
Дисциплина: Операционные системы

§ 5. Понятие ресурса ОС

ПЛАН:

1. Понятие ресурса в ОС.
2. Классификация ресурсов по признакам.

Понятие ресурса

Ресурс - всякий потребляемый объект (независимо от формы его существования), обладающий некоторой практической ценностью для потребителя.

Ресурсы по запасу выделяемых единиц ресурса бывают исчерпаемые и неисчерпаемые.

Ресурс - средство системы обработки данных, которое может быть выделено процессу обработки данных на определенный интервал времени.

Простыми словами, **ресурс** - это все те аппаратные программные средства и данные, которые необходимы для выполнения программы.

К основным ресурсам могут быть отнесены **процессоры, память, внешние устройства, данные и программы**

Классификация ресурсов

- По реальности существования: физический и виртуальный;
- По возможности расширения свойств: эластичный и жесткий
- По степени активности: активный (ЦП) и пассивный (область памяти, выделяемая по требованию)
- По времени существования: постоянный, временный
- По степени важности: главный (ЦП и ОП) и второстепенный
- По структуре: простой, составной;

Классификация ресурсов

- По восстанавливаемости: воспроизводимый, потребляемый
 - По форме реализации: мягкий (программные и информационные), твердый (аппаратные компоненты машины, а также человеческие ресурсы)
 - По функциональной избыточности: дорогой, дешевый.
 - По характеру использования: параллельно используемый, последовательно используемый;
-

Классификация ресурсов

Классификация 1:

1) **системные** - низкоуровневые, которыми управляет сама операционная система: время работы процессора, оперативная память, память на постоянных носителях, возможности разнообразных внешних устройств и время их работы

2) **пользовательские** - это требования к системе выраженные в терминах объектов или функциональных характеристик прикладной области: файл или таблица, окно для рисования в графической системе, документ в системе печати, мелодия в динамике, запущенное задание, массив памяти и т.д.

Классификация 2:

- первичные: обеспечиваются аппаратными средствами (процессор, память, устройства и каналы ввода-вывода и т.д.)
 - вторичные: порождаемые ОС (системные коды и структуры данных, файлы, семафоры, очереди и т.д.)
-

Классификация ресурсов

В последнее время в связи с развитием распределенных вычислений и распределенного хранения данных все большее значения приобретают такие ресурсы как данные и сообщения.

Классификация 3:

- 1) разделяемые - когда несколько процессов могут их использовать одновременно (в один и тот же момент времени) или параллельно (в течение некоторого интервала времени процессы используют ресурс попеременно)
 - 2) неделимые (используется только одним процессом).
-

Операционная среда - совокупность инструментов, методов их интеграции и приемов работы с ними, позволяющая решать любые задачи в инструментальной области и большинство задач в прикладных областях.

- в операционной среде есть средства решения задач в прикладных областях,
- если инструмента решения какой-то задачи нет, то средствами операционной среды его всегда можно задать.

Прикладная среда (ПС) - это модель окружения операционной системы обеспечивающего предоставления разнообразных интерфейсов.

Прикладная среда создает вокруг базовой операционной системы оболочку, предоставляющую набор необходимых интерфейсов.

В настоящее время понятие ПС дополнено вводом в обиход нового понятия - множественные прикладные среды.

Дисциплина: Операционные системы

§ 6. Понятие процесса ОС

План:

- 6.1. Понятие процесса.
 - 6.2. Классификация процессов
 - 6.3. Состояние и взаимодействие процессов
 - 6.4. Операции над процессом и их планирование
 - 6.5. Управление памятью
-

6.1. Понятие процесса

Процесс – это программный модуль, выполняемый в центральном процессоре.

Действия ОС по управлению процессами:

- Создание и удаление
- Планирование
- Синхронизация
- Коммуникация
- Разрешение тупиковых ситуаций

Программа – план действий
Процесс – действие

Процесс:

**программный код + данные + содержимое стека +
+ содержание регистров**

6.2. Классификация процессов

1. По временным характеристикам:

- *Интерактивные* (время существования интерактивного процесса определяется реакцией ЭВМ на запрос обслуживания и составляет секунды).
- *пакетные* (запускаются один вслед за другим и время реакции часы и более).
- *процессы реального времени* (имеют гарантированное время окончания работы и время реакции мсек).

2. По генеалогическому признаку: порождающие и порожденные процессы.

