МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Астраханский государственный университет имени В.Н. Татищева»
(Астраханский государственный университет им. В.Н. Татищева)

Филиал АГУ им. В.Н. Татищева в г. Знаменске Астраханской области

СОГЛАСОВАНО
Руководитель ОПОП
Бориско С.Н.
«13» ноября 2025 г.

УТВЕРЖДАЮ
Председатель ЦК (МО)
Фисенко Т.Ю.
протокол заседания ЦК (МО) №3
от «13» ноября 2025 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
по учебной дисциплине

ОПЕРАЦИОННЫЕ СИСТЕМЫ И СРЕДЫ

Составитель	Бориско С.Н., к.т.н., доцент, завкафедрой ЗнМИ; Мустафаев Н.Г., к.т.н., доцент кафедры ЗнМИ; Тимошкин А.А., к.т.н., доцент кафедры ЗнМИ; Устинов А.С., к.т.н., доцент кафедры ЗнМИ; Каштанов Д.Ю., ассистент кафедры ЗнМИ
Согласовано с работодателями	Литвинов С.П., к.т.н., заместитель командира войсковой части 15644 по научно-исследовательской и испытательной работе; Кирьянов М.Н., ведущий инженер ПАО «Ростелеком»
Наименование специальности	09.02.12 Техническая эксплуатация и сопровождение информационных систем
Квалификация выпускника	Специалист по технической эксплуатации и сопровождению информационных систем
Форма обучения	очная
Год приема (курс)	2026 (2 курс)

Знаменск, 2025 г.

СОДЕРЖАНИЕ

1. Общие положения

Фонд оценочных средств (далее – ФОС) предназначен для контроля и оценки результатов освоения обучающимися учебной дисциплины «Операционные системы и среды».

ФОС включают контрольные материалы для проведения текущего контроля успеваемости и промежуточной аттестации обучающихся, разработанные в соответствии с требованиями ФГОС СПО и содержанием рабочей программы учебной дисциплины.

2. Результаты освоения учебной дисциплины, подлежащие проверке

Код компетенции	Планируемые результаты освоения учебной дисциплины		
	Практический опыт	Умения	Знания
ОК 2	-способен применять теоретические знания на практике при работе с различными операционными системами; -умеет анализировать и решать задачи системного администрирования; - готов к освоению новых технологий в области операционных систем и сред.	- определять задачи для поиска информации, планировать процесс поиска, выбирать необходимые источники информации; -управлять параметрами загрузки операционной системы; -выполнять конфигурирование аппаратных устройств; -управлять учетными записями, настраивать параметры рабочей среды пользователей; -управлять дисками и файловыми системами, настраивать сетевые параметры, управлять разделением ресурсов в локальной сети.	-номенклатуры информационных источников, применяемых в профессиональной деятельности; -основные понятия, функции, состав и принципы работы операционных систем; - архитектуры современных операционных систем; - особенности построения и функционирования семейств операционных систем "Unix" и "Windows"; - принципы управления ресурсами в операционной системе; - основные задачи администрирования и способы их выполнения в изучаемых операционные системах.

3. Распределение оценивания результатов обучения по видам контроля

Наименование элемента практического опыта, умений или знаний	Наименование оценочного средства текущего контроля и промежуточной аттестации	
	Текущий контроль	Промежуточная аттестация
ПО.1. способен применять теоретические знания на практике при работе с различными операционными системами	Компьютерное тестирование на	Вопросы к экзамену

ПО.2. умеет анализировать и решать задачи системного администрирования; ПО.3 готов к освоению новых технологий в области операционных систем и сред	знание терминологии по теме. Контрольные задания, решение задач по теме.	
У1. определять задачи для поиска информации, планировать процесс поиска, выбирать необходимые источники информации; У2.использовать современное программное обеспечение; У3. управлять дисками и файловыми системами, настраивать сетевые параметры, управлять		
З1. номенклатуру информационных источников, применяемых в профессиональной деятельности; З2. основные понятия, функции, состав и принципы работы операционных систем; З3. особенности построения и функционирования семейств операционных систем "Unix" и "Windows"		

4. Контрольные задания для оценки результатов освоения учебной дисциплины

4.1. Контрольные задания для текущего контроля

Раздел 1. Основы операционных систем

Тема 1.1. История, назначение и функции операционных систем

Тестовые задания

1. (ОВ) Операционная система в первую очередь предназначена для:
a) Создания документов и таблиц
b) Управления ресурсами компьютера и обеспечения взаимодействия пользователя с аппаратурой
c) Программирования на языках высокого уровня
d) Защиты от компьютерных вирусов

2. (МВ) Какие из перечисленных задач решались самыми ранними операционными системами (пакетные ОС)?
a) Многозадачность с разделением времени
b) Автоматизация последовательности запуска программ
c) Управление памятью для нескольких программ
d) Удобный графический интерфейс пользователя

3. (СО) Идея, согласно которой несколько пользователей могут работать с одной ЭВМ одновременно, используя отдельные терминалы, называется _____.
Ответ: _____

4. (ОВ) Какая историческая ОС впервые широко популяризировала концепцию графического пользовательского интерфейса (GUI) для персональных компьютеров?
a) MS-DOS
b) UNIX
c) Windows 3.1
d) Mac OS (для компьютеров Apple Macintosh)

5. (СО) Операционная система, исходный код которой открыт для изучения и модификации, называется _____.
Ответ: _____

