Федеральное государственное бюджетное образовательное учреждение высшего образования
«Омский государственный аграрный университет имени П.А. Столыпина»

Экономический факультет

ОПОП по направлению подготовки
09.03.02 Информационные системы и технологии

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
по дисциплине

Б1.В.13 Операционные системы

Направленность (профиль) «Информационные системы и технологии в бизнесе»

Обеспечивающая преподавание дисциплины кафедра	Кафедра экономики, бухгалтерского учета и финансового контроля
Разработчик, канд. экон. наук, доцент	А.А. Ремизова

ВВЕДЕНИЕ

1. Фонд оценочных средств по дисциплине является обязательным обособленным приложением к Рабочей программе дисциплины.

2. Фонд оценочных средств является составной частью нормативно-методического обеспечения системы оценки качества освоения обучающимися указанной дисциплины.

3. При помощи ФОС осуществляется контроль и управление процессом формирования обучающимися компетенций, из числа предусмотренных ФГОС ВО в качестве результатов освоения дисциплины.

4. Фонд оценочных средств по дисциплине включает в себя: оценочные средства, применяемые для входного контроля; оценочные средства, применяемые в рамках индивидуализации выполнения, контроля фиксированных видов ВАРС; оценочные средства, применяемые для текущего контроля и оценочные средства, применяемые при промежуточной аттестации по итогам изучения дисциплины.

5. Разработчиками фонда оценочных средств по дисциплине являются преподаватели кафедры экономики, бухгалтерского учета и финансового контроля, обеспечивающей изучение обучающимися дисциплины в университете. Содержательной основой для разработки ФОС послужила Рабочая программа дисциплины.

.

1. ОЖИДАЕМЫЕ РЕЗУЛЬТАТЫ ИЗУЧЕНИЯ
учебной дисциплины, персональный уровень достижения которых проверяется с использованием представленных в п. 3 оценочных средств

Компетенции, в формировании которых задействована дисциплина		**Код и наименование индикатора достижений компетенции**	**Компоненты компетенций,** формируемые в рамках данной дисциплины (как ожидаемый результат ее освоения)		
код	наименование		знать и понимать	уметь делать (действовать)	владеть навыками (иметь навыки)
1			2	3	4
Профессиональные компетенции					
ПК-3	Способен к администрированию процесса управления сетевых устройств и программного обеспечения, настройки политики безопасности на сетевых устройствах	ИД-2$_{ПК-3}$ Применяет методы задания базовых параметров и параметров защиты от несанкционированного доступа к операционным системам	методики заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам	применять методики заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам	владения методикой заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам
		ИД-3$_{ПК-3}$ Использует методы статической и динамической конфигурации параметров операционных систем	структуры, основных компонентов, функционирования и приемов эффективного использования операционных систем состав и принципы работы операционных систем и сред; – понятие, основные функции, типы операционных систем; обработку прерываний, планирование процессов, обслуживание ввода-вывода, управление виртуальной памятью; - принципы построения современных операционных систем и особенности их применения	проводить исследования и анализ рынка ОС, использовать основные методы работы в ОС - инсталлировать, тестировать, испытывать и использовать программно-аппаратные средства вычислительных и информационных систем	владения методами и методиками работы в операционных системах использовать средства операционных систем и сред для обеспечения работы вычислительной техники; – работать в конкретной операционной системе; – работать со стандартными программами операционной системы - навыками работы с различными операционными системами и их администрирования

ЧАСТЬ 2. ОБЩАЯ СХЕМА ОЦЕНИВАНИЯ ХОДА И РЕЗУЛЬТАТОВ ИЗУЧЕНИЯ УЧЕБНОЙ ДИСЦИПЛИНЫ

Общие критерии оценки и реестр применяемых оценочных средств

2.1 Обзорная ведомость-матрица оценивания хода и результатов изучения учебной дисциплины в рамках педагогического контроля

Категория контроля и оценки		Режим контрольно-оценочных мероприятий				
		само-оценка	взаимо-оценка	Оценка со стороны		Комис-сионная оценка
				препода-вателя	представителя производства	
		1	2	3	4	5
Входной контроль	**1**					
Индивидуализация выполнения*, **контроль фиксированных видов ВАРС:**	**2**					
- реферат	2.1			Проверка выпаленной работы (реферата)		
Текущий контроль:	**3**					
- Самостоятельное изучение тем		Вопросы изучаемой темы	Обсуждение изученных тем	опрос		
- в рамках практических (семинарских) занятий и подготовки к ним	3.1	Вопросы темы занятия	Обсуждение изученных тем на семинарах	Проверка выполненных лабораторных работ на занятиях		
- в рамках обще-университетской системы контроля успеваемости	3.2					
Рубежный контроль:	3.3			тест		
Промежуточная аттестация* обучающихся по итогам изучения дисциплины	**4**			Итоговый тест, дифференцированный зачет		
* данным знаком помечены индивидуализируемые виды учебной работы						

2.2 Общие критерии оценки хода и результатов изучения учебной дисциплины

1. Формальный критерий получения обучающимися положительной оценки по итогам изучения дисциплины:	
1.1 Предусмотренная программа изучения дисциплины обучающимся выполнена полностью до начала процесса промежуточной аттестации	1.2 По каждой из предусмотренных программой видов работ по дисциплине обучающийся успешно отчитался перед преподавателем, демонстрируя при этом должный (не ниже минимально приемлемого) уровень сформированности элементов компетенций
2. Группы неформальных критериев качественной оценки работы обучающегося в рамках изучения дисциплины:	
2.1 Критерии оценки качества хода процесса изучения обучающимся программы дисциплины (текущей успеваемости)	**2.2.** Критерии оценки качества выполнения конкретных видов ВАРС
2.3 Критерии оценки качественного уровня итоговых результатов изучения дисциплины	**2.4**. Критерии аттестационной оценки качественного уровня результатов изучения дисциплины

2.3 РЕЕСТР элементов фонда оценочных средств по учебной дисциплине

Группа оценочных средств	Оценочное средство или его элемент
	Наименование
1	2
1. Средства для входного контроля	
2. Средства для индивидуализации выполнения, контроля фиксированных видов ВАРС	Перечень тем для написания КР. Процедура выбора темы обучающимся
	Шкала и критерии оценки реферата
	Вопросы для самостоятельного изучения темы
	Общий алгоритм самостоятельного изучения темы
	Шкала и критерии оценки самостоятельного изучения темы
3. Средства для текущего контроля	Вопросы для самоподготовки по темам лабораторных занятий
	Критерии оценки самоподготовки по темам лабораторных занятий
	Примерные задания для лабораторных работ
	План лекции-беседы с основными вопросами, подлежащими рассмотрению
	Рубежный контроль
4. Средства для промежуточной аттестации по итогам изучения дисциплины	Тестовые вопросы для проведения итогового тестирования
	Шкала и критерии оценки ответов на тестовые вопросы тестирования по итогам освоения дисциплины
	Плановая процедура проведения дифференцированного зачета

2.4 Описание показателей, критериев и шкал оценивания и этапов формирования компетенций в рамках дисциплины

Индекс и название компетенции	Код индикатора достижений компетенции	Индикаторы компетенции	Показатель оценивания – знания, умения, навыки (владения)	Уровни сформированности компетенций				Формы и средства контроля формирования компетенций
				компетенция не сформирована	минимальный	средний	высокий	
				Оценки сформированности компетенций				
				2	3	4	5	
				Оценка «неудовлетворительно»	*Оценка «удовлетворительно»*	*Оценка «хорошо»*	*Оценка «отлично»*	
				Характеристика сформированности компетенции				
				Компетенция в полной мере не сформирована. Имеющихся знаний, умений и навыков недостаточно для решения практических (профессиональных) задач	Сформированность компетенции соответствует минимальным требованиям. Имеющихся знаний, умений, навыков в целом достаточно для решения практических (профессиональных) задач	Сформированность компетенции в целом соответствует требованиям. Имеющихся знаний, умений, навыков и мотивации в целом достаточно для решения стандартных практических (профессиональных) задач	Сформированность компетенции полностью соответствует требованиям. Имеющихся знаний, умений, навыков и мотивации в полной мере достаточно для решения сложных практических (профессиональных) задач	
Критерии оценивания								
ПК-3 Способен к администрированию процесса управления сетевых устройств и программного обеспечения, настройки политики безопасности и на сетевых устройствах	ИД-2$_{ПК-3}$	Полнота знаний	**методики заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам**	**Не знает методики заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам**	**Поверхностно ориентируется в методиках заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам**	**Твердо знает методики заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам**	**В совершенстве знает методики заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам**	
		Наличие умений	**применять методики заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам**	**Не умеет применять методики заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам**	**Умеет применять методики заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам**	**Умеет самостоятельно применять методики заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам**	**Умеет грамотно и самостоятельно применять методики заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным системам**	
		Наличие навыков (владение опытом)	**владения методикой заданий базовых параметров и параметров**	**Не владеет методикой заданий базовых параметров и параметров защиты от несанкционированного доступа к операционным**	**Имеет поверхностные навыки владения методикой заданий базовых параметров и параметров защиты**	**Имеет навыки владения методикой заданий базовых параметров и параметров защиты от несанкционированного**	**В совершенстве владеет навыками применения методик задания базовых параметров и параметров защиты от**	

			защиты от несанкциониро ванного доступа к операционным системам	**системам**	**от несанкционированно го доступа к операционным системам**	**доступа к операционным системам**	**несанкционированного доступа к операционным системам**	
	ИД-З$_{ПК-3}$	Полнота знаний	**структуры, основных компонентов, функционирова ния и приемов эффективного использования операционных систем состав и принципы работы операционных систем и сред; – понятие, основные функции, типы операционных систем; обработку прерываний, планирование процессов, обслуживание ввода-вывода, управление виртуальной памятью; - принципы построения современных операционных систем и особенности их применения**	**Не знает структуры, основных компонентов, функционирования и приемов эффективного использования операционных систем состав и принципы работы операционных систем и сред; – понятие, основне функции, типы операционных систем; обработку прерываний, планирование процессов, обслуживание ввода-вывода, управление виртуальной памятью; - принципы построения современных операционных систем и особенности их применения**	**Знает особенности построения и эффективного использования современных операционных систем**	**Знает особенности построения и эффективного использования современных операционных систем; принципы организации мультипрограммных режимов в условиях реального времени, разделения времени, принципы взаимодействия операционной системы и пользовательских процессов, но допускает небольшие ошибки.**	**В совершенстве знает особенности построения и эффективного использования современных операционных систем; принципы организации мультипрограммных режимов в условиях реального времени, разделения времени, принципы взаимодействия операционной системы и пользовательских процессов.**	
		Наличие умений	**проводить исследования и анализ рынка ОС, использовать основные методы работы в ОС - инсталлироват ь, тестировать, испытывать и**	**Не умеет проводить исследования и анализ рынка ОС, использовать основные методы работы в ОС - инсталлировать, тестировать, испытывать и использовать программно-аппаратные средства вычислительных и**	**Не умеет проводить исследования и анализ рынка ОС, использовать основные методы работы в ОС; инсталлировать программно-аппаратные средства вычислительных и информационных систем**	**Умеет осуществлять генерацию и реконфигурацию операционных систем,**	**Умеет грамотно проводить исследования и анализ рынка ОС, использовать основные методы работы в ОС - инсталлировать, тестировать, испытывать и использовать программно-**	

