ПРОГРАММА

по дисциплине: **АРХИТЕКТУРА КОМПЬЮТЕРОВ И ОПЕРАЦИОННЫЕ СИСТЕМЫ**
по направлению подготовки: 01.03.02 «Прикладная математика и информатика»
физтех-школа: **ФАКТ**
кафедра: **информатики и вычислительной математики**
курс: 2
семестр: 3

Трудоемкость:
Вариативная часть – 3 зачет. ед.;

лекции – 30 часов	Экзамен – нет
практические (семинарские) занятия – 30 часов	Зачет дифф. – 3 семестр
лабораторные занятия – нет	Заданий (курсовых работ) – 2
	Самостоятельная работа 75 часов

ВСЕГО АУДИТОРНЫХ ЧАСОВ – 60

Структура преподавания дисциплины.

1. Введение. Цели и задачи курса. Понятие о вычислительном комплексе. Системное программное обеспечение и операционные системы. Краткая история эволюции вычислительных систем. Взаимное влияние software и hardware. Автономные, сетевые и распределенные операционные системы. Классификация автономных операционных систем по их назначению и структуре. Архитектурная классификация параллельных и распределенных операционных систем. Знакомство с операционной системой UNIX. Системные вызовы и библиотека libc. Понятия login и password. Упрощенное устройство файловой системы в UNIX. Полные имена файлов. Текущая директория. Относительные имена файлов. Домашняя директория пользователя. Команда man – универсальный справочник. Команды cd и ls. Перенаправление стандартного ввода и стандартного вывода. Простейшие команды работы с файлами – cat, cp, mkdir, mv, rm. Шаблоны имен файлов. Пользователь и группа. Системные вызовы getuid() и getgid(). Команды chown и chgrp. Права доступа к регулярному файлу и к директории. Команда chmod. Маска создания файлов. Команда umask. Редактирование файлов, компиляция и запуск программ.

2. Процессы и их планирование в операционной системе. Понятие процесса. Процесс и программа. Состояния процесса. Управляющий блок процесса и его контекст. Операции над процессами. Переключение контекста. Уровни планирования процессов. Критерии планирования и требования к алгоритмам планирования. Параметры планирования. Вытесняющее и невытесняющее планирование. Алгоритмы планирования: FCFS, RR, SJF, гарантированное планирование, приоритетное планирование, многоуровневые очереди, многоуровневые очереди с обратной связью. Планирование в многопроцессорных и распределенных средах. Понятие процесса в UNIX, его контекст. Идентификация процесса. Краткая диаграмма состояний процессов в UNIX. Иерархия процессов. Системные вызовы getpid() и getppid(). Создание процесса в UNIX. Системный вызов fork(). Завершение процесса. Функция exit(). Параметры функции main() в языке C. Переменные среды и аргументы командной строки. Изменение пользовательского контекста процесса. Семейство функций для системного вызова exec().

3. Кооперация процессов. Взаимодействующие и независимые процессы. Категории средств связи. Установление и завершение связи. Прямая и косвенная адресация. Информационная валентность процессов и средств коммуникации. Симплексная, дуплексная и полудуплексная связь. Потоки ввода вывода и сообщения. Буферизация данных. Надежность обмена информацией. Нити исполнения и их отличие от процессов. Interleaving, race condition и взаимоисключения. Условия Бернстай-