6. (MB) Какие из перечисленных функций являются основными функциями операционной системы?
a) Управление процессами (задачами)
b) Управление памятью (оперативной и виртуальной)
c) Управление файловой системой
d) Трансляция программ с языка высокого уровня на машинный код
7. (OB) Функция ОС, обеспечивающая изоляцию процессов друг от друга и контроль за их выполнением, называется:
a) Управление памятью
b) Управление процессами
c) Управление вводом-выводом
d) Защита информации
8. (CO) Механизм, создающий у каждой программы иллюзию наличия у неё собственного непрерывного адресного пространства, не зависящего от памяти других программ, называется _____ памятью.
Ответ: _____
9. (OB) Какая функция ОС позволяет работать программам, требующим для своей работы больше оперативной памяти, чем физически установлено в компьютере?
a) Кэширование диска
b) Файловая система
c) Виртуальная память
d) Многозадачность
10. (OB) Интерфейс, предоставляемый ОС для доступа программ к аппаратным ресурсам (дискам, сети, устройствам ввода-вывода), называется:
a) Пользовательский интерфейс (UI)
b) Аппаратный интерфейс
c) Системный вызов (API операционной системы)
d) Командная строка
11. (OB) Центральная, наиболее важная часть операционной системы, работающая в привилегированном режиме и имеющая полный доступ к аппаратуре, — это:
a) Командный процессор (shell)
b) Драйвер устройства
c) Файловая система
d) Ядро (kernel)
12. (MB) Какие из перечисленных компонентов обычно входят в состав ядра операционной системы (микроядро или монолитное)?
a) Драйверы устройств
b) Планировщик процессов
c) Графический редактор
d) Диспетчер памяти
13. (CO) Слой ПО, который преобразует запросы операционной системы к оборудованию в команды, понятные конкретному устройству, называется _____.
Ответ: _____
14. (OB) Компонент ОС, отвечающий за иерархическую организацию, хранение, именование и доступ к данным на долговременных носителях, — это:
a) Менеджер процессов
b) Файловая система
c) Подсистема безопасности
d) Диспетчер ввода-вывода
15. (CO) Программа, обеспечивающая интерпретацию команд пользователя (в текстовой или графической форме), называется _____ или оболочкой.
Ответ (английский термин): _____

16. (MB) Какие модели архитектуры ядра операционных систем существуют?
a) Монолитное ядро
b) Микроядро
c) Гибридное ядро
d) Многослойное ядро
17. (OB) В архитектуре на основе микроядра многие функции (управление файловой системой, сетевым стеком) выносятся:
a) В аппаратное обеспечение
b) В пользовательское пространство, в виде отдельных серверных процессов
c) В BIOS
d) В виртуальные машины
18. (CO) Режим работы процессора, в котором выполняется код ядра ОС и возможен прямой доступ к аппаратуре, называется _____ режимом (или режимом ядра).
Ответ: _____
19. (OB) Абстракция, создаваемая ОС для представления выполняющейся программы (код, данные, состояние регистров, ресурсы), называется:
a) Поток (thread)
b) Процесс
c) Задача
d) Дескриптор
20. (CO) Механизм, позволяющий одному процессу (родительскому) создать другой процесс (дочерний), который может быть копией родителя, называется _____.
Ответ (английский термин): _____

Ключ для проверки
1.b) Управления ресурсами компьютера и обеспечения взаимодействия пользователя с аппаратурой.
2.b) Автоматизация последовательности запуска программ.
3.Многопользовательский режим или Разделение времени (time-sharing).
4.d) Mac OS (для компьютеров Apple Macintosh).
5.Открытая или Open-source.
6. a, b, c (d — это функция компилятора/интерпретатора, не ядра ОС).
7. b) Управление процессами.
8. Виртуальной.
9. c) Виртуальная память.
10. c) Системный вызов (API операционной системы).
11. d) Ядро (kernel).
12. a, b, d (Графический редактор — прикладная программа).
13. Драйвер устройства.
14. b) Файловая система.
15. Shell (шелл).
16. a, b, c (Многослойная — это чаще подход к проектированию, а не отдельная модель архитектуры ядра).
17. b) В пользовательское пространство, в виде отдельных серверных процессов.
18. Привилегированный (или режим ядра, kernel mode).
19. b) Процесс.
20. Fork (форк, ветвление).

Практическое задание

Блок 1. История и эволюция ОС

Задание 1. Анализ исторической хронологии

1. Расположите следующие этапы развития ОС в хронологическом порядке:

    - Появление многозадачных ОС с разделением времени (time-sharing).
    - Широкое распространение ОС с графическим интерфейсом (GUI).
    - Пакетная обработка заданий (перфокарты, магнитные ленты).
    - Доминирование мобильных ОС (iOS, Android).
    - Эпоха персональных компьютеров с ОС, управляемых из командной строки.

2. Для каждого этапа напишите **одну ключевую характеристику**, которая отличает его от предыдущего.

Задание 2. Сравнительная таблица «Поколения ОС»
Заполните таблицу, указав основные черты, примеры ОС и ключевые проблемы, решаемые на каждом этапе.

Поколение / Эпоха	Характерные черты	Примеры ОС/Систем	Решаемая ключевая проблема
1. (1940-1950-е)	Отсутствие ОС, прямое управление аппаратурой.	Нет ОС	Эффективное использование машинного времени.
2. (1950-1960-е)	Пакетная обработка, мониторы.	IBM OS/360 (ранние версии), FORTRAN Monitor System	Автоматизация последовательности задач.
3. (1960-1970-е)	Многопрограммность, разделение времени, концепция процессов.	**UNIX**, Multics	Одновременная работа нескольких пользователей.
4. (1980-1990-е)	**Персональные компьютеры**, графический интерфейс, простота для неспециалиста.	**MS-DOS, Windows (3.1, 95), Mac OS, Linux (первые дистр.)**	?

Поколение / Эпоха	Характерные черты	Примеры ОС/Систем	Решаемая ключевая проблема
5. (2000-е — н.в.)	Сетевые, распределенные, мобильные, облачные ОС, виртуализация.	**Windows NT/10/11, macOS, Linux-дистрибутивы, Android, iOS**	?

Блок 2. Назначение и функции ОС

Задание 3. Анализ сценария «Что, если бы не было ОС?»
Представьте, что вы включаете современный компьютер (ПК или смартфон), но на нём **отсутствует операционная система**.

Опишите **5 конкретных действий**, которые вы НЕ сможете выполнить, и объясните почему (свяжите с отсутствием конкретной функции ОС).
Пример: «Я не смогу запустить текстовый редактор, потому что ОС не предоставляет интерфейс для поиска и запуска файлов программ (функция «пользовательский интерфейс») и не может загрузить код программы в память (функция «управление памятью»)».

