г.
«21» марта 2025, протокол №9.1

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ДИСЦИПЛИНЫ

Наименование дисциплины: Программирование в среде Linux

Автор: Петросян Давид Арменович

Направление подготовки: 01.03.02 Прикладная математика и информатика
Наименование образовательной программы01.03.02 Прикладная математика и информатика

1. **АННОТАЦИЯ**

1.1. Краткое описание содержания данной дисциплины;

Курс «Программирование в среде Linux» нацелен на формирование глубоких знаний и практических навыков во взаимодействии программного обеспечения с операционной системой и аппаратным обеспечением. Основное внимание уделяется C и C++ в UNIX-подобных операционных системах, в частности Linux.

Дисциплина охватывает ключевые аспекты системного программирования: от работы с файловыми системами и системными вызовами до управления процессами, потоками и синхронизации. Особое внимание уделяется межпроцессному взаимодействию (IPC), включая сигналы, разделяемую память и каналы. Курс завершается изучением основ сетевого программирования с использованием сокетов.

1.2. Трудоемкость в академических кредитах и часах, формы итогового контроля (экзамен/зачет);

Курс «Системное программирование» осваивается в течение одного семестра на 2 курсе бакалавриата (3 семестр).

Виды учебной работы	**В акад. часах**
Общее количество часов	64
Лекция	32
Практика	32
Итоговый контроль	Экзамен

1.3. Взаимосвязь дисциплины с другими дисциплинами учебного плана специальности (направления)

«Введение в C/C++» (или аналогичные курсы по основам программирования на C/C++): Курс предполагает уверенное владение языками C и C++, включая работу с указателями, структурами данных и базовыми алгоритмами.

«Архитектура компьютера» (или аналогичные курсы по архитектуре ЭВМ и низкоуровневому программированию): Эти дисциплины предоставляют необходимую базу для понимания взаимодействия программ с аппаратным обеспечением на низком уровне и принципов работы системных вызовов.

1.4. Результаты освоения программы дисциплины:

Код компетенции	Наименование компетенции	Код индикатора достижения компетенций	Наименование индикатора достижений компетенций
ПК-13	способностью применять существующие и разрабатывать новые методы и средства обучения	13.1	Знать основные теоретические положения разработки математических, информационных и имитационных моделей
		13.2	Применить существующие методы и средства обучения
		13.3	Владеть навыками разработки новых методов и средств обучения
УК-1	Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	1.1	Знает как осуществлять поиск, критический анализ и синтез информации для решения поставленных профессиональных задач
		1.2	Умеет применять системный подход на основе поиска, критического анализа и синтеза информации для решения задач профессиональной области
		1.3	Владеет навыками поиска, синтеза и критического анализа информации в своей

			профессиональной области; владеет системным подходом для решения поставленных задач

2. УЧЕБНАЯ ПРОГРАММА

2.1. Цели и задачи дисциплины

Цель дисциплины «Системное программирование» — сформировать у студентов глубокое понимание принципов взаимодействия программного обеспечения с операционной системой и аппаратными ресурсами, а также развить практические навыки разработки эффективных и надежных системных приложений на языках **C/C++** в среде **UNIX-подобных операционных систем**, в частности **Linux**.

Для достижения этой цели ставятся следующие задачи:

- Изучить основы архитектуры компьютеров и принципы функционирования операционных систем с точки зрения системного программиста.
- Освоить механизмы **системных вызовов** и **API файловых систем** для эффективного управления файлами и каталогами.
- Получить навыки работы с **процессами** и **потоками**, включая их создание, управление и взаимодействие.
- Понять и применять различные механизмы **синхронизации** для обеспечения корректной работы многопоточных и многопроцессных приложений.
- Изучить и практически освоить основные механизмы **межпроцессного взаимодействия (IPC)**, такие как **каналы**, **разделяемая память**, **сигналы** и **сокетов**.
- Приобрести опыт разработки базовых **сетевых приложений** с использованием сокетов.
- Развить навыки отладки, тестирования и оптимизации системного кода.
- Сформировать понимание лучших практик и стандартов в области системного программирования.