			использовать программно-аппаратные средства вычислительных и информационных систем	**информационных систем**			**аппаратные средства вычислительных и информационных систем**	
		Наличие навыков (владение опытом)	**владения методами и методиками работы в операционных системах использовать средства операционных систем и сред для обеспечения работы вычислительной техники; – работать в конкретной операционной системе; – работать со стандартными программами операционной системы - навыками работы с различными операционными системами и их администрирования**	**Не владеет методами и методиками работы в операционных системах использовать средства операционных систем и сред для обеспечения работы вычислительной техники; – работать в конкретной операционной системе; – работать со стандартными программами операционной системы - навыками работы с различными операционными системами и их администрирования**	**Имеет поверхностные навыки работы в операционных системах использовать средства операционных систем и сред для обеспечения работы вычислительной техники; – работать в конкретной операционной системе**	**Имеет основные навыки настройки и конфигурирования операционных систем с применением знаний о функционировании операционной системы.**	**В совершенстве владеет методиками работы в операционных системах использовать средства операционных систем и сред для обеспечения работы вычислительной техники; – работать в конкретной операционной системе; – работать со стандартными программами операционной системы - навыками работы с различными операционными системами и их администрирования**	

ЧАСТЬ 3 Методические материалы, определяющие процедуры оценивания знаний, умений, навыков, характеризующих этапы формирования компетенций

Часть 3.1. Типовые контрольные задания, необходимые для оценки знаний, умений, навыков

3.1.1 . Средства
для индивидуализации выполнения, контроля фиксированных видов ВАРС

Выполнение и сдача рефератов

ПРИМЕРНАЯ ТЕМАТИКА
реферата

1. Особенности построения серверных операционных систем
2. Операционные системы для мейнфреймов фирмы IBM
3. Структура и особенности построения IBM ОС Z/OS
4. Структура и особенности построения IBM ОС i5/OS
5. Структура и особенности построения IBM ОС AIX
6. Архитектура платформы IBM Virtualization Engine
7. Структура и особенности построения IBM OS/400
8. Основные производители операционных систем
9. Операционная система QNX
10. Микроядро операционной системы Mach
11. Микроядерные операционные системы
12. Основные характеристики и сравнение клиентских операционных систем
13. Кластерные операционные системы Microsoft
14. Обзор коммерческих Unix-операционных систем различных производителей
15. Обзор свободно распространяемых Unix-операционных систем различных произ-водителей
16. Обзор Linux-операционных систем различных производителей
17. Оптимизация операционной системы Windows 7
18. Реестр операционной системы Windows XP
19. Инсталляция операционной системы Windows 7
20. Установка нескольких операционных систем на ПК
21. Сравнительная характеристика операционных системы реального времени
22. Обзор стандартов, регламентирующих разработку операционных систем
23. Операционные системы многопроцессорных компьютеров
24. Виртуальные машины и их операционные системы
25. Средства виртуализации основных компаний-разработчиков операционных систем
26. Объектно-ориентированные технологии в разработке операционных систем
27. Операционные системы Интернет-серверов
28. Программные инструментальные средства анализа и оптимизации операционных систем
29. Настройка и оптимизация производительности операционных систем
30. Особенности построения сетевых операционных систем
31. Подготовка жесткого диска к установке операционной системы

Процедура выбора темы обучающимся

Очень важно правильно выбрать тему. Выбор темы не должен носить формальный характер, а иметь практическое и теоретическое обоснование.

Автор реферата должен осознанно выбрать тему с учетом его познавательных интересов. Если интересующая тема отсутствует в рекомендательном списке, то по согласованию с преподавателем обучающемуся предоставляется право самостоятельно предложить тему реферата, раскрывающую содержание изучаемой дисциплины. Тема не должна быть слишком общей и глобальной, так как небольшой объем работы (до 20 страниц) не позволит раскрыть ее.

При выборе темы необходимо учитывать полноту ее освещения в имеющейся научной литературе. Для этого можно воспользоваться тематическими каталогами библиотек и библиографическими указателями литературы, периодическими изданиями, либо справочно-библиографическими ссылками изданий посвященных данной теме.

После выбора темы составляется список изданной по теме (проблеме) литературы, опубликованных статей, необходимых справочных источников.

Знакомство с любой научной проблематикой следует начинать с освоения имеющейся основной научной литературы. При этом следует сразу же составлять библиографические выходные данные (автор, название, место и год издания, издательство, страницы) используемых источников. Названия работ иностранных авторов приводятся только на языке оригинала.

Начинать знакомство с избранной темой лучше всего с чтения обобщающих работ по данной проблеме, постепенно переходя к узкоспециальной литературе.

На основе анализа прочитанного и просмотренного материала по данной теме следует составить тезисы по основным смысловым блокам, с пометками, собственными суждениями и оценками. Предварительно подобранный в литературных источниках материал может превышать необходимый объем реферата, но его можно использовать для составления плана реферата.

ШКАЛА И КРИТЕРИИ ОЦЕНКИ
реферата

– «зачтено» Тема раскрыта. Продемонстрировано владение материалом. Использованы надлежащие источники в нужном количестве. Структура работы соответствует поставленным задачам. Степень самостоятельности работы высокая.
– «не зачтено» Тема не раскрыта. Продемонстрировано неудовлетворительное владение материалом. Использованные источники недостаточны. Структура работы не соответствует поставленным задачам. Работа несамостоятельна.

3.1.2. ВОПРОСЫ
для проведения входного контроля
не предусмотрено

3.1.3 Средства для текущего контроля

Вопросы для самостоятельного изучения тем (заочная форма обучения)
- Введение в операционные системы

ОБЩИЙ АЛГОРИТМ
самостоятельного изучения темы

1) Ознакомиться с рекомендованной учебной литературой и электронными ресурсами;
2) На этой основе составить развёрнутый план изложения темы
3) Выполнить тестовые задания или ответить на заданные вопросы.

Шкала и критерии оценки самостоятельного изучения темы

- оценка «*зачтено*» выставляется, если студент на основе самостоятельного изученного материала, смог всесторонне раскрыть теоретическое содержание темы.
- оценка «*не зачтено*» выставляется, если студент на основе самостоятельного изученного материала, не смог всесторонне раскрыть теоретическое содержание темы.

ВОПРОСЫ
для самоподготовки к лабораторным занятиям

Тема. UNIX–подобные операционные системы

1. В чем отличие дистрибутивов Linux от ядра Linux.
2. Монолитное ядро, микроядро. В чем особенности архитектуры ОС?
3. Ядро Linux, версии ядра.
4. Языки Shell, bash. Особенности использования.
5. Сценарии, для чего используются?
6. Регулярные выражения.
7. Потоки выполнения в терминале.
8. Правила использования команд в терминале.

Тема. Файловая система операционной системы

1 В чем отличие файловых систем ext2, ext3, ext4?
2. Стандарт файловой системы. Дерево файлов и каталогов Linux.

3. Создайте директории 3-й степени вложенности xxx/yyy/zzz, выведя информацию о каждом созданном каталоге. Перейдите в каталог zzz. Вернитесь в родительский каталог.
4. Перейдите в корневой каталог. Перейдите в папку /bin. Выведите на экран листинг файлов, хранящихся в папке /bin с информацией о правах доступа и пользователях с выводом каждого имени файла в отдельной строке.
5. Выведите на экран только имена каталогов текущей папки.
6. Создайте пустой файл file1.txt. Запишите в него фразу «Здравствуй мир!» и добавьте потом в конец фразу «Миру мир!» используя знак перенаправления вывода >>.
7. Измените содержимое файла file1.txt записав в начало файла фразу «Linux forever!»
8. Выведите на экран содержимое файла file1.
9. Создайте символическую и жесткую ссылки на файл file1.txt. Выведите на экран метаданные созданных файлов и исходного файла.
10. Создайте псевдоним для команды ls.
11. Скопируйте файл file1.txt в файл copy1.txt. Переименуйте файл file1.txt в файл newfile.txt.
12. Найдите в текущей папке и выведите на экран все файлы, в именах которых есть слово «file».
13. Объясните, что выполняет интерпретатор при вводе следующей команды: find /var -name "*log" -a -type f -exec cp {} test/logs/ 2>/dev/null. В каком случае такая команда не сработает?
14. Что значит выражение 2>/dev/null?
15. Создайте каталог tmptar. Создайте файлы f1, f2, f3, f4, f5, f6. Заархивируйте созданные файлы различными архиваторами, используя различные опции команды tar.
16. Разархивируйте один из созданных ранее архивов в новый каталог FILES. Удалите все файлы, в которых есть символ «f».
17. Что получим в результате выполнения команд: cat >copy1.txt и cat copy1.txt.
18. Создайте два текстовых файла first.txt и second.txt. Запишите в первый файл последовательность слов «Раз два три». Во второй файл – «четыре пять шесть». Объедините два файла в третий file.txt.
19. Откройте файл file.txt с помощью команды less. Вызовите текстовый редактор из программы less и добавьте слова «семь восемь». Выйдите из текстового редактора обратно в программу less и найдите в файле file.txt с помощью шаблона слово «семь».
20. В папке /var/log найдите системный журнал daemon.log, с помощью команды less просмотрите его и найдите в нем слово «eth0».
21. Выведите на экран последние 15 строк файла /var/log/daemon.log.
22. Посчитайте число строк в файле /var/log/daemon.log.
23. Объясните подробно, что выполнит команда: find /etc -type f -exec grep -Hn "nameserver" {} \; 2>/dev/null