Задание 4. Сопоставление функции и примера
Для каждой функции ОС приведите **реальный пример** её работы из вашего опыта работы с компьютером.
Пример для «Управление памятью»: «Когда я открываю много вкладок в браузере, ОС перемещает часть неиспользуемых данных из оперативной памяти в файл подкачки на диске, чтобы освободить место для новых задач».

Функция ОС	Ваш пример из жизни (1-2 предложения)
Управление процессами	
Управление памятью	
Управление файловой системой	
Управление вводом-выводом	
Защита и безопасность	
Сетевые функции	

Функция ОС	Ваш пример из жизни (1-2 предложения)
Пользовательский интерфейс	

Блок 3. Состав и взаимодействие компонентов

Задание 5. Моделирование работы ОС на примере «Ресторан»
Представьте, что операционная система — это большой ресторан. Сопоставьте компоненты ОС с ролями и объектами в ресторане. Обоснуйте свой выбор.
*Пример: **Ядро ОС** — это **управляющий (менеджер) ресторана**. Он координирует работу всех отделов, имеет доступ ко всему и принимает ключевые решения.*

Компонент/Объект в ОС	Аналог в ресторане	Краткое обоснование аналогии
1. Процесс (задача)	Заказ клиента	Имеет чёткую цель, требует ресурсов (повара, ингредиенты), проходит этапы (принят, готовится, выполнен).
2. Планировщик процессов	Администратор залов	Решает, какой заказ (процесс) и на какой кухне (процессор) выполнять следующим, чтобы избежать простоев.
3. Оперативная память (ОЗУ)	Столы на кухне	Рабочее пространство, где повара (процессор) готовят блюда (обрабатывают данные). Ограничено по размеру.
4. Файловая система	Склад и система хранения	Хранит ингредиенты и рецепты (данные и программы) в упорядоченном виде для быстрого поиска и доступа.
5. Драйвер устройства	Переводчик для специфичного оборудования	Если на кухню привезли новую французскую кофемашину, нужен специалист (драйвер), который умеет с ней работать и объяснит поварам, как её использовать.
6. Системные вызовы (API)	Стандартная форма заказа	Унифицированный способ для официантов (приложений) передать заказ (запрос) на кухню (ядру ОС).

Задание 6. Диаграмма взаимодействия «Запуск программы»
Нарисуйте упрощенную блок-схему (или опишите текстом последовательность), иллюстрирующую взаимодействие основных компонентов ОС при выполнении **простого действия пользователя**: *двойной щелчок по значку текстового редактора*.

- В схеме должны быть отражены:

1. Пользовательский интерфейс (рабочий стол).
2. Файловая система (поиск файла .exe на диске).
3. Диспетчер памяти (выделение ОЗУ под программу).
4. Управление процессами (создание нового процесса).
5. Планировщик (постановка процесса в очередь на выполнение).

- Стрелками укажите направление передачи запросов и данных.

Задание 7. Технический анализ (продвинутый уровень)

1. Откройте **Диспетчер задач** (Windows) или **Системный монитор** (Linux/macOS).
2. Найдите вкладки/разделы, которые отображают информацию о:

- Загрузке процессора (ЦП)
- Использовании оперативной памяти (память)
- Активных процессах и службах
- Активности диска и сети

3. Сделайте скриншот или запишите текущие показатели.
4. **Сопоставьте каждый из этих показателей с конкретной функцией ОС**, ответственной за управление соответствующим ресурсом.
Пример: «Высокий процент использования ЦП (90%) говорит о том, что функция «Управление процессами» и «Планирование» активно распределяет время процессора между многими задачами».

Критерии оценки выполнения заданий:

- **Блоки 1-2 (История, функции):** Оценивается полнота, точность сопоставлений, глубина понимания причинно-следственных связей между развитием технологий и появлением новых функций ОС.
- **Блок 3 (Состав и взаимодействие):** Оценивается корректность аналогий, логичность и полнота описания взаимодействий между компонентами, умение абстрагироваться и видеть систему в целом.
- **Общая оценка:** Учитывается самостоятельность выполнения, ясность изложения мыслей, использование правильной терминологии и творческий подход в заданиях на аналогию.

Тема 1.3. Общие сведения о процессах и потоках

Тестовые задания

1. (ОВ) Процесс в операционной системе — это:
a) Выполняющаяся программа со всем её контекстом (код, данные, состояние)
b) Физический файл на диске с расширением .exe или .bin

c) Любое приложение, установленное в системе
d) Только фоновые службы (службы Windows или демоны Linux)

2. (MB) Какие из перечисленных элементов входят в контекст (образ) процесса?
a) Код программы (текстовый сегмент)
b) Данные программы (глобальные переменные)
c) Состояние регистров процессора (счётчик команд, указатель стека)
d) Состояние открытых файлов и устройств ввода-вывода

3. (CO) Абстрактная структура данных в ядре ОС, которая хранит всю информацию, необходимую для описания и управления процессом, называется _____ процесса.
Ответ: _____

4. (OB) Основное отличие процесса от программы заключается в том, что процесс:
a) Занимает больше места на диске
b) Является пассивным набором инструкций, а процесс — его активным выполнением
c) Всегда имеет графический интерфейс
d) Не может быть остановлен

5. (CO) Процесс может находиться в одном из трёх основных состояний: Выполнение (Running), Готовность (Ready) и _____.
Ответ: _____

6. (MB) В состояние «Ожидание» (Blocked/Waiting) процесс переходит, когда:
a) Закончилось выделенное ему время процессора (квант времени)
b) Он ожидает завершения операции ввода-вывода
c) Он ожидает освобождения какого-либо ресурса (например, семафора)
d) Он был только что создан системным вызовом

7. (OB) Планировщик (Scheduler) отвечает за:
a) Переключение процесса из состояния «Выполнение» в состояние «Ожидание»
b) Выбор процесса из состояния «Готовность» для перевода в состояние «Выполнение»
c) Создание новых процессов
d) Завершение процессов

8. (CO) Переключение контекста (Context Switch) — это операция сохранения состояния одного процесса и загрузки состояния другого. Этот механизм реализуется _____ ОС.
Ответ: _____ *(ядром / пользовательской программой)*

