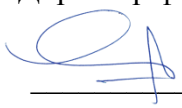
Федеральное государственное бюджетное образовательное
учреждение высшего образования
Московский государственный университет имени М.В.Ломоносова
Филиал Московского государственного университета имени М.В.Ломоносова
в городе Сарове

УТВЕРЖДАЮ

Директор филиала МГУ в городе Сарове



_____/В.В. Воеводин/

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Наименование дисциплины:

«Современные операционные системы (семейство Unix)»

Уровень высшего образования:

магистратура

Направление подготовки / специальность:

02.04.02 "Фундаментальная информатика и информационные технологии" (3++)

Направленность (профиль)/специализация ОПОП:

Суперкомпьютерные технологии и фундаментальная информатика

Форма обучения:

очная

Саров 2022

Рабочая программа дисциплины (модуля) разработана в соответствии с самостоятельно установленным МГУ образовательным стандартом (ОС МГУ) для реализуемых основных профессиональных образовательных программ высшего образования по направлению подготовки 02.04.02 "Фундаментальная информатика и информационные технологии" программы магистратуры в редакции приказа МГУ от 30 декабря 2020 г. №1366

1. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ

1.1. Целями дисциплины являются приобретение студентами знаний и навыков по современным операционным системам семейства Unix, включая:

- основы архитектуры операционных систем и их функции;
- основные понятия, связанные с процессами, потоками и их взаимодействием;
- основные понятия, связанные с файловой системой;
- системное программирование на языке Си;
- инструментальные средства разработки системных программ.

1.2. Задачи дисциплины: ознакомление с принципами построения, параметрами и характеристиками операционных систем; основными методами и технологиями, используемыми в операционных системах; теоретическими основами организации взаимодействия процессов; практическим инструментарием построения системных программ.

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОПОП

А. Информация об образовательном стандарте и учебном плане.

Тип Стандарта: магистр филиала МГУ в Сарове, учебный план магистратуры;

- **направление подготовки** «Фундаментальная информатика и информационные технологии»,
- **наименование учебного плана** «Суперкомпьютерные технологии и фундаментальная информатика»

Б. **Место дисциплины в образовательном стандарте и учебном плане:**

- вариативная часть (дисциплина по выбору студента);

В. Перечень дисциплин, которые должны быть освоены для начала освоения данной дисциплины: «Современные методы и технологии программирования».

Г. Общая трудоемкость (в ак. часах и зачетных единицах): 108 ак. ч., 3 зач. ед.

3. ПЕРЕЧЕНЬ КОМПЕТЕНЦИЙ, ФОРМИРУЕМЫХ В РЕЗУЛЬТАТЕ ОСВОЕНИЯ ДИСЦИПЛИНЫ

В результате изучения данной дисциплины у обучающихся формируются следующие компетенции:
инструментальные:

- владение навыками использования программных средств и работы в операционных системах, использования ресурсов операционных систем; владение основными методами, способами и средствами получения, хранения, переработки информации (ИК-3);
- способность использовать современную вычислительную технику и специализированное программное обеспечение в научно-исследовательской работе (ИК-4);

системные:

- способность к творчеству, порождению инновационных идей, выдвижению самостоятельных гипотез (СК-1);
- способность к поиску, критическому анализу, обобщению и систематизации научной информации, к постановке целей исследования и выбору оптимальных путей и методов их достижения (СК-2);

— способность к самостоятельному обучению и разработке новых методов исследования, к изменению научного и научно-производственного профиля деятельности; к инновационной научно-образовательной деятельности (СК-3);

Профессиональные компетенции:

в области научно-исследовательской деятельности:

— способность демонстрации общенаучных базовых знаний естественных наук, прикладной математики и информатики, понимание основных фактов, концепций, принципов и теорий, связанных с прикладной математикой и информатикой (ПК-1);

в проектной и производственно-технологической деятельности:

— способность применять в профессиональной деятельности современные языки программирования и методы параллельного взаимодействия процессов в операционных системах, электронные библиотеки и пакеты программ, сетевые технологии (ПК-3);

— способность осваивать информационные и суперкомпьютерные технологии при решении практических задач (ПК-4);