Тема. Процессы и потоки

1. Реализация процессов и потоков в Unix и Linux.
2. Атрибуты процессов.
3. Жизненный цикл процессов.
4. Каким образом возникает новый процесс?
5. Определить имя текущей LINIX-системы и узнать текущие пользовательские сеансы.
6. Определите, куда смонтировано устройство sda1. Что это за устройство?
7. Что выполняет следующая команда: find / > files.txt?
8. Как определить идентификатор запускаемого процесса, например ls с помощью команды ps?
9. Запустите команду ps –aux, выберите какой-нибудь текущий процесс и выведите информацию о нем в другом терминале. Принудительно удалите этот процесс.
10. Запустите на выполнение команду top и приостановите её выполнение с помощью комбинации клавиш <Ctrl-Z>.. Выполните просмотр списка запущенных задач, возобновите выполнение команды top в фоновом режиме.
11. Придумайте команду и сразу запустите ее в фоновом режиме.
12. Найдите все процессы пользователя student. Уменьшите приоритет найденных процессов на 2 единицы, а затем увеличьте их на 2 единицы.
13. Вывести информацию о размерах и степени загруженности смонтированных дисков.
14. С помощью команды ps –aux

Тема. Сетевые операционные системы

1. Классы сетей. Перечислите классы, в чем их отличия?
2. Пакет и кадр, в чем отличия, на каких уровнях модели ISO OSI используются?
3. Основные команды маршрутизации. Какие опции, что выполняют?
4. Поясните, какую информацию выдает команда ifconfig –a.
5. Определите MAC-адрес вашей сетевой карты.
6. С помощью соответствующей команды проверьте связь с обратной петлей.

7. Какие сетевые устройства помимо eth0 и lo могуть быть определены с помощью команды ifconfig?
8. Поясните, какую информацию выдает команда route.
9. Добавьте в таблицу маршрутизации адрес 192.168.1.15.
10. Удалите добавленный адрес из таблицы маршрутизации

Тема. Безопасность операционных систем

1. Как происходит загрузка ОС Linux?
2. Загрузчики 2-го уровня: универсальные, специальные. Приведите примеры.
3. Смените пароль пользователя через универсальный загрузчик Grub
4. Задайте пароль суперпользователя в Терминале.
5. Измените пароль суперпользователя в графической оболочке.
6. Создать новую учетную запись для пользователя user1, внеся изменения в файл /etc/passwd.
7. Создать пользовательский каталог двумя способами: через файл учетных записей и с помощью команд интерпретатора.
8. Создать группу пользователей group1 двумя способами: через соответствующий файл учета групп и с помощью команд интерпретатора.
9. Вписать пользователя user1 в группу пользователей group1.
10. Измените пароль пользователя user1.
11. Создать рабочие папки для созданных групп и пользователей. Создать текстовые файлы в рабочих папках. Создать для этих файлов жесткие и символические ссылки.
12. Удалите все файлы, принадлежащие пользователю user1.
13. Удалите все файлы, принадлежащие группе group1.
14. Удалить учетную запись пользователя user1.
15. Удалить учетную запись пользователя

ШКАЛА И КРИТЕРИИ ОЦЕНКИ
самоподготовки по темам лабораторных занятий

- оценка «зачтено» выставляется обучающемуся, если все вопросы темы раскрыты, во время дискуссии высказывается собственная точка зрения на обсуждаемую проблему, демонстрируется способность аргументировать доказываемые положения и выводы.

- оценка «не зачтено» выставляется, если обучающийся не способен доказать и аргументировать собственную точку зрения по изученной теме, не способен ссылаться на мнения ведущих специалистов по обсуждаемой проблеме.

Примерные задания для лабораторных работ

Введение в операционные системы

В процессе проведения лабораторной работы студент должен выполнить следующие задания: •

1. Вывести на экран таблицу маршрутизации хоста.
2. Выяснить имя компьютера и его архитектуру.
3. Создать текстовый файл. Написать в нем три цифры 1, каждая в новой строке. Затем используя команду sed заменить цифры 1 на 3.
4. Определить шлюз по умолчанию для хоста.
5. Вывести на экран информацию о запущенных процессах в структурированном виде.

Операционные оболочки и среды

В процессе проведения лабораторной работы студент должен выполнить следующие задания:

Мониторинг основных показателей работы ОС Linux.

3.1 Ознакомиться с параметрами системы, хранящимися в виртуальной файловой системе **/proc**. Добавить в отчет информацию из **/proc/meminfo** и **/proc/cpuinfo.**

3.2 Ознакомиться с журналом ядра, выполнив команду **dmesg**. Отфильтровать только сообщения о процессоре с помощью **dmesg | grep CPU**. Полученные сообщения включить в отчет.

3.3 Ознакомиться с возможностями команды **iostat**.

3.4 Используя команду **iostat**, получить информацию о состоянии процессора и блочных устройств, включить ее в отчет.

3.5 Ознакомиться с возможностями команды **sar**.

3.6 Используя команду **sar**, выполнить мониторинг состояния памяти системы, центрального процессора, блочных устройств и сетевых интерфейсов. Мониторинг выполнить 1 раз, информацию включить в отчет.

Требования к оформлению отчета

Отчет должен содержать описание порядка выполнения всех команд и содержание указанных при выполнении работы файлов.

Архитектура операционных систем

Задание на лабораторную работу

1. Создайте свой каталог в директории /home/user/ Все скрипты создавайте внутри этого каталога или его подкаталогов. (mkdir lab1)

2. Напишите скрипты, решающие следующие задачи:

i) В параметрах скрипта передаются две строки. Вывести сообщение о равенстве или неравенстве переданных строк.

ii) В параметрах при запуске скрипта передаются три целых числа. Вывести максимальное из них. iii)Считывать строки с клавиатуры, пока не будет введена строка "q". После этого вывести последовательность считанных строк в виде одной строки.

iv) Считывать с клавиатуры целые числа, пока не будет введено четное число. После этого вывести количество считанных чисел.

v) Создать текстовое меню с четырьмя пунктами. При вводе пользователем номера пункта меню происходит запуск редактора nano, редактора vi, браузера links или выход из меню.

vi) Если скрипт запущен из домашнего директория, вывести на экран путь к домашнему директорию и выйти с кодом 0. В противном случае вывести сообщение об ошибке и выйти с кодом 1.

3. Предъявите скрипты преподавателю и получите вопрос или задание для защиты лабораторной работы.

4. После защиты лабораторной работы удалите созданный директорий совсем его содержимым:

(rm –R lab1)

Процессы и потоки

Основным интерфейсом в операционных системах GNU/Linux является консольный интерфейс с текстовым вводом и выводом данных. Это определяет подход к управлению объектами операционной системы в их текстовом отображении. Например, состояние процессов отображается в виде набора текстовых файлов в псевдофайловой системе /proc, сведения о событиях в системе хранятся в текстовых файлах журналов, настройки отдельных пакетов в текстовых конфигурационных файлах. Это делает необходимым для решения дальнейших задач управления операционной системы освоение инструментария работы с текстовыми потоками.

Управление вводом-выводом команд (процессов) У любого процесса по умолчанию всегда открыты три файла – **stdin** (стандартный ввод, клавиатура), **stdout** (стандартный вывод, экран) и **stderr** (стандартный вывод сообщений об ошибках на экран). Эти и любые другие открытые файлы могут быть перенаправлены. В данном случае термин "перенаправление" означает: получить вывод из файла (команды, программы, сценария) и передать его на вход в другой файл (команду, программу, сценарий). Дескрипторы файлов открытых по умолчанию:

0 = stdin

1 = stdout

2 = stderr

команда > файл – перенаправление стандартного вывода в файл, содержимое существующего файла удаляется.

команда >> файл – перенаправление стандартного вывода в файл, поток дописывается в конец файла.

команда1 | команда2 – перенаправление стандартного вывода первой команды на стандартный ввод второй команды = образование конвейера команд.

команда1 $(команда2) – передача вывода команды 2 в качестве параметров при запуске команды 1. Внутри скрипта конструкция

$(команда2) может использоваться, например, для передачи результатов работы команды 2 в параметры цикла **for … in.**

Работа со строками (внутренние команды bash) ${#string} – выводит длину строки (**string** – имя переменной);

${string:position:length}** – извлекает **$length символов из**$string**, начиная с позиции **$position**. Частный случай: **${string:position}** извлекает подстроку из **$string**, начиная с позиции **$position**. **${string#substring}** – удаляет самую короткую из найденных подстрок **$substring** в строке **$string**. Поиск ведется с начала строки. **$substring** – регулярное выражение.

${string##substring}** – удаляет самую длинную из найденных подстрок **$substring в строке **$string.** Поиск ведется с начала строки.

$substring – регулярное выражение.

${string/substring/replacement}** – замещает первое вхождение **$substring строкой **$replacement**. $substring – регулярное выражение.

${string//substring/replacement}** – замещает все вхождения **$substring строкой **$replacement**. **$substring** – регулярное выражение.

Работа со строками (внешние команды)

Для каждой команды доступно управление с помощью передаваемых команде параметров. Рекомендуем ознакомиться с документацией по этим командам с помощью команды man.

sort – сортирует поток текста в порядке убывания или возрастания, в зависимости от заданных опций.

uniq – удаляет повторяющиеся строки из отсортированного файла.

cut – извлекает отдельные поля из текстовых файлов (поле – последовательность символов в строке до разделителя).

head – выводит начальные строки из файла на **stdout.**

tail – выводит последние строки из файла на **stdout.**

wc – подсчитывает количество слов/строк/символов в файле или в потоке

tr – заменяет одни символы на другие.

Полнофункциональные многоцелевые утилиты:

grep – многоцелевая поисковая утилита, использующая регулярные выражения.

grep pattern [file...] – утилита поиска участков текста в файле(ах), соответствующих шаблону **pattern**, где **pattern** может быть как обычной строкой, так и регулярным выражением.

Sed – неинтерактивный "потоковый редактор". Принимает текст либо с устройства **stdin**, либо из текстового файла, выполняет некоторые операции над строками и затем выводит результат на устройство **stdout** или в файл.