9. (OB) В иерархической модели процессов процесс, создавший другой процесс, называется:
a) Дочерним процессом (Child Process)
b) Сестринским процессом (Sibling Process)
c) Родительским процессом (Parent Process)
d) Потомком (Descendant)

10. (CO) В большинстве ОС при завершении родительского процесса обычно также завершаются все его _____ процессы.
Ответ: _____

11. (MB) Какие из перечисленных причин могут привести к завершению процесса?
a) Нормальное завершение (самостоятельный выход)
b) Принудительное завершение (сигнал KILL)

c) Ошибка времени выполнения (например, деление на ноль)
d) Завершение по истечении выделенного времени работы

12. (OB) В UNIX-подобных системах классическим механизмом создания нового процесса является системный вызов:
a) exec()
b) fork()
c) create()
d) spawn()

13. (CO) Системный вызов fork() **создаёт новый процесс, который является почти точной _____ родительского процесса.**
Ответ: _____

14. (MB) Какие из перечисленных объектов НЕ наследуются дочерним процессом при вызове fork() **в UNIX?**
a) Открытые файловые дескрипторы
b) Значения глобальных переменных
c) Идентификатор процесса (PID)
d) Текст программы (код)

15. (OB) Для того чтобы дочерний процесс начал выполнять другую программу (отличную от родительской), после fork() **используется системный вызов:**
a) wait()
b) exit()
c) exec()
d) clone()

16. (CO) В операционных системах Windows API для создания процесса чаще всего используется функция _____.
Ответ: _____ *(название функции WinAPI)*

17. (OB) Уникальный числовой идентификатор, присваиваемый каждому процессу в системе, называется:
a) UID (User Identifier)
b) GID (Group Identifier)
c) PPID (Parent Process Identifier)
d) PID (Process Identifier)

18. (CO) Идентификатор родительского процесса обозначается аббревиатурой _____.
Ответ: _____

19. (MB) Какие из перечисленных элементов являются типичными атрибутами (свойствами) процесса?
a) Приоритет планирования
b) Права доступа (владелец, пользователь/группа)
c) Использование процессорного времени и памяти (статистика)
d) Физический адрес в оперативной памяти

20. (OB) Совокупность всех процессов в системе и их иерархических связей («родитель-потомок») образует:
a) Лес процессов
b) Дерево процессов

c) Очередь процессов
d) Таблицу процессов

Ключ для проверки

1. **a)** Выполняющаяся программа со всем её контекстом (код, данные, состояние).
2. **a, b, c, d** (Все перечисленные элементы входят в контекст).
3. **Дескриптор** (или **блок управления процессом — PCB, Process Control Block**).
4. **b)** Является пассивным набором инструкций, а процесс — его активным выполнением.
5. **Ожидание** (Blocked/Waiting).
6. **b, c** (a — переход в «Готовность», d — начальное состояние может быть «Готовность»).
7. **b)** Выбор процесса из состояния «Готовность» для перевода в состояние «Выполнение».
8. **ядром** (Kernel).
9. **c)** Родительским процессом (Parent Process).
10. **дочерние** (child).
11. **a, b, c** (d — некорректно, процесс завершается не по таймеру, а переводится в готовность).
12. **b)** fork().
13. **копией** (replica).
14. **c)** Идентификатор процесса (PID) (Он всегда уникален и новый у дочернего процесса).
15. **c)** exec().
16. CreateProcess() (Или CreateProcessA/CreateProcessW).
17. **d)** PID (Process Identifier).
18. **PPID**.
19. **a, b, c** (d — не атрибут, а деталь реализации управления памятью; процесс оперирует виртуальными адресами).
20. **b)** Дерево процессов (Изначально есть корневой процесс, от которого порождаются все остальные).

Практическое задание

Блок 1. Модель процесса и его состояния

Задание 1. «Карта состояний процесса»

1. Нарисуйте диаграмму состояний процесса с тремя основными состояниями: **Выполнение (Running)**, **Готовность (Ready)**, **Ожидание/Блокировка (Blocked/Waiting)**.
2. На стрелках, соединяющих состояния, укажите **конкретные события**, которые вызывают каждое переход. Минимум по 2 события для каждого возможного перехода.
   Пример: Переход «Выполнение» → «Ожидание»: 1) Запрос операции ввода-вывода; 2) Ожидание семафора.

Задание 2. «Разбор PCB»
Представьте, что вы — ядро ОС. У вас есть следующий фрагмент «памяти» (воображаемый PCB) для процесса TextEditor.exe с PID=4512:

- **PID:** 4512
- **PPID:** 1120 (PID Explorer.exe)
- **Состояние:** READY
- **Счётчик команд (Program Counter):** 0x004015A0
- **Указатель стека:** 0x006FFA2C
- **Приоритет:** Нормальный
- **Список открытых файлов:** C:\docs\essay.txt (чтение/запись), stdout
- **Статистика:** Время ЦП = 1.2 сек, Память = 15 МБ

Вопросы:

1. Какой процесс является родителем для TextEditor.exe? Что это может означать с точки зрения пользователя?
2. Объясните, что такое «Счётчик команд» и «Указатель стека». Почему они критически важны при переключении контекста?
3. Если состояние процесса READY, что ему не хватает для выполнения? Кто решает, когда он получит этот ресурс?
4. Перечислите действия ядра ОС, если этот процесс запросит чтение следующей строки из файла essay.txt. Как изменятся его состояние и атрибуты?

Блок 2. Создание процессов (UNIX/Linux модель)

Задание 3. «Трассировка системных вызовов: fork() и exec()»
Рассмотрите следующий упрощённый псевдокод программы на C:

```c
int main() {
    pid_t pid = fork(); // Системный вызов 1
    if (pid == 0) {
        // Код дочернего процесса
        printf("I am child. My PID is %d, my PPID is %d\n", getpid(), getppid());
        execl("/bin/ls", "ls", "-l", NULL); // Системный вызов 2
        printf("This line should NOT be printed!\n");
    } else if (pid > 0) {
        // Код родительского процесса
        printf("I am parent. My PID is %d, my child's PID is %d\n", getpid(), pid);
        wait(NULL); // Системный вызов 3
        printf("Child process finished.\n");
    }
    return 0;
}
```

Вопросы для анализа:

1. Что возвращает fork() в родительский и в дочерний процесс? Почему?
2. Сколько всего процессов будет существовать сразу после успешного выполнения строки с fork()? Опишите их.
3. Что происходит при вызове execl() в дочернем процессе? Объясните, почему строка printf("This line...") никогда не выполняется.
4. Какую роль выполняет системный вызов wait() в родительском процессе? Что произойдет, если убрать эту строку?