— способность собирать, обрабатывать и интерпретировать экспериментальные данные, необходимые для проектной и производственно-технологической деятельности (ПК-5);

в организационно-управленческой деятельности:

— способность составлять и контролировать план выполняемой работы, планировать необходимые для выполнения работы ресурсы, оценивать результаты собственной работы (ПК-8);

в инновационной деятельности:

— способность приобретать новые научные и профессиональные знания, используя современные образовательные и информационные технологии (ПК-10);

— умение работать самостоятельно и в коллективе, руководить людьми, разъяснять и самостоятельно выполнять порученные задания (ПК-14).

В результате освоения дисциплины обучающийся должен

знать

— основные характеристики архитектур и системной организации вычислительной системы, компьютеров и операционных систем и взаимосвязь их основных компонентов;

— основные системные задачи и проблемы, решаемые в рамках операционных системам;

— типовую структуру операционной системы, задачи и основные характеристики функциональных модулей, составляющих операционную систему; основные понятия, алгоритмы и методы организации управления процессами в операционных системах;

— основные понятия, алгоритмы и методы организации взаимодействия процессов; основные понятия, алгоритмы и методы организации файловых систем; основные понятия, алгоритмы и методы организации планирования в операционных системах;

— основные понятия, алгоритмы и методы организации управления внешними устройствами;

— основные понятия, алгоритмы и методы организации управления оперативной памятью;

уметь

— формировать обоснованную оценку организации и функционирования тех или иных компонентов операционных систем в контексте их системной взаимосвязи с аппаратурой компьютера;

— использовать современные операционные системы;

— разрабатывать элементы распределенных компонентов системного программного обеспечения, основанных на использовании библиотек системных вызовов: использовать современные языки разработки системного программного обеспечения (язык Си);
— создавать алгоритмические модели типовых задач, проводить спецификацию задачи, реализовывать программы на алгоритмических языках высокого уровня, оценивать сложность полученных алгоритмов;

владеть

— профессиональными знаниями теории и практики операционных систем и методов разработки и реализации операционных систем, основами организации библиотеки системных вызовов;
— навыками решения практических задач, связанных с разработкой программного обеспечения на основе использования библиотек системных вызовов и системных библиотек программ;
— основами алгоритмизации, пониманием методов построения алгоритма на основе разбиения задачи на подзадачи;
— разработкой компонентов программного обеспечения в среде ОС UNIX с использованием языка программирования Си;
— базовыми навыками разработки параллельных программ на основе использования различных средств взаимодействия процессов ОС UNIX (базовые средства взаимодействия процессов ОС UNIX, IPC, сокеты и др.).

4. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

Общая трудоемкость дисциплины составляет 3 зачетные единицы или 108 ч., из них:

— лекции –36 часов;
— семинары – 0 часов;
— лабораторная работа – 0 часов;
— самостоятельная работа – 72 часа.

Форма контроля – зачет.

4.1. Распределение трудоемкости по разделам и темам, а также формам проведения занятий с указанием форм текущего контроля и промежуточной аттестации:

№ п/п	Наименование разделов и тем дисциплины /	Трудоемкость (в ак. часах) по формам занятий (для дисциплин) и видам работ (для практик)			Формы контроля
		Аудиторная работа (с разбивкой по формам и видам)		*Самостоя-тельная работа*	
		Лекции	*Практические занятия (семинары) / Полевые работы*		
1	**Раздел** 1. Операционные системы				
	Тема 1. Введение. Основы архитектуры	4	0	12	Домашнее задание

	вычислительной системы				
	Тема 2. Управление процессами	8	0	24	Домашнее задание, обсуждение
	Тема 3. Реализация межпроцессного взаимодействия в UNIX	8	0	24	Домашнее задание
	Тема 4. Файловые системы	8	0	24	Домашнее задание
	Тема 5. Управление памятью и внешними устройствами	4	0	12	Домашнее задание
	Тема 6. Инструментарий для системного программирования в ОС	4		12	Коллоквиум, контрольная работа
2	***Итого***	***36***	***0***	***72***	***зачет***

4.1. Содержание дисциплины по разделам и темам – аудиторная и самостоятельная работа:

Раздел 1. Операционные системы

Тема 1. **Введение. Основы архитектуры вычислительной системы**

Содержание*:* История развития вычислительной техники. Ранние вычислительные машины. Архитектурные принципы фон Неймана. Четыре поколения ЭВМ; ранние операционные системы. Задачи современных операционных систем.