2.2. Трудоемкость дисциплины и виды учебной работы (в академических часах и зачетных единицах)

Виды учебной работы	Всего, в акад. часах	Распределение по семестрам					
		3 сем	___ сем	___ сем	___ сем.	___ сем	____ сем.
1	2	3	4	5	6	7	8
1. Общая трудоемкость изучения дисциплины по семестрам, в т. ч.:	**144**						
1.1. Аудиторные занятия, в т. ч.:	**64**	**64**					
1.1.1. Лекции	**32**	**32**					
1.1.2. Практические занятия, в т. ч.	**32**	**32**					
1.2. Самостоятельная работа, в т. ч.:	**53**	**53**					
1.3. Контроль	**27**	**27**					
Итоговый контроль (Экзамен, Зачет, диф. зачет - указать)		**Экзамен**					

2.3. Содержание дисциплины

2.3.1. **Тематический план и трудоемкость аудиторных занятий (модули, разделы дисциплины и виды занятий) по рабочему учебному плану**

Разделы и темы дисциплины	Всего (ак. часов)	Лекции (ак. часов)	Практ. Занятия (ак. часов)	Семинары (ак. часов)	Лабор. (ак. часов)
1	2=3+4+5+6+7	3	4	5	6
Тема1: Введение в системное программирование. Взаимодействие аппаратного и программного обеспечения.	4	2	2		
Тема 2: Обзор файловых систем. Концепции и структура.	4	2	2		
Тема 3: Системные вызовы в UNIX/Linux. Основы работы с командной строкой.	4	2	2		

Тема 4: API файловых систем. Открытие, чтение, запись, закрытие файлов.	**4**	**2**	**2**		
Тема 5: Продвинутый файловый ввод-вывод. Буферизация, перемещение по файлу.	**4**	**2**	**2**		
Тема 6: Процессы: создание, управление, завершение. (fork, exec, wait).	**4**	**2**	**2**		
Тема 7: Потоки: создание и управление. Модель памяти процессов и потоков.	**4**	**2**	**2**		
Тема 8: Синхронизация потоков. Мьютексы, семафоры, условные переменные.	**4**	**2**	**2**		
Тема 9: Задача "Производитель-потребитель". Применение механизмов синхронизации.	**4**	**2**	**2**		
Тема 10: Межпроцессное взаимодействие (IPC): Сигналы.	**4**	**2**	**2**		
Тема 11: Межпроцессное взаимодействие (IPC): Разделяемая память.	**4**	**2**	**2**		
Тема 12: Межпроцессное взаимодействие (IPC): Каналы (pipes) и именованные каналы (fifos).	**6**	**3**	**3**		

Тема 13: Сетевое программирование: Основы сокетов. Клиент-серверные приложения.	**6**	**3**	**3**		
Тема 14: Сетевое программирование: Расширенные возможности сокетов и протоколы.	**8**	**4**	**4**		
ИТОГО	**64**	**32**	**32**		

2.3.2. **Краткое содержание разделов дисциплины в виде тематического плана**

Тема 1. (Введение в системное программирование)

Аппаратное и программное обеспечение.

Обзор основных концепций системного программирования, роли операционной системы и взаимодействия программ с аппаратными компонентами.

Тема 2. (Обзор файловых систем)

Изучение структуры и принципов работы файловых систем, включая иерархию, метаданные и способы хранения данных

Тема 3. (Системные вызовы.)

Понимание механизма системных вызовов как интерфейса между пользовательскими программами и ядром операционной системы

Тема 4. (API файловых систем.)

Практическое освоение стандартных функций и системных вызовов для работы с файлами и каталогами (открытие, чтение, запись, закрытие).

Тема 5. (Расширенный файловый ввод-вывод)

Изучение продвинутых методов и техник работы с файлами, включая неблокирующий ввод-вывод и отображение файлов в память.

Тема 6. (Процессы.)

Детальное рассмотрение жизненного цикла процессов, их создания (fork, exec), управления и завершения в UNIX-подобных системах.

Тема 7. (Потоки.)

Изучение концепции потоков, их создания, управления и преимуществ использования для распараллеливания вычислений.

Тема 8. (Синхронизация)

Изучение основных примитивов синхронизации, таких как мьютексы, семафоры и условные переменные, для обеспечения корректной работы многопоточных приложений.

Тема 9. (Задача «Производитель-потребитель» и пулы потоков (Thread Pool).)

Практическое применение механизмов синхронизации на примере классической задачи, а также изучение концепции пулов потоков для эффективного управления ресурсами.