5. Предположите и запишите возможный **порядок вывода строк** на экран при выполнении этой программы.

Задание 4. «Построение дерева процессов»
Используя командную строку Linux (или эмулятор), выполните команду:

```
bash
sleep 60 &
ps -ef --forest | grep -A2 -B2 sleep
```

Или на Windows в PowerShell:

```
powershell
Start-Process -NoNewWindow "cmd.exe" "/c timeout 60"
Get-WmiObject Win32_Process | Where-Object {$_.Name -eq "timeout.exe"} | Select-Object Pr
ocessId, ParentProcessId, Name
```

1. Найдите созданный процесс (sleep или timeout). Запишите его PID и PPID.
2. Используя команду pstree -p (Linux) или данные из PowerShell, постройте фрагмент дерева процессов, показав цепочку «предок → родитель → наш процесс».
3. Смоделируйте вручную дерево процессов для следующего сценария: Пользователь запускает терминал (bash, PID=5000). Из него запускается скрипт myscript.sh (PID=5050), который, в свою очередь, вызывает утилиту grep (PID=5055). Изобразите это дерево с указанием PID и PPID для каждого узла.

Блок 3. Сравнительный анализ и моделирование

Задание 5. «Windows vs. UNIX: Философия создания»
Заполните сравнительную таблицу:

Критерий	UNIX-подобные системы (fork/exec)	Windows (CreateProcess)
Основная модель	Двухэтапная: 1. Клонирование (fork), 2. Замена образа (exec).	Одноэтапная: Создание и загрузка образа в одном вызове.
Начальное состояние	Дочерний процесс — почти точная копия родителя (память, дескрипторы).	Дочерний процесс запускается с чистого листа (образ из файла).
Преимущество	? (Гибкость, возможность настроить среду до exec).	? (Эффективность, меньше накладных расходов).
Недостаток	? (Накладные расходы на копирование памяти, которые могут быть избыточны).	? (Меньше гибкости при подготовке окружения для нового процесса).
Аналог в природе/жизни	Деление клетки (митоз), где дочерняя клетка сначала копирует родительскую, а затем специализируется.	Строительство нового дома по готовому проекту на пустом участке.

Задание 6. «Симулятор планировщика» (Практическое моделирование)

1. **Цель:** Вручную промоделировать работу планировщика процессов.
2. **Исходные данные:** Есть 3 процесса (P1, P2, P3) в состоянии **Ready**.
   - P1: Приоритет высокий, требует 3 кванта времени.
   - P2: Приоритет нормальный, требует 2 кванта времени, но после 1-го кванта перейдет в **Blocked** (ожидание I/O) на 2 кванта.
   - P3: Приоритет низкий, требует 4 кванта времени.
3. **Алгоритм планирования:** Невытесняющий (кооперативный) с учетом приоритета. Выбирается процесс с наивысшим приоритетом и выполняется, пока не отдаст ЦП (завершится или перейдет в Blocked).
4. **Задание:** Составьте **таблицу хода выполнения**. Столбцы: Такт времени (1,2,3...), Состояние P1, Состояние P2, Состояние P3, Примечание (кто выполняется, почему).
   Пример начала:

Такт	P1	P2	P3	Примечание
1	Running	Ready	Ready	Выполняется P1 (высший приоритет).
2	Running	Ready	Ready	P1 продолжает выполняться.

5. Промоделируйте до полного завершения всех процессов.
6. **Вопрос для размышления:** Как изменилась бы таблица, если бы планировщик был вытесняющим с квантом времени = 1? Смоделируйте первые 3 такта для этого случая.

Критерии оценки:

- **Задание 1-2 (Понимание модели):** Оценивается правильность и полнота диаграммы, глубина понимания атрибутов процесса и их роли.
- **Задание 3-4 (Создание процессов):** Оценивается точность анализа кода, понимание последовательности и последствий системных вызовов, умение работать с системными утилитами.
- **Задание 5-6 (Сравнение и моделирование):** Оценивается способность к сравнительному анализу, логичность построения модели и таблицы выполнения, понимание различий алгоритмов планирования.
- **Общее:** Четкость изложения, использование правильной терминологии, самостоятельность выполнения.

Тема 1.5 Управление памятью

Тестовые задания

1. (ОВ) Основная цель абстракции памяти, предоставляемой операционной системой процессу, — это:
a) Увеличение физического объема оперативной памяти (ОЗУ)
b) Предоставление каждому процессу иллюзии, что он один владеет всей памятью компьютера
c) Ускорение работы процессора за счет кэширования
d) Организация обмена данными между процессами

2. (MB) Какие проблемы решает введение абстракции виртуального адресного пространства?
a) Изоляция процессов: ошибка в одной программе не портит память другой.
b) Упрощение программирования: программист не думает о реальном расположении данных в физической памяти.
c) Возможность загрузки программы в произвольное место физической памяти.
d) Гарантированное увеличение скорости выполнения программ.

3. (CO) Логический (виртуальный) адрес, генерируемый процессом, преобразуется в физический адрес с помощью аппаратного блока, называемого _____ (MMU).
Ответ (полное название): _____

4. (OB) Минимальная единица памяти, которой управляет ОС при распределении и защите, называется:
a) Байт
b) Кэш-линия
c) Страница (Page) или Сегмент (Segment)
d) Файл

5. (CO) Требование, чтобы программа для своего выполнения была целиком загружена в непрерывную область физической памяти, называется требованием _____ размещения.
Ответ: _____

6. (OB) При страничной организации виртуальное адресное пространство процесса и физическая память делятся на фиксированные блоки одинакового размера, называемые:
a) Сегментами
b) Кадрами (Frames) – в физической памяти, Страницами (Pages) – в виртуальной
c) Кластерами
d) Блоками

7. (MB) Какие утверждения о страничной организации верны?
a) Внешняя фрагментация (фрагментация свободной памяти между занятыми регионами) практически устранена.
b) Внутренняя фрагментация (недозаполнение последней страницы) возможна.
c) Размер страницы обычно является степенью двойки (например, 4 КБ).
d) Для преобразования адреса используется сегментная таблица.