История ОС Unix. ОС Unix и язык Си. Современные представители семейства Unix. Начальные сведения о работе в операционной среде Unix.

Задания для самостоятельной работы: Изучение темы "Этапы развития вычислительной техники и программного обеспечения".

Изучение темы "Основы архитектуры вычислительной системы".

Изучение темы "Основы архитектуры компьютеров".

Изучение темы "Основы архитектуры операционных систем".

Тема 2. **Управление процессами**

Содержание*:* Основные концепции. Определение процесса. Модельная ОС. Жизненный цикл, состояния процесса. Модель пакетной однопроцессной ОС, модель пакетной мультипроцессной системы. Модель ОС с разделением времени. Основные типы процессов. “Полновесные процессы”. ”Легковесные процессы”. Контекст процесса. Реализация процессов в ОС UNIX. Контекст процесса. Тело процесса. Аппаратный контекст. Системный контекст. Состояния процесса. Аппарат системных вызовов в ОС UNIX. Базовые средства управления процессами в ОС UNIX (fork() ,exec(), wait(), exit()…). Планирование.

Совокупность задач планирования. Планирование распределения времени ЦП между процессами – основные подходы: вытесняющие и невытесняющие стратегии; алгоритмы, основанные на квантовании (простой круговорот, алгоритмы с изменяющимся квантом времени и т.д.); алгоритмы, использующие приоритет (планирование по наивысшему приоритету, понятие относительного и абсолютного приоритета, класс алгоритмов, использующих линейно (нелинейно) изменяющийся приоритет, очереди с обратной связью (неявный приоритет)); смешанные алгоритмы планирования. Особенности планирования в системах реального времени. Примеры: организация планирования времени ЦП в ОС UNIX.

Взаимодействие процессов. Взаимодействие параллельных процессов и их синхронизация. Классификация средств межпроцессного взаимодействия. Разделяемые ресурсы и синхронизация доступа к ним. Взаимное исключение. Тупики. Некоторые способы реализации взаимного исключения: семафоры Дейкстры, мониторы, обмен сообщениями.

Классические задачи синхронизации процессов: “обедающие философы”, “читатели и писатели”, “спящий парикмахер”.

Задания для самостоятельной работы:

Решение теоретических задач по управлению процессами.

Написание программ реализации процессов в ОС UNIX.

Написание программ по теме "Взаимодействие процессов".

Тема 3. **Реализация межпроцессного взаимодействия в UNIX**

Содержание*:* Базовые средства реализации взаимодействия процессов в ОС UNIX. Сигналы. Работа с сигналами. Примеры программирования (signal(), kill()). Надежные сигналы. Неименованные каналы. Примеры программирования (pipe(),dup(),read(),write()). Именованные каналы (FIFO). Примеры программирования (mkfifo()). Взаимодействие процессов по схеме “подчиненный – главный” (ptrace”()). Общая схема трассировки процессов. IPC – система межпроцессного взаимодействия. Общие концепции. Проблема именования разделяемых объектов. Объекты IPC. Очередь сообщений (создание, доступ, управление). Разделяемая память (создание, доступ, управление). Массив семафоров (создание, доступ, управление). Сокеты – унифицированный интерфейс программирования программ взаимодействующих через сеть. Типы сокетов. Коммуникационный домен. Дейтаграммное соединение. Соединение с использованием виртуального канала. Схема работы с сокетами с установлением соединения. Схема работы с сокетами без установления соединения.

Задания для самостоятельной работы: Изучение базовых средств реализации взаимодействия процессов.

Изучение и написание программ на тему "система межпроцессного взаимодействия IPC".

Реализация клиент-серверного приложения на базе механизма сокетов.