Sed определяет, по заданному адресному пространству, над какими строками следует выполнить операции. Адресное пространство строк задается либо их порядковыми номерами, либо шаблоном. Например, команда **3d** заставит **sed** удалить третью строку, а команда **/windows/d** означает, что все строки, содержащие "**windows**", должны быть удалены. Наиболее часто используются команды **p** – печать (на **stdout**), **d** – удаление и **s** – замена.

awk – утилита контекстного поиска и преобразования текста, инструмент для извлечения и/или обработки полей (колонок) в структурированных текстовых файлах. **Awk** разбивает каждую строку на отдельные поля. По умолчанию поля – это последовательности символов, отделенные друг от друга пробелами, однако имеется возможность назначения других символов в качестве разделителя полей.

Awk анализирует и обрабатывает каждое поле в отдельности.

Регулярные выражения – это набор символов и/или метасимволов, которые наделены особыми свойствами. Их основное назначение – поиск текста по шаблону и работа со строками. При построении регулярных выражений используются нижеследующие конструкции (в порядке убывания приоритета), некоторые из которых могут быть использованы только в расширенных версиях соответствующих команд (например, при запуске **grep** с ключом -**E**).

c Любой неспециальный символ **c** соответствует самому себе

\c Указание убрать любое специальное значение символа **c**
(экранирование)

^ Начало строки

$** Конец строки; выражение **"^$" соответствует пустой строке.

. Любой одиночный символ, за исключением символа перевода строки

[...] Любой символ из **...**; допустимы диапазоны типа **a-z**; возможно объединение диапазонов, например **[a-z0-9]**

[^...] Любой символ не из ...; допустимы диапазоны

\n Строка, соответствующая **n**-му выражению **\(...\)**

r* Ноль или более вхождений символа **r**

r+ Одно или более вхождений символа **r**

r? Нуль или одно вхождение символа **r**

\<...\> Границы слова

\{ \} Число вхождений предыдущего выражения. Например, выражение **"[0-9]\{5\}"** соответствует подстроке из пяти десятичных цифр **r1r2** За **r1** следует **r2**

r1|r2 r1 или **r2**

(r) Регулярное выражение **r**; может быть вложенным

Классы символов POSIX

[:class:] альтернативный способ указания диапазона символов

[:alnum:] соответствует алфавитным символам и цифрам. Эквивалентно
выражению [**A-Za-z0-9**].

[:alpha:] соответствует символам алфавита. Эквивалентно выражению [**A-Za-z**]. **[:blank:]** соответствует символу пробела или символу табуляции. **[:cntrl:]** соответствует управляющим символам **[:digit:]** соответствует набору десятичных цифр. Эквивалентно выражению [**0-9**].

[:lower:] соответствует набору алфавитных символов в нижнем регистре. Эквивалентно выражению [**a-z**].

[:space:] соответствует пробельным символам (пробел и горизонтальная табуляция).

[:upper:] соответствует набору символов алфавита в верхнем регистре.

Эквивалентно выражению [**A-Z**].

[:xdigit:] соответствует набору шестнадцатиричных цифр. Эквивалентно выражению [**0-9A-Fa-f**].

Задание на лабораторную работу

1. Создайте свой каталог в директории /**home/user/** Все скрипты и файлы для вывода результатов создавайте внутри этого каталога или его подкаталогов. (**mkdir lab2**)

2. Напишите скрипты, решающие следующие задачи:

i) Создать файл **errors.log**, в который поместить все строки из всех доступных для чтения файлов директории /**var/log**/, начинающиеся с последовательности символов ACPI, без указания имени файла, в котором встретилась строка. Вывести на экран те строчки из получившегося файла, которые содержат полные имена каких-либо файлов.

ii) Создать **full.log**, в который вывести строки файла /**var/log/Xorg.0.log**, содержащие предупреждения и информационные сообщения, заменив маркеры предупреждений и информационных сообщений на слова **Warning**: и **Information:**, чтобы в получившемся файле сначала шли все информационные сообщения, а потом все предупреждения. Вывести этот файл на экран. iii) Создать файл **emails.lst**, в который вывести через запятую все адреса электронной почты, встречающиеся во всех файлах директории /**etc**.

iv) Найти в директории /**bin** все файлы, которые являются сценариями, ивывести на экран полное имя файла с интерпретатором, наиболее часто используемым в этих сценариях (только полное имя файла).

v) Вывести список пользователей системы с указанием их UID, отсортировав по UID. Сведения о пользователей хранятся в файле /**etc/passwd**. В каждой строке этого файла первое поле – имя пользователя, третье поле – UID. Разделитель – двоеточие.

vi) Подсчитать общее количество строк в файлах, находящихся в директории /**var/log**/ и имеющих расширение **log**.

vii) Вывести три наиболее часто встречающихся слова из **man** по команде **bash** длиной не менее четырех символов.

3. Предъявите скрипты преподавателю и получите вопрос или задание для защиты лабораторной работы.

4. После представления лабораторной работы удалите созданный каталог со всем его содержимым (**rm –R lab2**)

Мониторинг процессов в ОС GNU/Linux

Процесс – это совокупность набора исполняемых команд, ассоциированных с ним ресурсов и контекста выполнения управляемая операционной системы. Процесс может содержать несколько потоков исполнения. Потоки являются самостоятельными наборами исполняемых команд, но имеют доступ к общим ресурсам своего процесса. Как правило, диспетчирование операционная система выполняет именно на уровне потоков, но основной единицей управления является все же процесс.

Идентификация процессов

Система идентифицирует процессы по уникальному номеру, называемому идентификатором процесса или **PID** (process ID).

Все процессы, работающие в системе GNU/Linux, организованы в виде дерева. Корнем этого дерева является **init** – процесс системного уровня, запускаемый во время загрузки. Для каждого процесса хранится идентификатор его родительского процесса (**PPID**, Parent Process ID). У процесса **init PPID** равен 0.

Получение общих сведений о запущенных процессах

Команда **ps** (сокращение от process status)

Запуск **ps** без аргументов покажет только те процессы, которые были запущены Вами и привязаны к используемому Вами терминалу. Часто используемые параметры (указываются без "-"):

a – вывод процессов, запущенные всеми пользователями;

x – вывод процессов без управляющего терминала или с управляющим терминалом, но отличающимся от используемого Вами;

u – вывод для каждого из процессов имя запустившего его пользователя и времени запуска.

Обозначения колонок в типовом выводе команды **ps**: **PID**, **PPID** – идентификатор процесса и его родителя.

%CPU – доля процессорного времени, выделенная процессу.

%MEM – процент используемой оперативной памяти.
VSZ – виртуальный размер процесса.
TTY – управляющий терминал, из которого запущен процесс.
STAT– статус процесса:
START – время запуска процесса.
TIME – время исполнения на процессоре.
Обозначения состояний процессов (в колонке **STAT**)
R – процесс выполняется в данный момент
S – процесс ожидает (т.е. спит менее 20 секунд)
I – процесс бездействует (т.е. спит больше 20 секунд)
D – процесс ожидает ввода/вывода (или другого недолгого события), непрырываемый
Z – zombie-процесс
T – процесс остановлен
Команда **pstree**
Команда **pstree** выводит процессы в форме дерева: можно сразу увидеть родительские процессы.
Часто используемые параметры:
-p – вывод **PID** всех процессов
-u – вывод имени пользователя, запустившего процесс.
Команда **top**
top – программа, используемая для наблюдения за процессами в режиме реального времени. Полностью управляется с клавиатуры. Вы можете получить справку, нажав на клавишу **h**. Наиболее полезные команды для мониторинга процессов:
Shift+M – эта команда используется для сортировки процессов по объему занятой ими памяти (поле **%MEM**);
Shift+P – эта команда используется для сортировки процессов по занятому ими процессорному времени (поле **%CPU**). Это метод сортировки по умолчанию;
U – эта команда используется для вывода процессов заданного пользователя.
top спросит у вас его имя. Вам необходимо ввести имя пользователя, а не его UID. Если вы не введете никакого имени, будут показаны все процессы;
i – по умолчанию выводятся все процессы, даже спящие. Эта команда обеспечивает вывод информации только о работающих в данный момент процессах (процессы, у которых поле **STAT** имеет значение **R**, Running).
Повторное использование этой команды вернет Вас назад к списку всех процессов.
Получение детальных сведений о запущенных процессах
/proc – псевдо-файловая система, которая используется в качестве интерфейса к структурам данных в ядре. Большинство расположенных в ней файлов доступны только для чтения, но некоторые файлы позволяют изменять переменные ядра.
Каждому запущенному процессу соответствует подкаталог с именем, соответствующим идентификатору этого процесса (его **PID**). Каждый из этих подкаталогов содержит следующие псевдо-файлы и каталоги (указаны наиболее часто использующиеся для мониторинга процессов).
Внимание! Часть из этих файлов доступна только в директориях процессов, запущенных от имени данного пользователя или при обращении от имени **root**.
cmdline – файл, содержащий полную командную строку запуска процесса.
cwd – ссылка на текущий рабочий каталог процесса.
environ – файл, содержащий окружение процесса. Записи в файле разделяются нулевыми символами, и в конце файла также может быть нулевой символ.
exe – символьная ссылка, содержащая фактическое полное имя выполняемого файла.
fd – подкаталог, содержащий одну запись на каждый файл, который в данный момент открыт процессом. Имя каждой такой записи соответствует номеру файлового дескриптора и является символьной ссылкой на реальный файл. Так, **0** – это стандартный ввод, **1** – стандартный вывод, **2** – стандартный вывод ошибок и т. д.
maps – файл, содержащий адреса областей памяти, которые используются программой в данный момент, и права доступа к ним. Формат файла следующий:
address perms offset dev inode pathname
08048000-08056000 r-xp 00000000 03:0c 64593 /usr/sbin/gpm
08056000-08058000 rw-p 0000d000 03:0c 64593 /usr/sbin/gpm
08058000-0805b000 rwxp 00000000 00:00 0
40000000-40013000 r-xp 00000000 03:0c 4165 /lib/ld-2.2.so
bffff000-c0000000 rwxp 00000000 00:00 0
где **address** -- адресное пространство, занятое процессом; **perms** -- права доступа к нему:
r = можно читать
w = можно писать

x = можно выполнять

s = можно использовать несколькими процессами совместно

p = личная (копирование при записи);

offset -- смещение в файле, **dev --** устройство (старший номер : младший номер);

inode -- индексный дескриптор на данном устройстве: **0** означает, что с данной областью памяти не ассоциированы индексные дескрипторы;

stat – детальная информация о процессе в виде набора полей;

status – предоставляет большую часть информации из **stat** в более лёгком для прочтения формате.

sched – предоставляет информацию о процессе, использующуюся планировщиком задач.

statm – предоставляет информацию о состоянии памяти в страницах как единицах измерения. Список полей в файле:

size общий размер программы

resident размер резидентной части

share разделяемые страницы

trs текст (код)

drs данные/стек

lrs библиотека

dt "дикие" (dirty) страницы

Обработка данных о процессах

Обработка данных о процессах проводится, как правило, в рамках организации конвейера команд обработки текстовых потоков и (или) через циклическую обработку строк файлов. Советуем применять команды, изученные в рамках второй лабораторной работы – **grep, sed, awk, tr, sort, uniq, wc, paste**, а также функции для работы со строками.