на. Понятие критической секции процесса. Программные алгоритмы организации взаимодействия процессов и предъявляемые к ним требования. Семафоры, мониторы Хора и сообщения. Оптимистические стратегии в задачах синхронизации процессов и потоков. Понятие потока ввода-вывода в операционной системе UNIX. Работа с файлами через системные вызовы и через функции стандартной библиотеки. Файловый дескриптор. Наследование файловых дескрипторов при системных вызовах fork() и exec(). Системные вызовы open(), read(), write(), close(). FIFO и pipe. Системные вызовы pipe(), mknod(), функция mkfifo(). Особенности системных потоковых вызовов при работе с FIFO и pipe. Преимущества и недостатки потокового обмена данными. IPC в UNIX. Пространство имен. Адресация в System V IPC. Функция ftok(). Дескрипторы System V IPC. Разделяемая память. Системные вызовы shmget(), shmat(), shmdt(), shmctl(). Команды ipcs и ipcrm. Нить исполнения (thread) в UNIX, ее идентификатор. Функция pthread_self(). Создание и завершение нити исполнения. Функции pthread_create(), pthread_exit(), pthread_join(). Семафоры в UNIX. Отличие операций над UNIX семафорами от классических операций. Системные вызовы semget(), semop(), semctl(). Понятие о POSIX семафорах. Очереди сообщений в UNIX. Системные вызовы msgget(), msgsnd(), msgrcv(), msgctl(). Понятие мультиплексирования. Мультиплексирование сообщений. Модель взаимодействия процессов клиент–сервер. Неравноправность клиента и сервера.

4. Управление памятью. Связывание адресов. Простейшие схемы управления памятью: схема с фиксированными разделами, своппинг, схема с переменными разделами. Проблема размещения больших программ. Понятие виртуальной памяти. Страничная память. Сегментная и сегментно-страничная организации памяти. Таблица страниц. Ассоциативная память. Иерархия памяти. Размер страницы. Исключительные ситуации при работе с памятью. Стратегии управления страничной памятью: выборки, размещения и замещения страниц. Алгоритмы замещения страниц: FIFO, OPT, LRU и другие. Трэшинг (thrashing). Свойство локальности. Модель рабочего множества. Функционирование менеджера памяти ОС Linux и Windows.

5. Файловые системы. Имена, структура, типы и атрибуты файлов. Операции над файлами. Директории. Операции над директориями. Защита файлов. Методы выделения дискового пространства: непрерывная последовательность блоков, связный список, связный список с индексацией, индексные узлы. Управление свободным и занятым дисковым пространством: битовый вектор, связный список. Асинхронный доступ к файлам. Проблемы надежности и производительности файловых систем. Разделы носителя информации (partitions) в UNIX. Логическая структура файловой системы и типы файлов в UNIX. Организация файла на диске

в UNIX на примере файловой системы s5fs. Понятие индексного узла (inode). Организация директорий (каталогов) в UNIX. Понятие суперблока. Указатель текущей позиции в файле. Системная таблица файлов и таблица индексных узлов открытых файлов. Операции над файлами и директориями. Понятие жестких и мягких связей. Системные вызовы и команды для выполнения операций над файлами и директориями: chmod, chown, chgrp, open(), creat(), read(), write(), close(), stat(), fstat(), lstat(), ftruncate(), lseek(), link(), symlink(), unlink(). Функции для изучения содержимого директорий opendir(), readdir(), rewinddir(), closedir(). Понятие о файлах, отображаемых в память (memory mapped файлах). Системные вызовы mmap(), munmap(). Понятие виртуальной файловой системы. Монтирование файловых систем в UNIX.

6. Система управления вводом-выводом. Общие сведения об архитектуре компьютера. Структура контроллера устройства. Опрос устройств и прерывания. Исключительные ситуации и системные вызовы. Прямой доступ к памяти (Direct Memory Access – DMA). Структура системы ввода-вывода. Систематизация внешних устройств и интерфейс между базовой подсистемой ввода-вывода и драйверами. Функции базовой подсистемы ввода-вывода. Блокирующиеся, не блокирующиеся и асинхронные системные вызовы. Буферизация и кэширование. Spooling и захват устройств. Обработка прерываний и ошибок. Планирование запросов. Алгоритмы планирования запросов к жесткому диску: FCFS, SSTF, SCAN, C-SCAN, LOOK, C-LOOK. Блочные и символьные устройства в UNIX. Понятие драйвера. Блочные, символьные драйверы, драйверы низкого уровня. Файловый интерфейс к драйверам. Коммутатор устройств. Старший и младший номер устройства. Понятие сигнала в UNIX. Способы возникновения сигналов и виды их обработки. Понятия группы процессов, сеанса, лидера группы, лидера сеанса, управляющего терминала сеанса, текущей и фоновой групп процессов. Системные вызовы getpgrp(), setpgrp(), getpgid(), setpgid(), getsid(), setsid(). Системный вызов kill() и команда kill(). Особенности получения терминальных сигналов текущей и фоновой группой процессов. Получение сигнала SIGHUP процессами при завершении лидера сеанса. Системный вызов signal(). Установка собственного обработчика сигнала. Сигналы SIGUSR1 и SIGUSR2. Использование сигналов для синхронизации процессов. Завершение порожденного процесса. Системный вызов waitpid(). Сигнал SIGCHLD и его игнорирование. Возникновение сигнала SIGPIPE при попытке записи в pipe или FIFO, который никто не собирается читать. Понятие о надежности сигналов. POSIX функции для работы с сигналами.