Тема 4. **Файловые системы**

Содержание*:* Основные концепции. Структурная организация файлов. Атрибуты файлов. Основные правила работы с файлами. Типовые программные интерфейсы работы с файлами. Подходы в практической реализации файловой системы. Модели реализации файлов. Понятие индексного узла (дескриптора). Модели реализации каталогов.

Взаимно-однозначное соответствие идентификатора файла и содержимого файла. Координация использования пространства внешней памяти. Квотирование пространства файловой системы. Надежность файловой системы. Проверка целостности файловой системы.

Примеры реализаций файловых систем. Организация файловой системы ОС UNIX. Виды файлов. Права доступа. Логическая структура каталогов. Внутренняя организация ФС. Модель версии SYSTEM V– суперблок, область индексных дескрипторов, блоки файлов.

Работа с массивами номеров свободных блоков. Индексный дескриптор. Работа с массивом свободных индексных дескрипторов. Адресация блоков файла. Файл каталог. Достоинства и недостатки реализации. Модель версии FFS BSD. Стратегия размещения данных. Внутренняя организация блоков. Алгоритм выделения пространства для файла. Структура каталога FFS.

Задания для самостоятельной работы: Примеры реализации файловых систем: изучение на примере современных систем NTFS, ext3.

Тема 5. **Управление памятью и внешними устройствами**

Содержание*:* Управление оперативной памятью. Базовые концепции, задачи и стратегии управления оперативной памятью. Организация управления памятью при: одиночном непрерывном распределении; распределении разделами; распределении перемещаемыми разделами; страничном распределении (таблица страниц, TLB, иерархическая организация

таблицы страниц, хэширование таблицы страниц, инвертированные таблицы станиц, алгоритмы замещения страниц); сегментном распределении; сегментно-страничном распределении.

Управление внешними устройствами. Архитектура организации управления внешними устройствами. Программное управление внешними устройствами. Драйверы физических и логических устройств. Буферизация обмена. Планирование дисковых обменов. Примеры алгоритмов. RAID-системы. Уровни RAID. ОС UNIX – работа с внешними устройствами. Файлы устройств, драйверы. Системные таблицы драйверов устройств. Ситуации, вызывающие обращения к функциям драйвера. Включение, удаление драйверов в систему.

Организация обмена данных с файлами. Пример. Буферизация при блок-ориентированном обмене. Борьба со сбоями.

Задания для самостоятельной работы: Изучение стратегий управления оперативной памятью. Работа с внешними устройствами на примере ОС UNIX.

Тема 6. **Инструментарий для системного программирования в ОС**

Содержание*:* Пользовательские средства ОС Unix: текстовый редактор vim, командный интерпретатор shell и его основные команды, работа с файлами, запуск процессов, перенаправление ввода-вывода, конвейеризация.

Инструментарий программиста: компилятор gcc языка Си, отладчик gdb, утилита make, средство valgrind.

Типовые задачи для домашних заданий.

1. Пусть дано восьмеричное число 4321475, являющееся адресом оперативной памяти, расслоенной по четырем банкам. Банку с каким номером принадлежит заданный адрес?

2. Пусть дано восьмеричное число 173367, являющееся адресом оперативной памяти, расслоенной по 32 банкам. Банку с каким номером принадлежит заданный адрес?

3. Пусть дано восьмеричное число 125432, являющееся адресом оперативной памяти, расслоенной по 8 банкам. Банку с каким номером принадлежит заданный адрес?

4. В оперативном запоминающем устройстве 16-ти разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове четверичного числа 103323.

5. В оперативном запоминающем устройстве 32-х разрядного компьютера используется контроль целостности данных по четности. Описать возможную структуру ячейки памяти и ее побитовое содержимое для случая хранения в машинном слове восьмеричного числа 3560271.

6. Что такое Операционная система? Каковы ее главные функции?

7. Перечислите необходимые аппаратные возможности для поддержки мультипрограммного режима в ОС.

8. Сколько байт в памяти размером 1 терабайт?

9. Сколько нужно оперативной видеопамяти для поддержки текстового монохромного экрана размером 25x80 строк?