Получение данных об оперативной памяти

free - возвращает информацию о свободной и используемой памяти в системе, как физической, так и виртуальной (в разделе подкачки на жестком диске).

Поля вывода команды:

total – общее количество доступной физической памяти. Некоторая область оперативной памяти может быть зарезервирована ядром, поэтому показатель total может быть меньше реального объема оперативной памяти.

used – объем используемой памяти (used=total-free).

free - свободная память.

shared - память, распределенная между процессами.

buffers - память используемая в буферах.

cached - память используемая для кэширования.

-/+ buffers/cache - использованная память без учета буферов и кэшей/свободная память с учётом буферов и КЭШей.

swap - использование раздела подкачки.

Задание на лабораторную работу

1. Создайте свой каталог в директории /**home/user/** Все скрипты и файлы для вывода результатов создавайте внутри этого каталога или его подкаталогов. (**mkdir lab3**)

2. Напишите скрипты, решающие следующие задачи:
i) Посчитать количество процессов, запущенных пользователем **user**, и вывести в файл пары

PID:команда для таких процессов.

ii) Вывести на экран **PID** процесса, запущенного последним (с последним временем запуска).

iii)Вывести в файл список **PID** всех процессов, которые были запущены командами, расположенными в **/sbin/** iv) Для каждого процесса посчитать разность резидентной и разделяемой части памяти процесса (в страницах). Вывести в файл строки вида **PID:разность**, отсортированные по убыванию этой разности.

v) Для всех зарегистрированных в данный момент в системе процессов выведите в один файл строки

ProcessID=PID : Parent_ProcessID=PPID :

Average_Time=avg_atom.

Значения **PPid** и **Pid** возьмите из файлов **status**, значение **avg_atom** из файлов **sched**, которые находятся в директориях с названиями, соответствующими **PID** процессов в **/proc** .

Отсортируйте эти строки по идентификаторам родительских процессов.

vi) В полученном на предыдущем шаге файле после каждой группы записей с одинаковым идентификатором родительского процесса вставить строку вида **Average_Sleeping_Children_of_ParentID=N is M**,

где **N** = **PPID**, а **M** – среднее, посчитанное из **SleepAVG** для данного процесса.

3. Предъявите скрипты преподавателю

4. После защиты лабораторной работы удалите созданный каталог со всем его содержимым (**rm –R lab3**)

Управление процессами в ОС GNU/Linux

Методические рекомендации:

Основными задачами управления процессами в ОС GNU/Linux является управление приоритетами процессов, планирование запуска процессов по расписанию и организация обмена данными между процессами, например с помощью сигналов. Для автоматизации управления системные администраторы создают управляющие скрипты. Последовательности команд в управляющих скриптах могут быть построены с помощью традиционных операторов процедурного программирования (условный оператор, оператор цикла), но часто используются специальные директивы объединения команд.

Директивы (команды) объединения команд

Командный интерпретатор **bash** поддерживает следующие директивы объединения команд:

команда1 | команда2 – перенаправление стандартного вывода,

команда1 ; команда2 – последовательное выполнение команд,

команда1 && команда2 – выполнение команды при успешном завершении предыдущей,

команда1 || команда2 – выполнение команды при неудачном завершении предыдущей,

команда1 $(команда2) – передача результатов работы команды 2 в качестве аргументов запуска команды 1,

команда 1 > файл – направление стандартного вывода в файл (содержимое существующего файла удаляется),

команда 1 >> файл – направление стандартного вывода в файл (поток дописывается в конец файла).

{

команда1

команда 2

} – объединение команд после директив **||** , **&&** или в теле циклов и функций.

команда1 & – запуск команды в фоновом режиме (стандартный вход и стандартный выход не связаны с консолью, из которой запускается процесс; управление процессом возможно в общем случае только с помощью сигналов).

Команды для управления процессами

(с подробным описанием возможностей и синтаксисом команд можно ознакомиться в документации, доступной по команде **man команда**)

kill – передает сигнал процессу. Сигнал может передаваться в виде его номера или символьного обозначения. По умолчанию (без указания сигнала) передает сигнал завершения процесса. Идентификация процесса для команды **kill** производится по PID. Перечень системных сигналов, доступных в GNU/Linux, с указанием их номеров и символьных обозначений можно получить с помощью команды **kill –l**;

killall – работает аналогично команде kill, но для идентификации процесса использует его символьное имя, а не PID;

pidof – определяет PID процесса по его имени;

pgrep – определяет PID процессов с заданными характеристиками (например, запущенные конкретным пользователем);

pkill – позволяет отправить сигнал группе процессов с заданными характеристиками;

nice – запускает процесс с заданным значением приоритета. Уменьшение значения (повышение приоритета выполнения) может быть инициировано только пользователем root;

renice – изменяет значения приоритета для запущенного процесса. Уменьшение значения (повышение приоритета выполнения) может быть инициировано только пользователем root;

at – осуществляет однократный отсроченный запуск команды.

cron – демон, который занимается планированием и выполнением команд, запускаемых по определенным датам и в определенное время. Команды, выполняемые периодически, указываются в файле **/etc/crontab** (не через команду **cron**, а путем внесения строк в файл **crontab** использованием одноименной команды **crontab**). Команды, которые должны быть запущены лишь однажды, добавляются при помощи **at.** Синтаксис строки в **crontab** подробно описан здесь:

Каждая команда в файле crontab занимает одну строку и состоит из шести полей:

минута час день_месяца месяц день_недели команда

Допустимые значения:

минута	**от 0 до 59**	
час	**от 0 до 23**	
день_месяца	**от 1 до 31**	
месяц	**от 1 до 12**	**(или три буквы от jan до dec, независимо от регистра)**
день_недели	**от 0 до 6**	**(0 это воскресенье или три буквы от sun до sat)**

Если в соответствующее поле поместить символ ***** это будет соответствовать любому возможному значению. Для полей можно указывать диапазоны значений, разделенных дефисом, например:

0 11 6-9 1-3 * echo "Hello World!" – вывод "Hello World!" в 11:00 в 6,7,8,9 дни января, февраля и марта.

0 */2 * * mon echo "Hello World!" – вывод "Hello World!" каждый четный час каждого понедельника

tail – не только выводит последние n строк из файла, но и позволяет организовать "слежение" за файлом – обнаруживать и выводить новые строки, появляющиеся в конце файла.

sleep– задает паузу в выполнении скрипта.

Организация взаимодействия двух процессов

Существует несколько вариантов организации взаимодействия процессов. Поскольку суть взаимодействия состоит в передаче данных и/или управления от одного процесса к другому, рассмотрим два распространенных варианта организации такого взаимодействия: передачу данных через файл и передачу управления через сигнал.

Взаимодействие процессов через файл

Для демонстрации передачи информации через файл рассмотрим два скрипта – «Генератор» и «Обработчик». Требуется считывать информацию с консоли с помощью процесса «Генератор» и выводить ее на экран с помощью процесса «Обработчик», причем таким образом, чтобы считывание генератором строки «QUIT» приводило к завершению работы обработчика. Каждый скрипт запускается в своей виртуальной консоли. Переключаясь между консолями, можно управлять скриптами и наблюдать результаты их работы.

Генератор

```
#!/bin/bash
while true; do
        read LINE
        echo $LINE >> data.txt
done
```

Обработчик

```
#!/bin/bash
(tail -n 0 -f data.txt) |
while true; do
        read LINE;
        case $LINE in
                QUIT)
                        echo
"exit"
                        killall
tail
                        exit
                        ;;
                *)
                        echo
$LINE
                        ;;
        esac
done
```

Скрипт «Генератор» в бесконечном цикле считывает строки с консоли и дописывает их в конец файла data.txt. Скрипт «Обработчик» рассмотрим подробнее. Команда **tail** позволяет считывать последние **n** строк из файла. Но один из наиболее распространенных вариантов ее использования – организация «слежения» за файлом. При использовании конструкции **tail -f** считывание из файла будет происходить только в случае добавления информации в этот файл. При этом ключ **-n 0** предотвращает чтение из файла, пока его содержимое не обновилось после запуска команды **tail.** Поскольку необходимо передавать выход команды **tail** на вход скрипта «Обработчик», используем конструкцию **(команды)|** Круглые скобки позволяют запустить независимый подпроцесс (дочерний процесс) внутри родительского процесса «Обработчик», а оператор конвейера в конце позволит направить выход этого подпроцесса на вход родительского процесса. Таким образом, команда **read** в этом скрипте читает выход команды **tail.** Остальная часть скрипта основывается на конструкциях, изученных в предыдущих лабораторных работах, и не требует детального рассмотрения. Исключение составляет только команда **killall tail.** С ее помощью завершается вызванный в подпроцессе процесс **tail** перед завершением родительского процесса. Использование **killall** в этом случае используется для упрощения кода, но не всегда является корректным. Лучше определять PID конкретного процесса.