Тема 10. (Межпроцессное взаимодействие (IPC): Сигналы)

Изучение механизмов сигналов как средства асинхронного уведомления и управления процессами.

Тема 11. (Межпроцессное взаимодействие (IPC): Разделяемая память, Каналы (pipes) и Именованные каналы (fifos).)

Рассмотрение различных методов обмена данными между независимыми процессами, включая общую память и различные типы каналов.

Тема 12. (Введение в сетевые технологии. Физические соединения и сети.)

Обзор базовых понятий сетевых технологий, принципов организации физических соединений и структуры компьютерных сетей.

Тема 13. (Понимание TCP/IP: Транспортные и прикладные протоколы.)

Глубокое изучение стека протоколов TCP/IP, включая функции транспортного (TCP, UDP) и прикладного уровней (HTTP, FTP и др.).

Тема 14. (Введение в сокеты. Сетевое программирование)

Основы работы с сокетами как программным интерфейсом для сетевого взаимодействия, а также разработка простых клиент-серверных приложений.

2.4. Модульная структура дисциплины с распределением весов по формам контролей

Формы контролей	Вес формы (форм) текущего контроля в результирующей оценке текущего контроля (по модулям)		Вес формы промежуточного контроля в итоговой оценке промежуточного контроля		Вес итоговой оценки промежуточного контроля в результирующей оценке промежуточных контролей		Вес итоговой оценки промежуточного контроля в результирующей оценке промежуточных контролей (семестровой оценке)	Веса результирующей оценки промежуточных контролей и оценки итогового контроля в результирующей оценке итогового контроля
Вид учебной работы/контроля	**M1** [1]	**M2**	**M1**	**M2**	**M1**	**M2**		
Контрольная работа *(при наличии)*								
Устный опрос *(при наличии)*								
Тест *(при наличии)*								
Лабораторные работы *(при наличии)*								
Письменные домашние задания *(при наличии)*								
Реферат *(при наличии)*								
Эссе *(при наличии)*								
Проект *(при наличии)*								
Другие формы (при наличии)								
Веса результирующих оценок текущих контролей в итоговых оценках промежуточных контролей						0.4		
Веса оценок промежуточных контролей в итоговых оценках промежуточных контролей						0.6		
Вес итоговой оценки 1-го промежуточного контроля в результирующей оценке промежуточных контролей								
Вес итоговой оценки 2-го промежуточного контроля в								

[1] Учебный Модуль

результирующей оценке промежуточных контролей								
Вес результирующей оценки промежуточных контролей в результирующей оценке итогового контроля								0.4
Вес итогового контроля (Экзамен/зачет) в результирующей оценке итогового контроля								0.6
	$\sum = 1$	$\sum = 1$	$\sum = 1$	$\sum = 1$	$\sum = 1$	$\sum = 1$	$\sum = 1$	$\sum = 1$

3. **Теоретический блок**

3.1. Материалы по теоретической части курса

1. Andrew S. Tanenbaum, Herbert Bos, 2015 Modern Operating Systems
2. Michael Kerrisk, The Linux Programming Interface
3. Robert Love, LINUX System Programming
4. Robert Love, Linux Kernel Development, Third Edition
5. Peter Jay Salzman, Michael Burian, Ori Pomerantz, The Linux Kernel Module Programming Guide
6. Programming via Operating Systems, Vahram Martirosyan, Feb 2022
7. Operating Systems: Design and Implementation, Andrew S. Tanenbaum Albert S. Woodhull, Jan 1997
8. Structured Computer Organization, Andrew S. Tanenbaum, Jan
9. Operating system design: the XINU approach, D. Comer Timothy V. Fossum, Jan
10. Linux Kernel in a nutshell, Greg Kroah-Hartman, Jan 2006

4. **Фонды оценочных средств**

4.1. Планы практических и семинарских занятий
Практические и семинарские занятия направлены на закрепление теоретических знаний и формирование навыков работы в среде Linux.
Темы занятий:
Введение в Linux. Основы командной строки.
Работа с файловой системой. Права доступа.
Потоки, перенаправление и каналы.
Bash-скрипты: синтаксис и написание простых утилит.
Системные вызовы и работа с процессами.
Средства разработки: редакторы, компиляторы, отладчики.
Создание пользовательских утилит.
Использование cron и системных журналов.