10. Сколько нужно оперативной видеопамяти для поддержки 24-битового цветного отображения на экране размером 1024x768 пикселов?

11. Дан 32-разрядный IP-адрес, имеющий в восьмеричном представлении вид: 15502410544. Определить, к какому классу относится данный IP-адрес, номер сети (в восьмеричном представлении), к которой относится IP адрес.

12. Дан 32-разрядный IP-адрес, имеющий в шестнадцатеричном представлении вид: DF00BE20. Определить, к какому классу относится данный IP-адрес, номер сети (в 16-ричном представлении), к которой относится IP адрес, и десятичный номер хоста в сети.

13. Пусть процесс с PID = 2021 породил два сыновних процесса с PID'ами 2022 и 2023:

```
int main (int argc, char ** argv)   /* PID=2021 */
{
  If (fork()==0) { /*PID = 2022 */
    printf("%d %d\n", getppid(), getpid());
    exit(0);
  }
  If (fork()==0) { /*PID = 2023 */
   printf("%d\n", getpid());
   exit(0);
  }
  return 0;
}
```

Считаем, что printf работает атомарно (печатает сразу всё) и обращения ко всем системным вызовам успешно срабатывают. Перечислить ВСЕ возможные комбинации значений, которые могут быть выведены на стандартное устройство вывода в результате выполнения данной программы.

14. Пусть процесс с PID = 3047 породил два сыновних процесса с PID'ами 3048 и 3049:

```
int main (int argc, char ** argv)   /* PID=3047 */
{
  If (fork()==0) { /*PID = 3048 */
    printf("%d\n",  getpid());
    exit(0);
  }
  wait(NULL);
  If (fork()==0) { /*PID = 3049 */
   printf("%d %d\n", getpid(), getppid());
   exit(0);
  }
  return 0;
}
```

Считаем, что printf работает атомарно и обращения ко всем системным вызовам успешно срабатывают. Перечислить ВСЕ возможные комбинации значений, которые могут быть выведены на стандартное устройство вывода в результате выполнения данной программы.

15. Написать два варианта программы, которая исполняет следующую команду shell:

```
gcc  1.c -o 1.exe -lm
```

16. Реализовать конвейер из N процессов: pr1 | pr2 | ... |prN

То есть написать программу prog с аргументами pr1 pr2 ... prN, которая ведет себя так же, как заданный в командной строке конвейер. Обратите внимание на то, что конструкция вида

pr1 | pr2 | ... |prN

должна завершаться (т.е. шелл должен выдавать приглашение на ввод следующей команды) лишь тогда, когда завершатся все входящие в нее команды (а не только prN). Например, конструкция sleep 5 | ls завершится через 5 секунд, хотя ls завершится намного раньше. Если непонятно, как должна вести себя та или иная конструкция,

рекомендуется проверять ее в bash. Конструкция yes | head должна завершаться, и после нее должно оставаться незавершенных процессов (кроме bash).

Проверить существующие в данные момент процессы можно с помощью команды ps.

17. Написать программу, моделирующую команду SHELL: (здесь pri - имена процессов, argj - аргументы процессов, f.dat - файл входных данных, f.res - файл результатов; в каждом из процессов pri использован стандартный ввод-вывод). Аргументы, необходимые этой программе, задаются в командной строке.

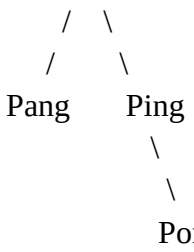
1)pr1 | pr2 > f.res

2) pr1; pr2; ... ; prn

3) pr1 < f.dat > f.res

18. Написать программу, моделирующую команду SHELL pr1&&pr2 (выполнить pr1; в случае успешного завершения pr1 выполнить pr2, иначе за-вершить работу). Имена процессов задаются в командной строке. Процесс считается выполненным успешно, если он вернул 0.

19. Организовать игру в волейбол (по аналогии с игрой в Пинг-Понг в материалах) между тремя китайскими министрами по имени Пинг, Панг и Понг. Министры-процессы порождаются по схеме:



Передача "мяча" осуществляется через каналы (3 канала) по кругу: либо по часовой стрелке Pang-->Ping-->Pong-->Pang-->... либо против часовой.