Взаимодействие процессов с помощью сигналов Сигналы являются основной формой передачи управления от одного процесса к другому. Существуют «стандартные» (системные) сигналы, имеющие фиксированные имена и названия (например, SIGTERM, SIGKILL и т.д.), но существует возможность передавать процессу и вновь создаваемый, пользовательский сигнал. Таблица 1. Часто используемые сигналы

№	Имя	Описание	Можно перехватывать	Можно блокировать	Комбинация клавиш
1	HUP	Hangup. Отбой. Получение этого сигнала как правило означает, что завершил работу терминал из которого был запущен процесс и следовательно процесс тоже должен быть завершен.	Да	Да	
2	INT	Interrupt. В случае выполнения простых команд вызывает прекращение выполнения, в интерактивных программах - прекращение активного процесса	Да	Да	<Ctrl>+<C> или <Del>
3	QUIT	Как правило, сильнее сигнала Interrupt	Да	Да	<Ctrl>+<\>

4	ILL	Illegal Instruction. Центральный процессор столкнулся с незнакомой командой (в большинстве случаев это означает, что допущена программная ошибка). Сигнал отправляется программе, в которой возникла проблема	Да	Да	
8	FPE	Floating Point Exception. Вычислительная ошибка, например, деление на ноль	Да	Да	
9	KILL	Всегда прекращает выполнение процесса	Нет	Нет	
11	SEGV	Segmentation Violation. Доступ к недозволенной области памяти	Да	Да	
13	PIPE	Была предпринята попытка передачи данных с помощью конвейера или очереди FIFO, однако не существует процесса, способного принять эти данные	Да	Да	
15	TERM	Software Termination. Требование закончить процесс (программное завершение)	Да	Да	
17	CHLD	Изменение статуса порожденного процесса	Да	Да	
18	CONT	Продолжение выполнения приостановленного процесса	Да	Да	
19	STOP	Приостановка выполнения процесса	Нет	Нет	
20	TSTR	Сигнал останова, генерируемый клавиатурой. Переводит процесс в фоновый режим	Да	Да	<Ctrl>+<Z>

С помощью команды trap можно не только задать обработчик для пользовательского сигнала, но и подменить обработчик для некоторых из системных сигналов (кроме тех, перехват которых запрещен). В этом случае обработка сигнала перейдет к указанному в trap обработчику. Для демонстрации передачи управления от одного процесса к другому рассмотрим еще одну пару скриптов.

Генератор

```
#!/bin/bash
while true; do
        read LINE
        case $LINE in
                STOP)
                    kill -USR1
$(cat .pid)
                        ;;
                *)
                        :
                        ;;
        esac
done
```

Обработчик

```
#!/bin/bash
echo $$ > .pid
A=1
MODE="rabota"
usr1()
{
     MODE="ostanov"
}
trap 'usr1' USR1
while true; do
     case $MODE in
          "rabota")
               let A=$A+1
               echo $A
               ;;
          "ostanov")
               echo "Stopped by
SIGUSR1"
               exit
               ;;
     esac
     sleep 1
done
```

В этом случае скрипт «Генератор» будет в бесконечном цикле считывать строки с консоли и бездействовать (используется оператор **:**) для любой входной строки, кроме строки STOP, получив которую, он отправит пользовательский сигнал **USR1** процессу «Обработчик». Поскольку процесс «Генератор» должен знать PID процесса «Обработчик», передача этого идентификационного номера осуществляется через скрытый файл. В процессе «Обработчик» определение PID процесса производится с помощью системной переменной **$$.** Процесс «Обработчик» выводит на экран последовательность

натуральных чисел до момента получения сигнала **USR1**

запускается обработчик **usr1()**, который меняет значение переменной MODE. В результате на следующем шаге цикла будет выведено сообщение о прекращении работы в связи с появлением сигнала, и работа скрипта будет завершена.

Задание на лабораторную работу

Создайте скрипты или запишите последовательности выполнения команд для перечисленных заданий и предъявите их преподавателю. 1. Создайте и однократно выполните скрипт (в этом скрипте нельзя использовать условный оператор и операторы проверки свойств и значений), который будет пытаться создать директорию **test** в домашней директории. Если создание директории пройдет успешно, скрипт выведет в файл **~/report** сообщение вида "**catalog test was created successfully**" и создаст в директории **test** файл с именем **Дата_Время_Запуска_Скрипта**. Затем независимо от результатов предыдущего шага скрипт должен опросить с помощью команды **ping** хост www.net_nikogo.ru и, если этот хост недоступен, дописать сообщение об ошибке в файл **~/report.**

2. Задайте еще один однократный запуск скрипта из пункта 1 через 2 минуты. Организуйте слежение за файлом **~/report** и выведите на консоль новые строки из этого файла, как только они появятся. 3. Задайте запуск скрипта из пункта 1 каждые 5 минут каждого часа в день недели, в который вы будете выполнять работу.

4. Создайте два фоновых процесса, выполняющих одинаковый бесконечный цикл вычисления (например, перемножение двух чисел). После запуска процессов должна сохраниться возможность использовать виртуальные консоли, с которых их запустили. Используя команду **top**, проанализируйте процент использования ресурсов процессора этими процессами. Добейтесь, чтобы тот процесс, который был запущен первым, использовал ресурс процессора не более чем на 20%.

5. Процесс «Генератор» передает информацию процессу «Обработчик» с помощью файла. Процесс «Обработчик» должен осуществлять следующую обработку новых строк в этом файле: если строка содержит единственный символ «**+**», то процесс «Обработчик» переключает режим на *сложение* и ждет ввода численных данных. Если строка содержит единственный символ «

предыдущего вычисления со считанным числом). При запуске скрипта режим устанавливается в сложение, а вычисляемая переменная приравнивается к 1. В случае получения строки **QUIT** скрипт выдает сообщение о плановой остановке и завершает работу. В случае получения любых других значений строки скрипт завершает работу с сообщением об ошибке входных данных.

6. Процесс «Генератор» считывает строки с консоли, пока ему на вход не поступит строка **TERM**. В этом случае он посылает системный сигнал **SIGTERM** процессу обработчику. Процесс «Обработчик» (как и в примере, выводящий в бесконечном цикле натуральное число каждую секунду) должен перехватить системный сигнал **SIGTERM** и завершить работу, предварительно выведя сообщение о завершении работы по сигналу от другого процесса.

7. Процесс «Генератор» считывает с консоли строки в бесконечном цикле. Если считанная строка содержит единственный символ «**+**», он посылает процессу «Обработчик» сигнал **USR1**. Если строка содержит единственный символ «*****», генератор посылает обработчику сигнал **USR2**. Если строка содержит слово **TERM**, генератор посылает обработчику сигнал **SIGTERM**. Другие значения входных строк игнорируются. Обработчик добавляет 2 или умножает на 2 текущее значение обрабатываемого числа (начальное значение принять на единицу) в зависимости от полученного пользовательского сигнала и выводит результат на экран. Вычисление и вывод производятся один раз

Управление памятью

Задание на лабораторную работу

1. Опишите все известные Вам виды компьютерной памяти.
2. Найдите и укажите объем оперативной памяти на Вашем рабочем ПК. Укажите при помощи скриншота этому подтверждение. Также укажите в отчете путь, где расположена эта информация.
3. Укажите объем жесткого диска и каждого имеющегося раздела. Сколько свободного и занятого пространства на винчестере. Подтвердите это скриншотом.
4. Найдите версию БИОС при помощи одной из служебных программ. Напишите ее в отчет.
5. Укажите, сколько используется оперативной памяти в данный момент. Подтвердите это скриншотом. Укажите скриншот диаграммы памяти из «монитора ресурсов».
6. Найдите самостоятельно и измените размер и месторасположение файла подкачки. Укажите в отчете полный путь и скриншот окна измененных настроек.

Файловая система ОС

Задание на лабораторную работу

Создайте скрипты для перечисленных заданий и предъявите их преподавателю.

1. Скрипт **rmtrash** a. Скрипту передается один параметр – имя файла в текущем каталоге вызова скрипта.

b. Скрипт проверяет, создан ли скрытый каталог **trash** в домашнем каталоге пользователя. Если он не создан – создает его.

c. После этого скрипт создает в этом каталоге жесткую ссылку на переданный файл с уникальным именем (например, присваивает каждой новой ссылке имя, соответствующее следующему натуральному числу) и удаляет файл в текущем каталоге.

d. Затем в скрытый файл **trash.log** в домашнем каталоге пользователя помещается запись, содержащая полный исходный путь к удаленному файлу и имя созданной жесткой ссылки.

2. Скрипт **untrash**

a. Скрипту передается один параметр – имя файла, который нужно восстановить (без полного пути – только имя).

b. Скрипт по файлу **trash.log** должен найти все записи, содержащие в качестве имени файла переданный параметр, и выводить по одному на экран полные имена таких файлов с запросом подтверждения.

c. Если пользователь отвечает на подтверждение положительно, то предпринимается попытка восстановить файл по указанному полному пути (создать в соответствующем каталоге жесткую ссылку на файл из **trash** и удалить соответствующий файл из **trash**). Если каталога, указанного в полном пути к файлу, уже не существует, то файл восстанавливается в домашний каталог пользователя с выводом соответствующего сообщения.

3. Скрипт **backup**

a. Скрипт создаст в **/home/user/** каталог с именем **Backup-YYYYMM-DD**, где YYYY-MM-DD – дата запуска скрипта, если в **/home/user/** нет каталога с именем, соответствующим дате, отстоящей от текущей менее чем на 7 дней. Если в **/home/user/** уже есть «действующий» каталог резервного копирования (созданный не ранее 7 дней от даты запуска скрипта), то новый каталог не создается. Для определения текущей даты можно воспользоваться командой **date**.

b. Если новый каталог был создан, то скрипт скопирует в этот каталог все файлы из каталога **/home/user/source/** (для тестирования скрипта создайте такую директорию и набор файлов в ней). После этого скрипт выведет в режиме дополнения в файл **/home/user/backup-report** следующую информацию: строка со сведениями о создании нового каталога с резервными копиями с указанием его имени и даты создания; список файлов из **/home/user/source/**, которые были скопированы в этот каталог.

c. Если каталог не был создан (есть «действующий» каталог резервного копирования), то скрипт должен скопировать в него все файлы из **/home/user/source/** по следующим правилам: если файла с таким именем в каталоге резервного копирования нет, то он копируется из **/home/user/source**. Если файл с таким именем есть, то его размер сравнивается с размером одноименного файла в действующем каталоге резервного копирования. Если размеры совпадают, файл не копируется. Если размеры отличаются, то файл копируется с автоматическим созданием версионной копии, таким образом, в действующем каталоге резервного копирования появляются обе версии файла (уже имеющийся файл переименовывается путем добавления дополнительного расширения «**.YYYY-MM-DD**» (дата запуска скрипта), а скопированный сохраняет имя). После окончания копирования в файл **/home/user/backup-report** выводится строка о внесении изменений в действующий каталог резервного

копирования с указанием его имени и даты внесения изменений, затем строки, содержащие имена добавленных файлов с новыми именами, а затем строки с именами добавленных файлов с существовавшими в действующем каталоге резервного копирования именами с указанием через пробел нового имени, присвоенного предыдущей версии этого файла.