4.2. Планы лабораторных работ и практикумов
Примерные лабораторные работы:

Навигация по файловой системе и управление файлами.
Автоматизация задач с помощью Bash-скриптов.
Создание и управление процессами.
Разработка простого текстового приложения на C с использованием системных вызовов.
Отладка программ с использованием gdb.
Создание Makefile и управление сборкой проекта.
Мониторинг процессов и ресурсов системы.
Работа с сетевыми утилитами в Linux.

4.3. Материалы по практической части курса
4.3.1. Учебно-методические пособия
«Linux. Основы системного администрирования» — И. Юмашев
«Программирование в Linux: практический курс» — А. Столяров
4.3.2. Учебные справочники
The Linux Command Line — William Shotts
man-pages (официальные справочные страницы Linux)
4.3.3. Задачники (практикумы)
Сборник задач по программированию в Bash
Практикум по системному программированию в среде Linux (внутренние материалы курса)
4.3.4. Наглядно-иллюстративные материалы
Слайды и схемы по архитектуре Linux
Диаграммы взаимодействия процессов и сигналов
Демонстрационные Bash-скрипты
4.3.5. Другие виды материалов
Видеоуроки по работе в терминале
Интерактивные тренажёры командной строки

4.4. Вопросы и задания для самостоятельной работы студентов
Изучить документацию по утилитам grep, sed, awk.
Написать скрипт архивации и резервного копирования файлов.
Настроить планировщик заданий cron.
Реализовать простую программу на C для чтения и записи файлов.

4.5. Тематика рефератов, эссе и других форм самостоятельных работ
История и философия Unix/Linux
Сравнение shell-интерпретаторов: Bash, Zsh, Fish
Безопасность в Linux: основные механизмы защиты
Роль open-source-сообщества в развитии Linux

4.6. Образцы вариантов контрольных работ, тестов и/или других форм текущих и промежуточных контролей
Тесты с выбором ответа по базовым командам
Задания на написание и анализ Bash-скриптов
Практические задачи по системному программированию

4.7. Перечень экзаменационных вопросов
Структура файловой системы Linux
Отличия между системными вызовами и библиотечными функциями

Потоки ввода/вывода и перенаправление
Основы работы с Bash-скриптами
Управление процессами и сигналами
Сборка программ и использование Makefile
Утилиты мониторинга: top, ps, htop
Принципы безопасности в Linux

4.8. Образцы экзаменационных билетов
Билет №3
Опишите основные этапы работы компилятора в Linux.
Реализуйте Bash-скрипт, проверяющий наличие заданных файлов и создающий архив.
Как работает система сигналов в Linux? Пример использования.

4.9. Образцы экзаменационных практических заданий
Написать Bash-скрипт, создающий резервную копию заданного каталога с логированием.
Реализовать на C программу, создающую дочерний процесс, передающую ему данные через pipe и выводящую результат в файл.
Проанализировать поведение программы с утечкой памяти с помощью valgrind.

4.10. Банк тестовых заданий для самоконтроля
Что делает команда chmod +x script.sh?
Какой утилитой можно узнать текущую загрузку процессора?
Что означает конструкция 2>&1 в Bash?
Какая команда используется для поиска текста в файле?

4.11. Методики решения и ответы к образцам тестовых заданий
Вопрос: Что делает команда ls -l | grep '^d'?
Ответ: Показывает список директорий в текущем каталоге.
Пояснение: ls -l — подробный список; ^d — строки, начинающиеся на "d", что означает директорию.
Вопрос: Как перенаправить stderr в файл?
Ответ: command 2> error.log
Пояснение: 2> указывает на поток ошибок (stderr).

5. **Методический блок**

5.1. Методика преподавания

Преподавание данной учебной дисциплины строится на сочетании лекционных занятий, практических семинаров и самостоятельной работы студентов.

На лекциях излагаются основные концепции программирования в среде Linux, включая работу с командной строкой, системными вызовами, управлением процессами, файловой системой и инструментами автоматизации. Рассматриваются как фундаментальные аспекты операционной системы, так и прикладные темы, такие как скриптинг на Bash, работа с компиляторами и отладчиками, взаимодействие с системными журналами.

Особое внимание уделяется практической направленности курса: студенты учатся писать и отлаживать программы в реальной среде, осваивают инструменты командной строки и получают навыки, необходимые для разработки и администрирования в Unix-подобных системах.