20. С помощью именованного канала организовать клиент-серверную обработку данных следующим образом. Процесс-сервер запускается первым, обнуляет целую переменную-сумматор, создает именованный канал, открывает его на чтение, затем в бесконечном цикле читает очередное целое из канала (во внутреннем представлении int) и добавляет считанное число к сумматору. Получив сигнал SIGINT (сделать соответствующий обработчик сигнала), сервер выводит на экран текущее значение сумматора, закрывает и удаляет именованный канал, и завершается. Процесс-клиент получает в качестве argv[1] число в строковом виде, переводит его во внутреннее представление (число типа int) , открывает именованный канал на запись и отправляет число серверу, после чего завершается. Запустить сразу несколько параллельных клиентов из командной строки можно так: $> client 5 & client 3 & client 4 & client 1

Ответ сервера после нажатия Ctl-C: sum=13

21. Организовать игру Пинг-Понг между двумя процессами-братьями, используя синхронизацию с помощью очереди сообщений. Очередь создает (ключ IPC_PRIVATE) и удаляет процесс-отец, сыновья ее наследуют. Игра идет до нажатия Ctll-C — отец должен обработать эту ситуацию и удалить очередь.

22. Организовать игру Пинг-Понг между двумя процессами через семафоры и разделяемую память. Первый процесс создает массив из двух семафоров и устанавливает значения <1, 0> для него, записывает в разделяемую память слово «Ping». Далее первый процесс в бесконечном цикле : опускает первый семафор, считывает из разделяемой памяти слово и печатает на экран, спит одну секунду — sleep(1) , записывает в разделяемую память слово Pong, после чего поднимает второй семафор и идет на начало цикла. Второй процесс подсоединяется к созданным ресурсам и в

бесконечном цикле: опускает второй семафор, считывает из разделяемой памяти слово и печатает на экран, спит одну секунду — sleep(1), записывает в разделяемую память слово Ping, после чего поднимает первый семафор и идет на начало цикла. При получении сигнала SIGINT первый процесс удаляет массив семафоров и разделяемую память и завершается, а второй просто завершается.

23. Реализовать программу, которая запускает пять процессов-философов из классической задачи о пяти философах (при этом каждый философ печатает на экран свой номер или имя, и что сейчас делает – гуляет, кушает).

24. Написать клиент-серверную программу «Калькулятор» :

Процесс-сервер – создает "слушающий" сокет и ждет запросов на соединение от клиентов.

Приняв запрос, процесс-сервер создает сыновний процесс, который должен обслужить клиента и завершиться, а процесс-отец продолжает принимать запросы на соединение и создавать новых сыновей.

Задача сына (обслуживание) – выполнять возможные команды от клиента:

1) \+ <число> – установить число на которое сервер будет увеличивать числа для данного клиента (по умолчанию 1), вернуть клиенту “Ok”;

2) <число> – запрос на увеличение числа от клиента, задача – увеличить данное число на заданное и вернуть клиенту;

3) \? – получить от сервера число, на которое тот увеличивает числа для текущего клиента;

4) \- – сообщить серверу о завершении работы, при этом сервер-сын завершается.

Программа-клиент:

1) запрашивает у пользователя очередную команду, отправляет ее серверу (адрес сервера можно задать в командной строке (передаются в main() через argv), или спросить у пользователя первым действием;

2) получает от сервера ответ и печатает его на экран.

В качестве программы клиента возможно использование утилит telnet или netcat (в разных системах может называться nc, netcat, ncat, pnetcat, не входит в стандарт POSIX).

25. Написать клиент-серверную программу «Чат»

Процесс-сервер:

1) создает "слушающий" сокет и ждет запросов на соединение от клиентов;

2) при поступлении запроса, устанавливается соединение с очередным клиентом, от клиента сервер получает имя (ник) вошедшего в "комнату для разговоров", клиент заносится в список присутствующих;

3) всем присутствующим рассылается сообщение, что в комнату вошел такой-то (имя);

4) от разных клиентов могут поступать реплики – получив реплику от клиента, сервер рассылает ее всем "присутствующим" (включая самого автора) с указанием автора реплики;

5) при разрыве связи (команда \quit) с клиентом сервер сообщает всем, что-такой-то (имя) нас покинул (ушел) и выводит его прощальное сообщение.