8. (CO) Структура данных в ядре ОС, хранящая соответствие виртуальных страниц процесса физическим кадрам, называется _____.
Ответ: _____

9. (OB) Запись в таблице страниц (Page Table Entry, PTE), которая указывает, что запрошенная страница в данный момент не находится в оперативной памяти, помечается специальным битом, называемым битом _____.
a) Доступа (Access bit)
b) Изменения (Dirty bit)
c) Присутствия/Отсутствия (Present/Valid bit)
d) Защиты (Protection bit)

10. (CO) Быстрое аппаратное буферное хранилище, содержащее недавно использованные отображения «виртуальная страница → физический кадр»,

называется _____ (TLB).
Ответ (полное название): _____

11. (OB) Концепция виртуальной памяти позволяет:
a) Выполнять программу, размер которой превышает объем доступной физической оперативной памяти (ОЗУ)
b) Увеличить тактовую частоту процессора
c) Объединить оперативную память нескольких компьютеров в общий пул
d) Отключить файл подкачки для ускорения работы

12. (MB) Какие из перечисленных утверждений характеризуют работу виртуальной памяти?
a) Активные страницы процессов хранятся в ОЗУ.
b) Неактивные страницы могут быть вытеснены (сброшены) на диск, в специальную область — файл подкачки (swap file/pagefile).
c) При обращении к странице, отсутствующей в ОЗУ, происходит аппаратное прерывание — страничная ошибка (page fault).
d) Виртуальная память делает избыточным использование кэш-памяти процессора.

13. (CO) Аппаратное прерывание, возникающее при попытке доступа к виртуальной странице, которая отсутствует в оперативной памяти, называется _____.
Ответ: _____

14. (OB) Алгоритм, используемый для выбора страницы-жертвы (victim page) для вытеснения на диск при нехватке свободных кадров, называется алгоритмом:
a) Планирования процессов
b) Замены страниц (Page Replacement Algorithm)
c) Диспетчеризации ввода-вывода
d) Синхронизации памяти

15. (CO) Простейший и неэффективный алгоритм замены страниц, вытесняющий ту страницу, которая дольше всего находилась в памяти, называется _____.
Ответ: _____

16. (OB) При сегментной организации памяти виртуальное адресное пространство делится на блоки переменного размера (сегменты) в соответствии с:
a) Логической структурой программы (код, данные, стек, куча)
b) Фиксированным размером, задаваемым аппаратурой
c) Временем последнего обращения к блоку
d) Частотой обращений к блоку

17. (MB) Какие из перечисленных особенностей характерны для сегментной организации по сравнению со страничной?
a) Естественная поддержка защиты (например, сегмент кода только для чтения).
b) Отсутствие внутренней фрагментации.
c) Наличие внешней фрагментации.
d) Простота и высокая скорость преобразования адресов.

18. (CO) Для преобразования логического адреса в физический при сегментной организации используется структура данных, называемая _____.
Ответ: _____

19. (ОВ) Современные процессорные архитектуры (например, x86-64) для управления памятью используют:
a) Чисто сегментную модель
b) Чисто страничную модель
c) Комбинированную сегментно-страничную модель (хотя сегментация часто сводится к минимуму)
d) Файловую модель

20. (СО) С точки зрения программиста, виртуальное адресное пространство процесса часто делится на несколько областей. Область для динамического выделения памяти во время выполнения (например, через malloc/new) называется _____.
Ответ: _____

Ключ для проверки

Раздел 1:

1. **b)** Предоставление каждому процессу иллюзии, что он один владеет всей памятью компьютера.
2. **a, b, c** (d — скорость не гарантируется, может даже снизиться из-за накладных расходов).
3. **Блок управления памятью** (Memory Management Unit).
4. **c)** Страница (Page) или Сегмент (Segment).
5. **непрерывного**.

Раздел 2:
6. **b)** Кадрами (Frames) – в физической памяти, Страницами (Pages) – в виртуальной.
7. **a, b, c** (d — для преобразования адреса используется **таблица страниц (page table)**).
8. **Таблица страниц** (Page Table).
9. **c)** Присутствия/Отсутствия (Present/Valid bit).
10. **Буфер ассоциативной трансляции** (Translation Lookaside Buffer).

Раздел 3:
11. **a)** Выполнять программу, размер которой превышает объем доступной физической оперативной памяти (ОЗУ).
12. **a, b, c** (d — неверно, кэш-память критически важна для скорости, виртуальная память решает другую задачу).
13. **Страничная ошибка** (Page Fault).
14. **b)** Замены страниц (Page Replacement Algorithm).
15. **FIFO** (First-In, First-Out).

Раздел 4:
16. **a)** Логической структурой программы (код, данные, стек, куча).
17. **a, b, c** (d — как правило, преобразование адреса в сегментной системе сложнее).
18. **Таблица сегментов** (Segment Table).
19. **c)** Комбинированную сегментно-страничную модель (хотя сегментация часто сводится к минимуму).
20. **Куча** (Heap).

Практическое задание

Блок 1: Абстракция памяти и таблицы страниц

Задание 1. «Карта виртуальной памяти процесса»

1. Используя утилиту /proc/self/maps в Linux (или vmmap на macOS, Process Explorer на Windows), изучите карту памяти **любого запущенного процесса** (например, вашего браузера или текстового редактора).
2. Выпишите несколько строк вывода и для каждой области определите:
   - **Диапазон виртуальных адресов** (например, 00400000-00401000)
   - **Права доступа** (rwxp, r--p и т.д.)
   - **Назначение области** (код программы, данные, стек, куча, разделяемые библиотеки, файл подкачки).
3. Создайте **схематическую диаграмму** виртуального адресного пространства этого процесса, расположив области в порядке возрастания адресов. Подпишите, какая область для чего используется.