4. Скрипт **upback**

a. Скрипт должен скопировать в каталог **/home/user/restore/** все файлы из актуального на данный момент каталога резервного копирования (имеющего в имени наиболее свежую дату), за исключением файлов с предыдущими версиями.

5. Все скрипты должны корректно обрабатывать любые передаваемые им входные параметры и значения. Не допускается вывод сообщений об ошибках от отдельных команд, использующихся в скрипте. В случае некорректных входных данных или невозможности выполнить операцию, пользователю должно выводиться отдельное сообщение, формирующееся в скрипте. Перед предъявление результатов выполнения лабораторной работы преподавателю необходимо провести тестирование разработанных скриптов.

Безопасность операционных систем

Порядок выполнения работы

1. Определить для обычного пользователя возможность для запуска команды **tcpdump** через команду **sudo**.

2. Установить пароль на загрузчик операционной системы.

3. Отредактировать существующую политику для SELinux с сервисом **Samba** таким образом, чтобы можно было настроить работу с разделяемым ресурсом, находящимся в произвольном месте файловой системы.

4. Настроить программу **logwatch** на отсылку оповещений по почте о неудачных попытках входа в систему.

5. Настроить ограничения для работы программы ssh путем редактирования файла конфигурации. Запретить удаленный доступ к системе суперпользователю, изменить порт для подключения с 22 на иной (например 6622),

6. Настроить аутентификацию программы SSH по ключевой паре вместо паролей.

Требования к оформлению отчета

Отчет должен содержать описание порядка выполнения всех команд и вывод, полученный при их исполнении.

План лекции-беседы с основными вопросами, подлежащими рассмотрению

Тема: *Введение в операционные системы*

План

1. Предмет и содержание курса, взаимосвязь курса со смежными дисциплинами..
2. Назначение и функции операционных систем
3. История развития и поколения операционных систем (ОС).
4. Функциональные компоненты ОС.
5. Операционные системы универсального и специального назначения

Вопросы, подлежащие рассмотрению

1. Основные функции операционной системы
2. Монолитный подход к построению ядра операционной системы
3. Микроядерный подход к построению ядра операционной системы
4. Что понимается под системным вызовом?
5. Для чего применяются системные вызовы?
6. В чем отличие между пространством ядра и пространством пользователя?

Тема: *Операционные оболочки и среды*

План

1. Назначение и основные функции.
2. Графические оболочки.
3. Командные интерпретаторы.
4. Операционные среды

Вопросы, подлежащие рассмотрению

1. Что понимается под операционной системой?
2. Основные функции операционных систем
3. Что понимается под термином «Драйвер устройства»?
4. Типы информационных структур
5. Этапы загрузки операционных систем

3.1.3 Рубежный контроль

Рубежное тестирование по темам 1-4

1) Операционная система (ОС) – это система…

a) программ, осуществляющая связь между процессором и периферийным устройством;

b) средств, методов и персонала, используемая для сохранения, обработки и выдачи информации с целью решения конкретной задачи.

c) программ, реализующая интерфейс между аппаратурой ЭВМ и пользователями;

d) средств, которые фиксируют результаты определенных операций, свойства объектов и субъектов управления, параметры процессов, содержание нормативных и юридических актов и т.п.

2) Функции операционной системы:

a) обеспечение пользователей программами доступа к различным устройствам ввода/вывода

b) управление распределением ресурсов вычислительной системы для обеспечения ее эффективной работы

c) обеспечение пользователей набором средств для облегчения проектирования, программирования

d) управление распределением долговременной памяти для обеспечения эффективной работы устройства

3) Основные подсистемы управления ресурсами – это подсистемы:

a) управления процессами

b) управление сетевыми дисками

c) управления памятью

d) управления файлами и внешними устройствами

4) Интерфейсом прикладного программирования, называется…

a) средства, предоставляемые операционной системой для обращения к ее возможностям при написании приложений.

b) программа, обеспечивающая оптимальное использование ресурсов вычислительной машины в режиме многозадачности.

c) свойство программы, позволяющее одновременно выполнять эту программу нескольким процессам

d) последовательность операций при выполнении программы или ее части в совокупности с используемыми данными.

5) Что не является классификацией ОС?

a) типы аппаратной платформы;

b) загрузка пользовательских программ в оперативную память

c) реализация внутренних алгоритмов управления ресурсами.

d) методы проектирования

6) ОС Windows поддерживают следующие типы разделов

a) Основной

b) Базовый

c) Подкачки

d) Дополнительный

7) Сопоставьте понятия классификация по области использования:

настольные ОС	ориентированные на решение узких классов задач с жестким набором требований
серверные ОС	использующиеся в серверах сетей как центральное звено, а также в качестве элементов систем управления
мобильные ОС	ориентированные на работу отдельного пользователя в различных предметных областях
специализированные ОС	вариант развития настольных ОС на аппаратной платформе КПК

8) В какой период времени в технической базе произошел переход от отдельных полупроводнико- вых элементов типа транзисторов к интегральным микросхемам?

a) 1945 – 1955 гг.

b) 1955 – 1965 гг.

c) 1965-1980 гг.

d) 1980 – наше время

9) Потребность в синхронизации возникает в:

a. однопрограммных ОС

b. многопрограммных ОС

c. многозадачных ОС

d. однозадачных ОС

10) Программы ОС группируются согласно выполняемым функциям и называются…

a) подсистемами ОС

b) системами ОС

c) процессами ОС

d) данные ОС

Рубежное тестирование по темам 5-8

1) Виртуальная память –

a) картина памяти, не опирающаяся на операционную систему для выполнения процесса
b) картина памяти, формируемая операционной системой для процесса
c) картина памяти, предоставляемый процесс посредством механизма
d) картина памяти, предоставляемый процесс посредством механизма операционной системы

2) Объем страницы:
a) выбирается по возможности максимальный
b) для процессоров Intel стандартно равен 4 Кбайта
c) выбирается минимальным

3) Распределение памяти без использования внешней памяти является
a) фиксированными разделами
b) тактическими разделами
c) динамическими разделами
d) перемещаемыми разделами

4) Распределение памяти с использованием внешней памяти:
a) стратегическое разделение
b) страничное распределение
c) сегментное распределение
d) сегментно-страничное распределении

5) Специальная информационная структура, описатель процесса
a) ресурс процесса
b) идентификатор процесса
c) дескриптор процесса
d) утилита процесса
e) прерывания процесса

6) Образ процесса –
a) термин, обозначающий содержимое назначенного процессу виртуального адресного пространства, т.е. коды команд и данные
b) термин, обозначающий изменение назначенного процессу виртуального адресного пространства, т.е. изменение кодов команд и данных
c) термин, обозначающий удаление назначенного процессу виртуального адресного пространства, т.е. удаление кодов команд и данных
d) термин, обозначающий копирование назначенного процессу виртуального адресного пространства, т.е.копирование кодов команд и данных

7) … - это виртуальное адресное пространство представлено в виде непрерывной линейной последовательности адресов. Линейный виртуальный адрес – число, представляющее собой смещение относительно начала виртуального адресного пространства. (Плоская структура.)

8) Модель памяти «сегмент-смещение» была реализована:
a) в 32-разрядной архитектуре IBM-360
b) в 32-разрядной архитектуре x-86
c) в 16-разрядной архитектуре x-86
d) в 16-разрядной архитектуре IBM-360

9) Для процессов в Windows NT
a) память представляется плоской и имеет объем 2 Гб
b) память представляется плоской и имеет объем 4 Гб
c) память представляется плоской и имеет объем 16 Гб
d) память представляется плоской и имеет объем 32 Гб

10) MSDOS1 — это
a) многопрограммная операционная система для персонального компьютера типа IBMPC
b) однопрограммная операционная система для персонального компьютера типа AT386
c) многопрограммная операционная система для персонального компьютера типа AT386
d) однопрограммная операционная система для персонального компьютера типа IBMPC

11) Командный процессор является файлом?
a) файл COMMAND.COM
b) файл CONFIG.SYS
c) файл MSDOS.SYS
d) файл CONFIG.COM

12) Алгоритмы какого класса предполагают, что размер виртуального адресного пространства каждого процесса меньше объема оперативной памяти?

a) Первого класса
b) Второго класса
c) Третьего класса
d) Четвертого класса

13) Средства предоставления собственных ресурсов и услуг в общее пользование - это

a) Ресурсы
b) Запросы
c) Услуги
d) Сервер

14) Клиентская часть ОС является?

a) Редиректор
b) Сервер
c) Файловая система
d) Каталог

15) Сети отделов –

a) используются большой группой сотрудников, решающих общие задачи
b) используются небольшой группой людей, решающих разные задачи
c) используются небольшой группой сотрудников, решающих общие задачи
d) используются большой группой людей, решающих разные задачи

16) Сети кампусов –

a) соединяют несколько компьютеров внутри отдельного здания или внутри одной территории предприятия
b) соединяют несколько сетей территории жилого комплекса
c) соединяют несколько сетей отделов небольших групп людей, решающих разные задачи
d) соединяют несколько сетей отделов внутри отдельного здания или внутри одной территории предприятия

17) При использовании не блокирующего примитива управление возвращается вызывающему процессу немедленно, еще до того, как требуемая работа будет

a) выполнена
b) удалена
c) редактирована
d) сохранена

18) Какой подход заключается в использовании ответа в качестве подтверждения в тех системах, в которых запрос всегда сопровождается ответом?

a) Первый
b) Второй
c) Третий

19) Идея вызова удаленных процедур состоит

a) В расширении не известного механизма передачи управления и данных программы, выполняющейся на одной машине, на передачу управления и данных через сеть
b) В расширении передачи управления данных программы, выполняющейся на разных машинах, на передачу управления и данных через сеть
c) В расширении хорошо известного и понятного механизма передачи управления и данных внутри программы, выполняющейся на нескольких машинах, на передачу управления и данных через сеть
d) В расширении хорошо известного и понятного механизма передачи управления и данных внутри программы, выполняющейся на одной машине, на передачу управления и данных через сеть

20) Характерными чертами вызова локальных процедур являются:

a) Асимметричность
b) Локальность
c) Удаленность
d) Синхронность

21) …. — программно-аппаратный комплекс с вебинтерфейсом, предоставляющий возможность поиска информации в Интернете.