7. Сети и сетевые операционные системы. Причины объединения компьютеров в сети. Сетевые и распределенные операционные системы.

Взаимодействие удаленных процессов как основа работы вычислительных сетей. Многоуровневая модель построения сетевых вычислительных систем. Семейства и стеки протоколов. Эталонная модель OSI/ISO. Удаленная адресация и разрешение адресов. Понятие о DNS. Локальная адресация. Понятие порта. Полные адреса. Понятие сокета (socket). Фиксированная, виртуальная и динамическая маршрутизация. Связь с установлением логического соединения и передача данных с помощью сообщений. Краткая история семейства протоколов TCP/IP. Общие сведения об архитектуре семейства протоколов TCP/IP. Уровень сетевого интерфейса. Уровень Internet. Протоколы IP, ICMP, ARP, RARP. Internet-адреса. Транспортный уровень. Протоколы TCP и UDP. Понятие порта. Понятие encapsulation. Уровень приложений/процессов. Использование модели клиент–сервер для взаимодействия удаленных процессов. Понятие socket в UNIX. Организация связи между удаленными процессами с помощью датаграмм. Организация связи между процессами с помощью установки логического соединения. Сетевой порядок байт. Функции htons(), htonl(), ntohs(), ntohl(). Функции преобразования IP-адресов inet_ntoa(), inet_aton(). Функция bzero(). Системные вызовы socket(), bind(), sendto(), recvfrom(), accept(), listen(), connect().

8. Проблемы безопасности операционных систем. Классификация угроз. Формализация подхода к обеспечению информационной безопасности. Классы безопасности. Политика безопасности. Криптография как одна из базовых технологий безопасности ОС. Шифрование с симметричными и ассиметричными ключами. Правило Кирхгофа. Алгоритм RSA. Идентификация и аутентификация. Пароли, уязвимость паролей. Авторизация. Разграничение доступа к объектам ОС. Домены безопасности. Матрица доступа. Недопустимость повторного использования объектов. Аудит, учет использования системы защиты. Организация защищенных каналов и проблемы безопасности распределенных операционных систем.

Учебно-методическое и информационное обеспечение дисциплины

Основная литература

1. Основы операционных систем. Курс лекций: учеб. пособие для вузов / В. Е. Карпов, К. А. Коньков .— 2-е изд., доп. и испр. — М. : Интернет - Ун-т информац. технологий, 2009, 2011 .— 536 с.
2. Операционная система UNIX: учеб. пособие для вузов / А. М. Робачевский, С. А. Немнюгин, О. Л. Стесик .— 2-е изд., перераб. и доп. — СПб. : БХВ-Петербург, 2005, 2007, 2010 .— 656 с.
3. Северов Д.С. Лекции по архитектуре ЭВМ и языку Ассемблера (см. http://cs.mipt.ru).

4. Коротин П.Н. Лекции по архитектуре ЭВМ и языку Ассемблера (см. http://cs.mipt.ru).