3. По результативности:

- *эквивалентные* (реализовываются как на одном, так и на многих процессорах по одному или разным алгоритмам, то есть они имеют разные трассы, которые определяют порядок и длительность пребывания процесса в разных состояниях)
- *тождественные* (реализуются по одной и той же программе, но имеют разные трассы).
- *равные процессы* (реализуются по одной программе и имеют одинаковые трассы). Все они имеют одинаковый конечный результат, но эквивалентные процессы могут.

4. По времени развития: последовательные, параллельные и комбинированные (для последних есть точки, в которых существуют оба процесса, и точки, в которых существует только один процесс).

6.2. Классификация процессов

5. По месту развития: *внутренние* (реализуются на центральном процессоре) и *внешние* (реализуются на внешних процессорах).

6. По принадлежности к операционной системе: *системные* (исполняют программу из состава операционной системы) и *пользовательские*.

7. По связности различают процессы:

- *взаимосвязанные*, которые имеют какую-то связь (пространственно-временную, управляющую, информационную);
 - *изолированные* — слабо связанные;
 - *информационно-независимые*, которые используют совместные ресурсы, но имеют собственные информационные базы;
 - *взаимодействующие* — имеют информационные связи и разделяют общие структуры данных;
 - *взаимосвязанные по ресурсам*;
 - *конкурирующие*.
-

6.2. Классификация процессов

Порядок взаимосвязи процессов – отношения между процессами:

- предшествования — один всегда находится в активном состоянии раньше, чем другой;
 - приоритетности — когда процесс может быть переведен в активное состояние только в том случае, если в состоянии готовности нет процессов с более высоким приоритетом, или процессор свободен, или на нем реализуется процесс с меньшим приоритетом;
 - в) взаимного исключения - в процессе используется общий критический ресурс, и процессы не могут развиваться одновременно: если один из них использует критический ресурс, то другой находится в состоянии ожидания.
-

6.3. Состояния процесса

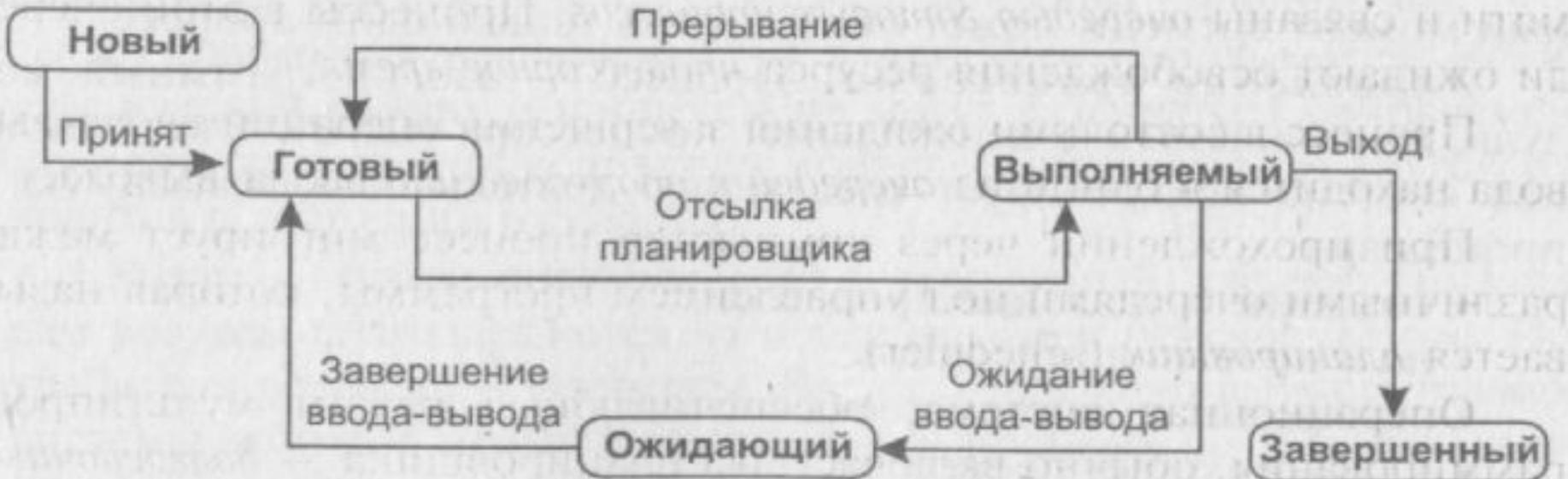
Новый (только что созданный)

Выполняемый (команды программы выполняются в ЦП)

Ожидающий (ожидает завершения какой-либо операции, чаще ввода-вывода)