Задание 2. «Вручную декодируем виртуальный адрес» (Страничная организация)

Дано:

- Размер страницы = **4 КБ (4096 байт)**.
- Размер виртуального адреса = **32 бита**.
- Виртуальный адрес, сгенерированный программой: **0x3A7C29F1** (в шестнадцатеричном формате).

Шаги:

1. Рассчитайте, сколько бит отводится под **смещение внутри страницы (offset)**. Обоснуйте.
2. Рассчитайте, сколько бит отводится под **номер виртуальной страницы (VPN - Virtual Page Number)**.
3. Переведите адрес 0x3A7C29F1 в двоичный вид (или используйте шестнадцатеричное деление). **Визуально разделите** двоичное представление на поле VPN и поле Offset.
4. Извлеките **шестнадцатеричное значение VPN** из этого адреса.
5. Предположим, что в таблице страниц для этого процесса VPN, который вы нашли, соответствует **номеру физического кадра (PFN) = 0x0AB**.
   Рассчитайте **физический адрес** в памяти, соответствующий виртуальному адресу 0x3A7C29F1. Ответ представьте в шестнадцатеричном виде.

Блок 2: Работа виртуальной памяти и алгоритмы замены

Задание 3. «Моделирование алгоритмов замены страниц»

Дана последовательность обращений к виртуальным страницам процесса:
1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

Пусть размер физической памяти (число доступных кадров) = **3**.

Промоделируйте работу следующих алгоритмов замены страниц, заполнив таблицу. Для каждого шага укажите содержимое физических кадров и отметьте, было ли **промах страницы (Page Fault, PF)**.

Алгоритм FIFO (First-In, First-Out):

Шаг (Обращение)	Кадр 1	Кадр 2	Кадр 3	Промах (PF)?	Примечание (какая страница вытеснена)
1 (страница 1)	1	-	-	**Да**	
2 (2)	1	2	-	Да	
3 (3)	1	2	3	Да	
4 (4)	...	...	...	...	... (продолжить самостоятельно)

Алгоритм LRU (Least Recently Used):

- Постройте аналогичную таблицу для алгоритма LRU. Помните: при выборе жертвы удаляется страница, к которой **давнее всего не обращались**.

Вопросы для анализа:

1. Какой алгоритм показал меньше промахов страниц для данной последовательности?
2. Что такое «аномалия Белади» (Belady's Anomaly)? Можете ли вы обнаружить её признаки в своих таблицах? (Подсказка: попробуйте увеличить число кадров до 4 и снова промоделировать FIFO для той же последовательности).

Задание 4. «Анализ влияния размера страницы»
Представьте, что у вас есть два типа вычислительных нагрузок:

- **Нагрузка А:** Программа последовательно обрабатывает большой массив данных.
- **Нагрузка В:** Программа выполняет много случайных обращений к небольшому объему разрозненных данных (например, работа с указателями в графе).

Задача: Проанализируйте, как выбор **большего** (например, 1 МБ) или **меньшего** (например, 4 КБ) размера страницы повлияет на:

1. **Частоту страничных ошибок (page fault rate).**
2. **Эффективность TLB (буфера ассоциативной трансляции).**
3. **Уровень внутренней фрагментации.**
4. **Накладные расходы на обслуживание таблиц страниц.**

Заполните таблицу, отметив для каждой нагрузки, какой размер страницы (большой/маленький) был бы **предпочтительнее** по каждому критерию и почему.

Критерий	Нагрузка A (последовательный доступ)	Нагрузка B (случайный доступ)
Частота страничных ошибок	? (Большая/Маленькая страница)	?
Эффективность TLB	?	?
Внутренняя фрагментация	?	?
Накладные расходы на таблицы	?	?

Блок 3: Сравнительный анализ и проектирование

Задание 5. «Сегменты vs. Страницы: Выбор архитектуры»
Вы — архитектор новой ОС для встраиваемой системы с жесткими требованиями к безопасности и предсказуемости.

Требования:

1. Критически важно, чтобы код не мог быть случайно перезаписан данными.
2. Нужна простая и быстрая проверка прав доступа (например, «эта область только для выполнения»).
3. Внешняя фрагментация допустима, так как набор задач фиксирован.
4. Размер памяти мал, накладные расходы на метаданные должны быть минимальны.

Задача: Обоснуйте, какая модель организации памяти — **сегментная** или **страничная** — лучше подходит под эти требования. Для каждого из 4-х требований объясните, как выбранная вами модель его удовлетворяет, а альтернативная — нет. Составьте сравнительную таблицу с вашей аргументацией.

Задание 6. «Расследование: Высокая загрузка диска из-за свопинга»
Ситуация: Пользователь жалуется на сильные «тормоза» системы. Вы замечаете, что индикатор активности диска почти постоянно горит, хотя активных задач мало. Подозрение падает на чрезмерный **свопинг** (swapping/thrashing).

Ваши действия (опишите по шагам):

1. Какие **системные утилиты или счетчики производительности** вы будете использовать для первичной диагностики? (Например, для Linux: `top`, `vmstat`, `sar`; для Windows: Диспетчер задач, Performance Monitor).
2. На какие **ключевые метрики** вы будете смотреть, чтобы подтвердить гипотезу о thrashing? (Назовите минимум 3).
3. Каковы **возможные причины** возникновения thrashing?
4. Предложите **практические шаги по устранению** проблемы (как краткосрочные, так и долгосрочные).

Критерии оценки:

- **Блок 1 (Понимание основ):** Оценивается точность расчетов, корректность интерпретации данных системных утилит, понимание процесса трансляции адресов.
- **Блок 2 (Анализ механизмов):** Оценивается корректность моделирования алгоритмов, качество сравнительного анализа влияния параметров, глубина понимания компромиссов (trade-offs).
- **Блок 3 (Применение и синтез):** Оценивается логичность и убедительность аргументации при выборе модели, системность подхода к диагностике реальной проблемы, полнота предлагаемых решений.
- **Общее:** Четкость изложения, использование профессиональной терминологии, творческий подход в заданиях на моделирование и анализ.