Дополнительная литература

1. Программирование на языке ассемблера IBM PC: [учеб. пособие для вузов] / В. Н. Пильщиков .— М. : ДИАЛОГ-МИФИ, 2005, 2010. — 288 с.
2. Архитектура IBM PC и язык Ассемблера: учеб. пособие для вузов / В. Я. Митницкий ; М-во образования Рос. Федерации, Моск. физ.-техн. ин-т (гос. ун-т) .— М. : МФТИ, 2000 .— 148 с.
3. Язык ассемблера для процессоров Intel: учеб. пособие для вузов. / К. Ирвин; пер. с англ. С. Г. Тригуб .— 4-е изд. — М. : Вильямс, 2005 .— 912 с.
4. Современные операционные системы: учеб. пособие для вузов. / Э. Таненбаум; пер. с англ. Н. Вильчинский, А. Лашкевич. — 3-е изд. — СПб.: Питер, 2015. — 1120 с
5. Операционные системы, в 2 т.: учеб. пособие для вузов. Т. 1. Основы и принципы / Х. М. Дейтел, П. Дж. Дейтел, Д. Р. Чофнес; пер. с англ. под ред. С. М. Молявко .— 3-е изд. — М. : БИНОМ, 2006 .— 1024 с.

Электронные ресурсы

1. http://www.citforum.ru/
2. http://www.opennet.ru/man.shtml (оригиналы и переводы справочных руководств Unix).
3. https://www.freebsd.org/cgi/man.cgi (оригинальные MAN-страницы ОС FreeBSD).

Задание 1

(*срок сдачи 26 октября– 1 ноября*)

1. Напишите программу useless (Unix System Extremely Late Execution Software System), которая читает файл и запускает указанные в нём программы с указанной задержкой от времени старта программы useless. Формат записи в файле: «время задержки в секундах» «программа для выполнения».

Например:

```
5  ls
1  ./test
3  ps
```

2. Для того чтобы не допустить потерю информации при сбое диска, обычно используют резервное копирование файлов (backup). Простейшей формой backup'а является копирование файлов из одного каталога в другой. Этот способ требует много времени и места на диске. Напишите программу, осуществляющую более интеллектуальный подход. Программа должна брать из командной строки два параметра: имена исходного каталога и каталога назначения. Она должна рекурсивно сканировать исходный каталог, делать копии всех файлов, для которых ранее не делались копии или которые были изменены с момента последнего backup'а, размещая их в соответствующих местах директории назначения. После копирования каждого файла должна вызываться команда сжатия gzip. Это уменьшит требуемое дисковое пространство, а файл будет переименована с добавлением расширения .gz. Все возникающие ошибки (нет исходного каталога, файл не доступен для чтения и т.д.) должны корректно обрабатываться с выдачей соответствующего сообщения.

Задание 2

(*срок сдачи 7–13 декабря*)

1. Напишите программу runsim, которая осуществляет контроль количества одновременно работающих Unix-приложений, запущенных с её помощью. Программа читает имя Unix-команды со стандартного ввода и запускает её на выполнение. Количество одновременно работающих команд не должно превышать N, где N – параметр командной строки при запуске runsim. При попытке запустить более чем N приложений выдайте сообщение об ошибке и продолжите ожидание ввода команд на исполнение. Программа runsim должна прекращать свою работу при возникновении признака конца файла на стандартном вводе.

2. На мойке посуды в ресторане работают два человека. Один из них моет посуду, второй вытирает уже вымытую. Времена выполнения операций мытья и вытирания посуды меняются в зависимости от того, что моется. Стол для вымытой, но не вытертой посуды имеет ограниченные размеры. Смоделируйте процесс работы персонала следующим образом: каждому работнику соответствует свой процесс. Времена выполнения операций содержатся в двух файлах. Каждый файл имеет формат записей:

«тип посуды» : «время операции»

Стол вмещает N предметов независимо от их наименования. Значение N задаётся как переменная окружения TABLE_LIMIT перед стартом процессов. Грязная посуда, поступающая на мойку, описывается файлом с форматом записи:

«тип посуды» : «количество предметов»

Записи с одинаковым типом посуды могут встречаться неоднократно.
Организуйте передачу посуды от процесса к процессу:
а) через файл, используя семафоры для синхронизации;
б) через pipe;
в) через сообщения;
г) через разделяемую память;
д) через sockets.