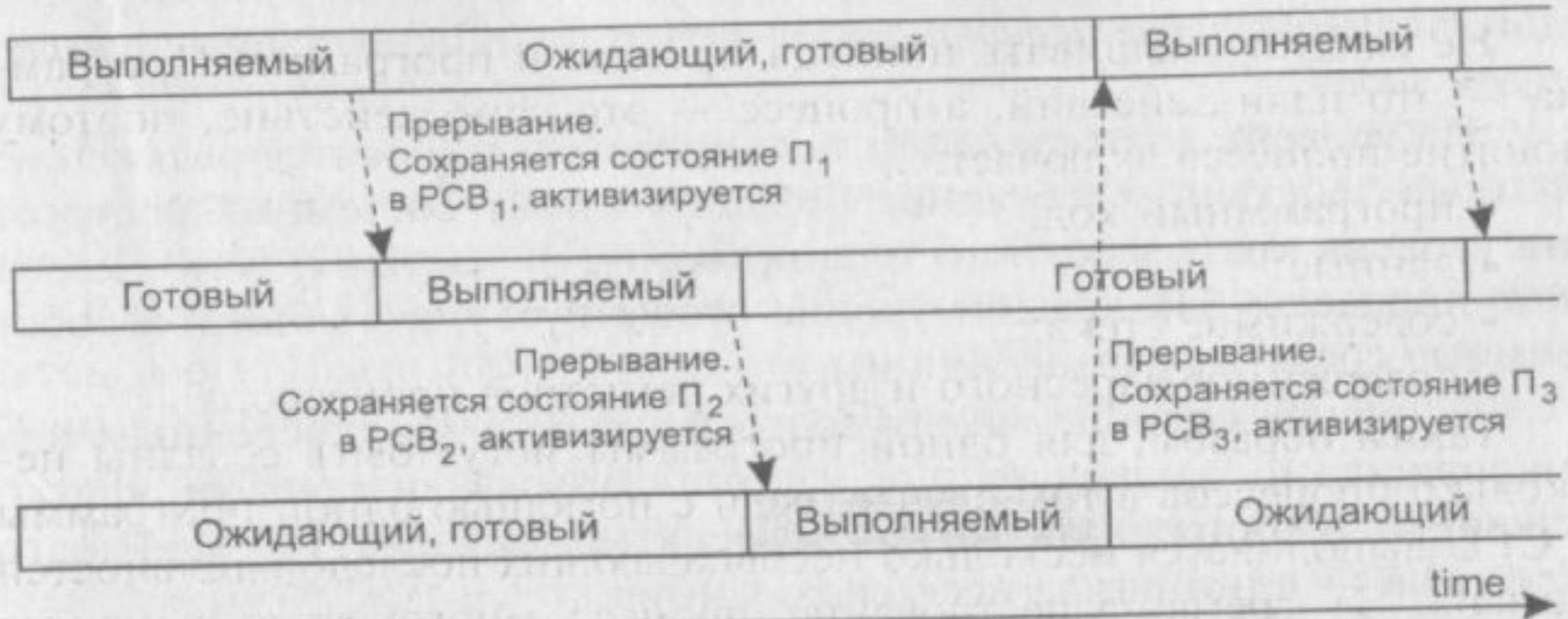
Готовый (ожидает освобождения ЦП)

Завершенный (завершил работу)



6.3. Состояния процесса

Представление процесса в ОС –
таблица управления процессом
(ТУП – PCB – process control blok)



6.3. Взаимодействие процессов

Задание: дайте ответы на вопросы (стр. 46-51, Партыка, Попов):

Вопрос (о планировании)	Ответ
1. Как называется действие: распределение процессов между имеющимися ресурсами?	Планирование процессов
2. Что такое «метод очередей ресурсов»?	Метод планирования процессов, ориентированный на эффективную загрузку ресурсов
3. Где располагаются и чем связаны готовые к выполнению процессы?	Основная память, очередь готовых процессов
4. Как называется программа, управляющая перемещением процесса между очередями?	планировщик
5. Долгосрочный планировщик решает -	Какой из процессов, находящийся во входной очереди, должен быть переведен в очередь готовых процессов в случае освобождения ресурсов памяти
6. Краткосрочный планировщик решает -	Какой из процессов, находящийся во очереди готовых процессов, должен быть передан на управление ЦП
7. В чем заключается основное отличие между разными видами планировщиков	Частота запуска: краткосрочный – каждые 100мс, долгосрочный 1 раз в неск.мин