*Поисковая система

3.1.4. Средства для промежуточной аттестации по итогам изучения дисциплины

Заключительное тестирование по итогам изучения дисциплины

По итогам изучения дисциплины, обучающиеся проходят заключительное тестирование. Тестирование является формой контроля, направленной на проверку владения терминологическим аппаратом, современными информационными технологиями и конкретными знаниями в области фундаментальных и прикладных дисциплин.

Подготовка к заключительному тестированию по итогам изучения дисциплины

Тестирование осуществляется по всем темам и разделам дисциплины, включая темы, выносимые на самостоятельное изучение.

Процедура тестирования ограничена во времени и предполагает максимальное сосредоточение обучающегося на выполнении теста, содержащего несколько тестовых заданий.

Тестирование проводится в письменной форме (на бумажном носителе). Тест включает в себя 30 вопросов. Время, отводимое на выполнение теста - 30 минут. В каждый вариант теста включаются вопросы в следующем соотношении: закрытые (одиночный выбор) – 25-30%, закрытые (множественный выбор) – 25-30%, открытые – 25-30%, на упорядочение и соответствие – 5-10%

На тестирование выносится по 5 вопросов из каждого раздела дисциплины.

Бланк теста

Образец

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Омский государственный аграрный университет имени П.А. Столыпина»

Тестирование по итогам освоения дисциплины «Операционные системы»
Для обучающихся направления подготовки 09.03.02 Информационные системы и технологии
ФИО_____группа_____
Дата_____

Уважаемые обучающиеся!

Прежде чем приступить к выполнению заданий внимательно ознакомьтесь с инструкцией:

1. Отвечая на вопрос с выбором правильного ответа, правильный, на ваш взгляд, ответ (ответы) обведите в кружок.
2. В заданиях открытой формы впишите ответ в пропуск.
3. В заданиях на соответствие заполните таблицу.
4. В заданиях на правильную последовательность впишите порядковый номер в квадрат.
4. Время на выполнение теста – 30 минут
5. За каждый верный ответ Вы получаете 1 балл, за неверный – 0 баллов.

Желаем удачи!

Вариант № 1

1) Модель памяти «сегмент-смещение» была реализована:
a) в 32-разрядной архитектуре IBM-360
b) в 32-разрядной архитектуре x-86
c) в 16-разрядной архитектуре x-86
d) в 16-разрядной архитектуре IBM-360

2) Для процессов в Windows NT
a) память представляется плоской и имеет объем 2 Гб
b) память представляется плоской и имеет объем 4 Гб
c) память представляется плоской и имеет объем 16 Гб
d) память представляется плоской и имеет объем 32 Гб

3) MSDOS1 — это
a) многопрограммная операционная система для персонального компьютера типа IBMPC

b) однопрограммная операционная система для персонального компьютера типа AT386
c) многопрограммная операционная система для персонального компьютера типа AT386
d) однопрограммная операционная система для персонального компьютера типа IBMPC

4) При использовании не блокирующего примитива управление возвращается вызывающему процессунемедленно, еще до того, как требуемая работа будет
a) выполнена
b) удалена
c) редактирована
d) сохранена

5) Какой подход заключается в использовании ответа в качестве подтверждения в тех системах, в которых запрос всегда сопровождается ответом?
a) Первый
b) Второй
c) Третий

6) Потребность в синхронизации возникает в:
a. однопрограммных ОС
b. многопрограммных ОС
c. многозадачных ОС
d. однозадачных ОС

7) Программы ОС группируются согласно выполняемым функциям и называются…
a) подсистемами ОС
b) системами ОС
c) процессами ОС
d) данные ОС

8) Специальная информационная структура, описатель процесса
a) ресурс процесса
b) идентификатор процесса
c) дескриптор процесса
d) утилита процесса
e) прерывания процесса

9) Образ процесса –
a) термин, обозначающий содержимое назначенного процессу виртуального адресного пространства, т.е. коды команд и данные
b) термин, обозначающий изменение назначенного процессу виртуального адресного пространства, т.е. изменение кодов команд и данных
c) термин, обозначающий удаление назначенного процессу виртуального адресного пространства, т.е. удаление кодов команд и данных
d) термин, обозначающий копирование назначенного процессу виртуального адресного пространства, т.е.копирование кодов команд и данных

10) … - это виртуальное адресное пространство представлено в виде непрерывной линейной последовательности адресов. Линейный виртуальный адрес – число, представляющее собой смещение относительно начала виртуального адресного пространства.
*Плоская структура

11) Модель памяти «сегмент-смещение» была реализована:
a) в 32-разрядной архитектуре IBM-360
b) в 32-разрядной архитектуре x-86
c) в 16-разрядной архитектуре x-86
d) в 16-разрядной архитектуре IBM-360
9) Для процессов в Windows NT
a) память представляется плоской и имеет объем 2 Гб
b) память представляется плоской и имеет объем 4 Гб
c) память представляется плоской и имеет объем 16 Гб
d) память представляется плоской и имеет объем 32 Гб

12) MSDOS1 — это

a) многопрограммная операционная система для персонального компьютера типа IBMPC
b) однопрограммная операционная система для персонального компьютера типа AT386
c) многопрограммная операционная система для персонального компьютера типа AT386
d) однопрограммная операционная система для персонального компьютера типа IBMPC

13) Командный процессор является файлом?
a) файл COMMAND.COM
b) файл CONFIG.SYS

14) Средства предоставления собственных ресурсов и услуг в общее пользование - это
a) ресурсы
b) запросы
c) услуги
d) сервер

15) Клиентская часть ОС является?
a) редиректор
b) сервер
c) файловая система
d) каталог

16) В ОС на основе микроядра при обращении к функции ядра, смена режимов происходит … раза
1) 1
2) 4
3) 5
4) 3
5) 2

17) Некоторое число (номер) в диапазоне 0-255, указывающее на одну из 256 программ обработки прерываний, адреса которых хранятся в таблице прерываний, называется … прерывания (ий)
1) адресом
2) вектором
3) адресом обработчика
4) номером
5) номером обработчика

18) Способ реализации системных вызовов зависит от структурной организации ОС, связанной с особенностями:
1) оперативной памяти
2) внешней памяти
3) обработки прерываний
4) приоритетного обслуживания
5) аппаратной платформы

19) Выберите верные утверждения:
1) Совместимость на уровне исходных текстов требует наличия соответствующего компилятора на вычислительной машине, на которой планируют выполнять данное приложение, а также совместимости на уровне системных вызовов
2)совместимость на уровне исходных текстов требует наличия соответствующего компилятора на вычислительной машине, на которой планируют выполнять данное приложение, а также совместимости на уровне системных вызовов и идентичности внутренней структуры исполняемого файла приложения
3)совместимость на уровне исходных текстов требует наличия соответствующего компилятора на вычислительной машине, на которой планируют выполнять данное приложение, а также идентичности внутренней структуры исполняемого файла приложения, в тоже время совместимость на уровне системных вызовов не является обязательной
4) совместимость на уровне исходных текстов требует наличия соответствующего компилятора на вычислительной машине, на которой

ШКАЛА И КРИТЕРИИ ОЦЕНИВАНИЯ

ответов на тестовые вопросы тестирования по итогам освоения дисциплины

- оценка «отлично» выставляется обучающемуся, если получено более 81% правильных ответов.

- оценка «хорошо» - получено от 71 до 80% правильных ответов.
- оценка «удовлетворительно» - получено от 61 до 70% правильных ответов.
- оценка «неудовлетворительно» - получено менее 61% правильных ответов.

ПЛАНОВАЯ ПРОЦЕДУРА
дифференцированного зачета

1) обучающийся выполнил все виды учебной работы (включая самостоятельную) и отчитался об их выполнении в сроки, установленные графиком учебного процесса по дисциплине;
2) прошёл заключительное тестирование

Нормативная база проведения промежуточной аттестации обучающихся по результатам изучения дисциплины:	
Действующее «Положение о текущем контроле успеваемости, промежуточной аттестации обучающихся по программам высшего образования (бакалавриат, специалитет, магистратура) и среднего профессионального образования в ФГБОУ ВО Омский ГАУ»	
Основные характеристики промежуточной аттестации обучающихся по итогам изучения дисциплины	
Цель промежуточной аттестации -	Установление уровня достижения каждым обучающимся целей и задач обучения по данной дисциплине, изложенным в п.2.2 настоящей программы
Форма промежуточной аттестации -	Дифференцированный зачет
Место процедуры получения зачёта в графике учебного процесса	1. Участие обучающегося в процедуре получения зачёта осуществляется за счёт учебного времени (трудоёмкости), отведённого на изучение дисциплины
	2. Процедура проводится в рамках ВАРО, на последней неделе семестра
Основные условия получения обучающимся зачёта:	1) обучающийся выполнил все виды учебной работы (включая самостоятельную) и отчитался об их выполнении в сроки, установленные графиком учебного процесса по дисциплине; 2) прошёл заключительное тестирование

ЛИСТ РАССМОТРЕНИЙ И ОДОБРЕНИЙ

Фонд оценочных средств учебной дисциплины Б1.В.13 Операционные системы в составе ОПОП 09.03.02 Информационные системы и технологии

1. Рассмотрен и одобрен в качестве базового варианта:

а) На заседании обеспечивающей кафедры экономики, бухгалтерского учета и финансового контроля

протокол № 11 от 19 .05.2022.

Зав. кафедрой, канд. экон. наук, доцент _____ О.А. Блинов

б) На заседании методической комиссии по направлению 09.03.02 Информационные системы и технологии

протокол № 9 от 24 .05.2022.

Председатель МКН – 09.03.02, канд. экон. наук _____ С.А. Нардина

2. Рассмотрен и одобрен внешним экспертом

Директор ООО «Епортал» _____ И.И. Линник

М.П.

ИЗМЕНЕНИЯ И ДОПОЛНЕНИЯ
к фонду оценочных средств учебной дисциплины
Б1.В.13 Операционные системы
в составе ОПОП 09.03.02 Информационные системы и технологии

Ведомость изменений

Срок, с которого вводится изменение	Номер и основное содержание изменения и/или дополнения	Отметка об утверждении/ согласовании изменений	
		инициатор изменения	руководитель ОПОП или председатель МКН