УТВЕРЖДЕНО
Проректор по учебной работе
и довузовской подготовке
А. А. Воронов
27 июня 2017 г.

ПРОГРАММА

по дисциплине: **КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ**
по направлению подготовки: 03.03.01 «Прикладные математика и физика»
физтех-школа: **ФРКТ**
факультет: **ФРТК**
кафедра: **информатики и вычислительной математики**
курс: 2
семестр: 3

Трудоемкость:
Вариативная часть – 4 зачет. ед.;

лекции – 30 часов	Экзамен – нет
практические (семинарские) занятия – нет	Зачет дифф. – 3 семестр
лабораторные занятия – 60 часов	Две контрольные работы
	Самостоятельная работа 90 часов

ВСЕГО АУДИТОРНЫХ ЧАСОВ – 90

Программу составили: академик РАН В. П. Иванников,
доцент, к.ф.-м.н. В. Е. Карпов,
доцент, к.ф.-м.н. К. А. Коньков

Программа принята на заседании
кафедры информатики и вычислительной математики
16 июня 2017 г.

Заведующий кафедрой,
чл.-корр. РАН И. Б. Петров

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.

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.

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 С. 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 Бернстайна. Понятие критической секции процесса. Программные алгоритмы

организации взаимодействия процессов и предъявляемые к ним требования. Аппаратная поддержка взаимоисключений. Семафоры, мониторы Хора и сообщения. Доказательство эквивалентности мониторов, семафоров и сообщений.

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. *Thrashing*. Свойство локальности. Модель рабочего множества. Аппаратно-независимая модель памяти процесса.

5. **Файловые системы.** Имена, структура, типы и атрибуты файлов. Операции над файлами. Директории. Операции над директориями. Защита файлов. Интерфейс файловой системы и ее общая структура. 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. Размер блока на диске. Реализация директорий. Монтирование файловых систем. Надежность и целостность 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 *SIGUSR*1 и *SIGUSR*2. 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. Локальные и глобальные вычислительные сети. Топология компьютерных сетей. Взаимная синхронизация вычислительных комплексов: обнаружение коллизий (*CSMA*/*CD*), метод передачи эстафетной палочки (*token passing*), использование слотов для данных. Удаленная адресация и разрешение адресов. Понятие о *DNS*. Локальная адресация. Понятие порта. Полные адреса. Понятие сокета (*socket*). Фиксированная, виртуальная и динамическая маршрутизация. Сети, коммутируемые цепями, сообщениями и пакетами данных. Связь с установлением логического соединения и передача данных с помощью сообщений. Многоуровневая модель построения сетевых вычислительных систем. Семейства и стеки протоколов. Эталонная модель *OSI*/*ISO*.

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.

Учебно-методическое и информационное обеспечение дисциплины

Основная литература

1. *Карпов В. Е., Коньков К. А.* Основы операционных систем. – М.: ИНТУИТ.РУ «Интернет-университет информационных технологий», 2005.

Дополнительная литература

1. *Столингс В.* Операционные системы. – М.: Издательский Дом «Вильямс», 2001.
2. *Олифер В.Г., Олифер Н.А.* Сетевые операционные системы. – СПб.: Питер, 2001.
3. *Стивенс У. UNIX*: Взаимодействие процессов. – СПб.: Питер, 2002.
4. *Стивенс У. UNIX*: Разработка сетевых приложений. – СПб.: Питер, 2003.
5. *Таненбаум Э.* Современные операционные системы. – СПб.: Питер, 2002.
6. *Дейтел Х.М., Дейтел П.Дж., Чофнес Д.Р.* Операционные системы. Основные принципы: 3-е издание. – М.: ООО «Бином-Пресс», 2006.

Электронные ресурсы

1. http://cs.mipt.ru
2. http://acm.mipt.ru

Задание 1

(*срок сдачи 23–28 октября*)

1. Напишите программу *useless* (*UNIX SYSTEM EXTREMELY LATE EXECUTION SOFTWARE SYSTEM*), которая читает файл и запускает указанные в нем программы с указанной задержкой от времени старта программы *useless*. Формат записи в файле:

<время задержки в секундах> <программа для выполнения>

Файл не является упорядоченным по временам задержки.

2. Для того чтобы не допустить потерю информации при порче диска, обычно используют резервное копирование файлов (*backup*). Простейшей формой *backup*'а является копирование всех файлов из одной директории в другую. Этот способ требует много времени и места на диске. Напишите программу, осуществляющую более интеллектуальный подход.

Программа должна брать из командной строки два параметра: исходную директорию и директорию назначения. Она должна рекурсивно сканировать исходную директорию, 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 требуемый размер дисковой памяти; а файл будет переименован с добавлением расширения *.gz*. Все возникающие ошибки (нет исходной директории, файл закрыт для чтения и т.д.) должны корректно обрабатываться с выдачей соответствующего сообщения.

Задание 2

(*срок сдачи 4–9 декабря*)

1. Напишите программу *runsim*, осуществляющую контроль количества одновременно работающих *UNIX*-приложений. Программа читает *UNIX*-команду со стандартного ввода и запускает ее 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. 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* задается как параметр среды *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*.

Усл. печ. л. 0,5. Тираж 150 экз.