4.2 Контрольные задания для промежуточной аттестации

Вопросы для экзамена

1. Дайте определение операционной системы с точки зрения её двух основных функций.
2. Назовите ключевые вехи в истории развития ОС. Чем характеризовался переход от пакетных систем к системам с разделением времени?
3. В чём заключается идея абстракции ресурсов, предоставляемой ОС? Приведите три конкретных примера таких абстракций.
4. Что такое ядро (Kernel) ОС? Опишите разницу между монолитным и микроядром.
5. Какие функции ОС обеспечивают безопасность и защиту данных? Приведите примеры механизмов реализации этих функций.
6. Объясните, как функции управления вводом-выводом упрощают разработку прикладных программ.

7. Опишите слоистую (многоуровневую) архитектуру ОС. Каковы её преимущества и недостатки?
8. Что такое системные вызовы (System Calls)? Какова их роль в архитектуре «пользовательский режим — режим ядра»?
9. Дайте определение понятию «драйвер устройства». Почему большинство драйверов работает в режиме ядра, и какие риски это несет?
10. Что такое виртуальная машина (Virtual Machine) с точки зрения архитектуры ОС? Чем гипервизор (монитор виртуальных машин) первого типа отличается от второго?
11. Объясните назначение и принцип работы командного процессора (shell). Как он взаимодействует с ядром ОС?

12. Дайте развернутое определение процесса. Что входит в его контекст (образ)?
13. Опишите жизненный цикл процесса, перечислив его основные состояния и возможные переходы между ними.
14. Что такое поток исполнения (thread)? В чём его принципиальное отличие от процесса с точки зрения разделения ресурсов?
15. Опишите модель «один процесс – один поток» и модель «один процесс – много потоков». Каковы области применения каждой?

16. Что происходит в системе при системном вызове fork() в Unix-подобных ОС? Чем порожденный процесс отличается от родительского?
17. Для чего используется системный вызов exec()? Почему он обычно вызывается в дочернем процессе после fork()?
18. Что такое дескриптор процесса (PCB)? Какая информация в нём хранится и для чего она используется ядром?

19. Какие проблемы возникают при параллельном выполнении процессов/потоков? Дайте определение состояния гонки (race condition).
20. Опишите классические проблемы синхронизации: «Обедающие философы» и «Читатели-писатели». В чём их суть?
21. Что такое семафор? Объясните разницу между двоичным и счетным семафором. Приведите пример использования.
22. Что такое взаимная блокировка (deadlock)? Сформулируйте четыре необходимых условия для её возникновения.
23. Опишите различия между вытесняющим (preemptive) и невытесняющим (non-preemptive) планированием задач.
24. Сравните алгоритмы планирования процессов: Round Robin (циклическое планирование), Shortest Job First (наикратчайшая задача первая), планирование по приоритетам. Укажите их сильные и слабые стороны.

25. Сформулируйте проблему, которую решает механизм виртуальной памяти. Какие возможности он предоставляет программисту и системе?
26. Опишите принцип страничной организации памяти. Что такое страница, кадр, таблица страниц?
27. Что такое TLB (буфер ассоциативной трансляции)? Как он ускоряет преобразование виртуальных адресов в физические?
28. Дайте определение страничной ошибки (page fault). Опишите последовательность действий ядра при её обработке.
29. Сравните алгоритмы замещения страниц FIFO и LRU. Какой из них более эффективен и почему? Что такое «аномалия Беллади»?
30. В чём заключаются основные различия между страничной и сегментной организацией памяти? Какие преимущества и недостатки есть у каждой?

31. Перечислите основные задачи файловой системы. Что такое файл и каталог с точки зрения ОС?
32. Опишите иерархическую структуру файловой системы. Что такое абсолютный и относительный путь к файлу?
33. Что такое индексный дескриптор (inode) в Unix-подобных системах? Какую информацию о файле он хранит?
34. Объясните разницу между блочными и символьными (символьно-ориентированными) устройствами. Приведите примеры каждого типа.
35. Что такое буферизация и кэширование в контексте ввода-вывода? Как эти механизмы повышают производительность системы?

36. Опишите основные принципы разграничения прав доступа к файлам в Unix-подобных системах (права rwx для user, group, other).
37. Что такое переменная окружения (environment variable)? Какова её роль в работе командной оболочки и приложений?
38. Объясните назначение и основные возможности системных утилит для мониторинга процессов и ресурсов (например, top, ps, htop в Linux; Диспетчер задач в Windows).

39. Что такое конвейер (pipeline) в командной строке? Приведите пример его использования для решения задачи фильтрации и обработки данных.
40. Опишите этапы загрузки операционной системы: от включения питания до появления приглашения командной строки или графического интерфейса.

Критерии оценки

Оценка «5» - (отлично)

При ответе материал изложен грамотным языком в определенной логической последовательности, точно использована терминология, полно раскрыто содержание материала в объеме, предусмотренном программой, продемонстрировано усвоение ранее изученных сопутствующих вопросов. Возможны одна - две неточности при освещении второстепенных вопросов.

Оценка «4» - (хорошо)

Ответ удовлетворяет в основном требованиям на оценку «5», но при этом имеет один из недостатков: в изложении допущены небольшие пробелы; допущены один – два недочета при освещении основного содержания ответа, исправленные по замечанию преподавателя; допущены ошибка или более двух недочетов при освещении второстепенных вопросов, легко исправленные по замечанию преподавателя.

Оценка «3» - (удовлетворительно)

При ответе неполно или непоследовательно раскрыто содержание материала, но показано общее понимание, имелись затруднения или допущены ошибки в определении понятий.

Оценка «2» - (неудовлетворительно)

При ответе не раскрыто основное содержание учебного материала; обнаружено незнание или непонимание обучающимся большей или наиболее важной части учебного материала; допущены ошибки в определении понятий, допущены существенные ошибки, показавшие, что обучающийся не владеет обязательными умениями по данной теме в полной мере.