6.3. Взаимодействие процессов

Вопрос (о взаимодействии)	Ответ
8. В каких отношениях могут находиться выполняемые процессы?	Независимы, взаимодействующие
9. По какой схеме описывают взаимодействие процессов?	«Производитель-потребитель»
10. В чем суть взаимодействия процессов?	Передача данных между процессами или совместное использование ресурсов
11. Перечислите механизмы взаимодействия процессов: ?	Транспортер(канал), очередь, сигнал, семафор
12. Опишите транспортер:	<ul style="list-style-type: none">-средство взаимодействия родственных процессов-область памяти с файловой организацией, для которой обеспечены запись и считывание данных-порядок записи данных неизменен, не допускается повторное считывание-размер задает вызвавший процесс,-дочерние процессы могут использовать родительский транспортер
13. Опишите очередь:	<ul style="list-style-type: none">-обеспечивает передачу или использование общих данных без перемещения данных, с передачей элемента очереди, содержащего указатель данных и объем массива данных-чтение элементов только создающий процесс, остальные – запись в очередь-чтение с уничтожением или без из очереди-доп.функции: определение количества элементов в очереди в текущий момент, очистка очереди созданным процессом

6.3. Взаимодействие процессов

Вопрос (о взаимодействии)	Ответ
14. Опишите сигнал:	<ul style="list-style-type: none">-передача требований одного процесса к другому на немедленное выполнение действия-действия: обработка системной ошибки при появлении сигнала, блокирование сигнала, передача управления подпрограмме
15. Опишите семафор:	<ul style="list-style-type: none">-передача сообщений от одного потока к другому о наступлении некоторого события-виды:<ul style="list-style-type: none">системные (ОС контролирует завершение каждого процесса, владеющего системным семафором)оперативной памяти (устанавливаются в определенное состояние, не обслуживаются ОС, ОС не сообщает об их освобождении или захвате)- Функции для управления семафорами: установка с целью сигнализации, ожидание вызывающим потоком пока не будет выключен, ожидание потоком выключения одного из нескольких семафоров

Пример: интерфейс межпрограммного взаимодействия – буфер обмена

6.4. Планирование работы процессора

Критерии сравнения краткосрочных планировщиков:

1. Утилизация CPU (использование процессора)

0-100% - теоретически

40% - легко загружен

90% - тяжело загружен

2. Пропускная способность CPU (кол-во выполняемых процессов в единицу времени)

3. Время оборота (время выполнения: входная очередь – завершение)

4. Время ожидания (суммарное время нахождения процесса в очереди готовых процессов)

5. Время отклика (время между входной очередью до первого обращения)

6.4. Стратегии планирования процессора

1. Первый пришел – первый обслуживается, FIFO – first come – first served (FCFS)

- велико среднее время ожидания

2.«наиболее короткая работа выполняется первой», SJF - Shortrest Job First

- трудность за ранее определить величину времени последующего обслуживания

3.Приоритетное планирование

- каждому процессу присваивается приоритет , определяющий очередность

Приоритет – целое положительное число, некоторого диапазона

Факторы, влияющие на назначение приоритета:

Внешние:

- Требования к памяти
- Количество открытых файлов
- Отношение среднего времени ввода-вывода к среднему времени использования ресурса CPU

Внутренние:

- Важность процесса
- Тип и величина файлов, используемых для оплаты
- Отделение, выполняющее работы

Недостаток: блокирование на неопределенно долгое время низкоприоритетных процессов

Выход: автоматическое повышение приоритета после некоторого времени ожидания

6.4. Стратегии планирования процессора

4.«карусельная» стратегия планирования (RR-Round Robin)

- Применяется в системах разделения времени
- Пциклическое перемещение процессов, CPU используется в течение 1 кванта времени $t = 10...100\text{мс}$
- Производительность процессора $1/N$

5.Планирование с использованием многоуровневой очереди (Multi-level queue scheduling)

- Для процессов, классифицируемых на группы (интерактивные и пакетные (фоновые))
 - Правило: ни один процесс с более низким приоритетом не может быть запущен, пока не выполняются процессы во всех очередях с более высоким приоритетом
-

6.4. Стратегии планирования процессора

6. Многоуровневая очередь с обратными связями (multilevel feedback queue scheduling) - предполагает, что процессы при определенных условиях могут перемещаться между очередями.

предполагает, что процессы при определенных условиях могут перемещаться между очередями.