Необходима реализация команд:

а) \users получить от сервера список всех пользователей (имена), которые сейчас онлайн;

б) \quit <message> – выход из чата с прощальным сообщением.

В качестве программы клиента возможно использование telnet (или nc, netcat) . Имена пользователям могут выдаваться автоматически, но они должны быть уникальными. Иной подход – сервер запрашивает имя у клиента

26. Реализовать многомодульный проект на языке Си для работы с матрицами и написать make-файл к нему.

27. Реализовать с помощью скрипта для bash калькулятор выражений с числовыми константами, выражения удовлетворяют синтаксису языка Си. Например: (5%2+3)>>4 . Использовать метод динамической компиляции: получаем у пользователя выражение, которое вставляем в заготовку Си-файла, компилируем этот файл, запускаем получившийся исполняемый файл, который вычислит ответ и напечатает результат.

28. Написать на языке bash командный файл, осуществляющий автоматизированное тестирование какой-нибудь программы (например, калькулятора).

5. ИСПОЛЬЗУЕМЫЕ ОБРАЗОВАТЕЛЬНЫЕ, НАУЧНО-ИССЛЕДОВАТЕЛЬСКИЕ И НАУЧНО-ПРОИЗВОДСТВЕННЫЕ ТЕХНОЛОГИИ

А. Образовательные технологии.

Работа в аудитории: лекции; консультации перед экзаменом.
Процесс изложения учебного материала сопровождается презентациями. При чтении лекций используются элементы интерактивности – наиболее важные элементы лекций обсуждаются с аудиторией в режиме «вопрос-ответ».

Внеаудиторная работа: изучение пройденных на лекциях тем; самостоятельное изучение литературы по операционным системами; самостоятельная работа, включающая выполнение практических заданий, предназначенная для закрепления теоретической части курса и получения практических навыков их применения (в объеме 72 часов).

Б. Научно-исследовательские технологии: в ходе самостоятельной работы студенты изучают функционирование операционных систем.
В. Научно-производственные технологии: используются технологии работы с современным программным обеспечением.

6. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ. ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ИТОГАМ ОСВОЕНИЯ ДИСЦИПЛИНЫ.

А. Учебно-методические рекомендации для обеспечения самостоятельной работы студентов: в ходе самостоятельной работы студенту следует использовать конспект лекций, основную и дополнительную литературу по курсу и Интернет-ресурсы.

Б. Примерный список заданий для проведения текущей и промежуточной аттестации (темы для докладов, рефератов, презентаций):

1. Архитектура ОС, ее определение.
2. Организация параллельных вычислений
3. Файловая структура.
4. Файлы и каталоги.
5. Файловые системы.
6. Типовые программные интерфейсы работы с файлами.
7. Надежность и проверка целостности файловой системы.
8. Взаимодействие параллельных процессов и их синхронизация.
9. Вытесняющие и невытесняющие стратегии по распределению времени между процессами.

10. Взаимоисключение и способы его реализации.
11. Тупики.
12. Задачи управления виртуальной памятью.
13. Варианты организации оперативной памятью.
14. Алгоритмы замещения страниц.
15. RAID-системы.
16. Архитектурные принципы фон Неймана.
17. Современные операционные системы.
18. Этапы развития ПО.

В. Примерный список вопросов для проведения текущей и промежуточной аттестации:

1. Задачи современных ОС. Основы архитектуры ОС.
2. Именованные каналы FIFO.
3. ОС UNIX и язык Си.
4. Проблемы взаимодействия параллельных вычислительных процессов.
5. Синхронизация параллельных вычислительных процессов.
6. Возникновение тупиковых ситуаций.
7. Проблемы управления вычислительными ресурсами на примере оперативной памяти. Схемы управления.
8. Массив семафоров (создание, доступ, управление).
9. Взаимодействие процессов с помощью сигналов.
10. Примеры реализации файловых систем.
11. Утилита make и средство valgrind для сборки и отладки системных программ.
12. Управляемый прерываниями ввод-вывод.
13. Классические задачи синхронизации процессов.
14. Сокеты, их типы.
15. Типы файлов в операционной системе Unix.
16. Описание клиент-серверного приложения на сокетах.

7. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

7.1. Основная литература:

	Автор	**Название книги/статьи**	**Место издания**	**Издательство**	**Год издания**
1	Волкова И.А., Головин И.Г., Столяров А.В.	Операционная система Unix	Москва	ВМК МГУ	2006
2	Керниган Б., Ритчи Д.	Язык программирования Си.	Москва	Диалектика	2017
3	Столлингс В.	Операционные системы. Внутреннее устройство и принципы проектирования	Москва	Диалектика	2020
4	Танненбаум Э.	Современные операционные системы	СПб	Питер	2015
5	М.А.Казачук, И.В.Машечкин, И.С.Попов, А.Н.Терехин, В.В.Тюляева	Программирование в ОС UNIX на языке Си	Москва	МаксПресс	2020

7.2. Дополнительная литература:

	Автор	**Название книги/статьи**	**Место**	**Издательс**	**Год**

			издания	**тво**	**издания**
1	Робачевский А., Немнюгин С., Стесик О.	Операционная система Unix	СПб	БХВ	2014
2	Керниган Б., Пайк Р.	UNIX. Программное окружение. – Пер. с англ. .	СПб	Символ Плюс	2003
3	Чан Т.	Системное программное обеспечение на C++ для Unix	СПб	БХВ	2007

Перечень программного обеспечения

1. Программный продукт Red Hat Enterprise Linux Server for HPC Compute Node for Power, LE, Self-support
2. Программный продукт Red Hat Enterprise Linux Server for HPC Head Node for Power, LE, Standard
3. Операционная система SUSE Linux Enterprise Server 11 SP4 for x86_64
4. Операционная система Red Hat Enterprise Linux Server 5.0 for x86_64
5. Операционная система SUSE Linux Enterprise Server 10 SP3 for ppc64
6. Операционная система Ubuntu 18.04.

Перечень профессиональных баз данных и информационных справочных систем

1. http://www.edu.ru – портал Министерства образования и науки РФ
2. http://www.ict.edu.ru – система федеральных образовательных порталов «ИКТ в образовании»
3. http://www.openet.ru - Российский портал открытого образования
4. http://www.mon.gov.ru - Министерство образования и науки Российской Федерации
5. http://www.fasi.gov.ru - Федеральное агентство по науке и инновациям

Перечень ресурсов информационно-телекоммуникационной сети «Интернет»

1. Math-Net.Ru [Электронный ресурс] : общероссийский математический портал / Математический институт им. В. А. Стеклова РАН ; Российская академия наук, Отделение математических наук. - М. : [б. и.], 2010. - Загл. с титул. экрана. - Б. ц.
URL: http://www.mathnet.ru
2. Университетская библиотека Online [Электронный ресурс] : электронная библиотечная система / ООО "Директ-Медиа" . - М. : [б. и.], 2001. - Загл. с титул. экрана. - Б. ц.
URL: www.biblioclub.ru
3. Универсальные базы данных EastView [Электронный ресурс] : информационный ресурс / EastViewInformationServices. - М. : [б. и.], 2012. - Загл. с титул. экрана. - Б. ц.
URL: www.ebiblioteka.ru
4. Научная электронная библиотека eLIBRARY.RU [Электронный ресурс] : информационный портал / ООО "РУНЭБ" ; Санкт-Петербургский государственный университет. - М. : [б. и.], 2005. - Загл. с титул. экрана. - Б. ц.
URL: www.eLibrary.ru

8. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

А. Помещения: Оборудованные лекционные аудитории.

Б. Оборудование: Ноутбук, мультимедийный проектор, экран для демонстрации решения задач в интерактивном режиме, компьютерный класс для практической апробации студентами лекционных примеров.

Автор-составитель: к.ф.-м.н., доцент факультета ВМК МГУ Вылиток А.А.