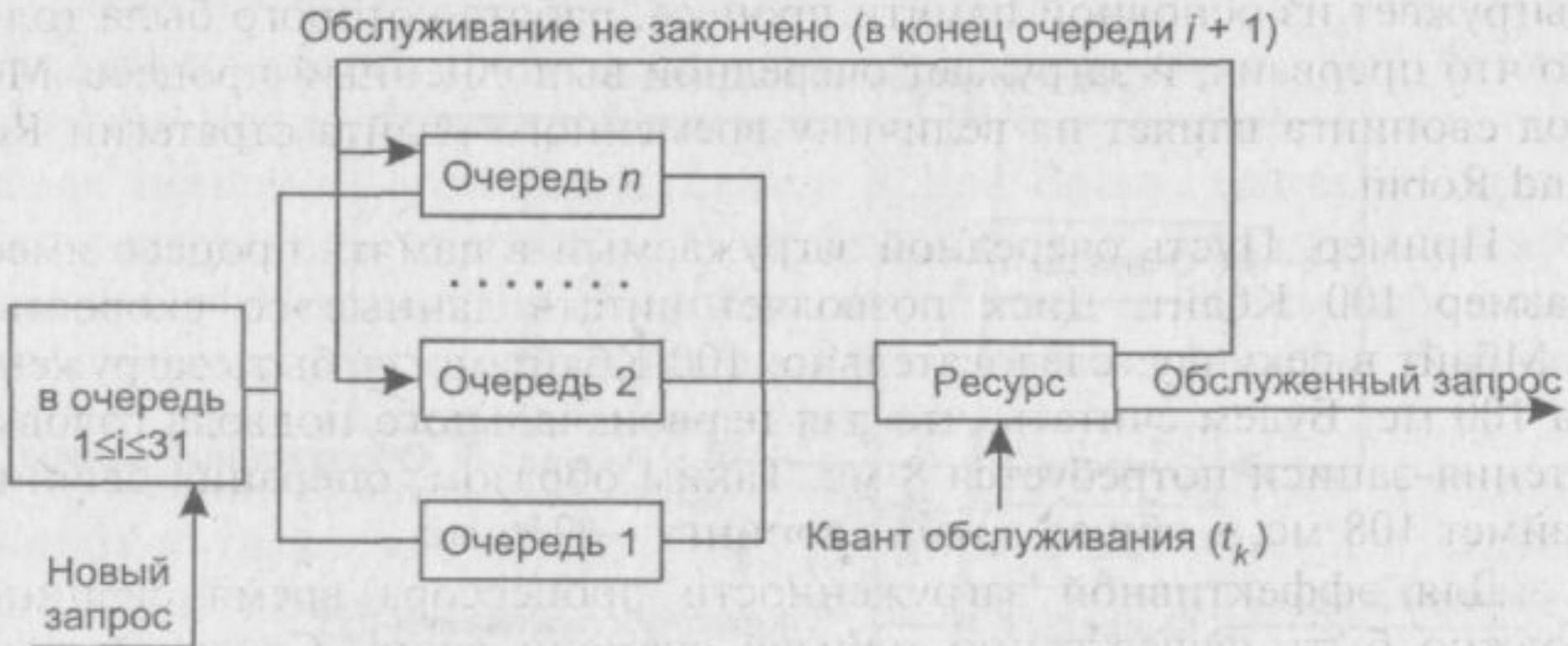
- организуется N очередей.
- все новые запросы поступают в конец первой очереди.
- первый запрос из i -й очереди поступает на обслуживание лишь тогда, когда все очереди от 1-й до $i - 1$ -й пустые.
- на обслуживание выделяется квант времени t_k . Если за это время обслуживание запроса завершается полностью, то он покидает систему. В противном случае недообслуженный запрос поступает в конец $i + 1$ -й очереди
- после обслуживания запроса из i -й очереди система выбирает для обслуживания запрос из непустой очереди с самым младшим номером.



6.4. Стратегии планирования процессора

7. Приоритетная многоочередная дисциплина обслуживания

- вновь поступающие в систему запросы устанавливаются не обязательно в 1-ю очередь, а в очередь в соответствии с имеющимися приоритетами, которые определяются параметрами обслуживания процессов.
- приоритетные многоочередные дисциплины обслуживания могут использовать обслуживание с абсолютным и относительным приоритетом.
- при обслуживании с абсолютным приоритетом приоритет определяется номером



6.5.1. Управление памятью: неvirtуальная

Самостоятельная работа: заполните таблицу (стр. 58-63 Партыка, Попов)

Составляющие метода управления неvirtуальной памятью	Схема	Особенности
метод оверлейных сегментов		
процесс мультипрограммирование с фиксированными разделами		
процесс мультипрограммирование с переменными разделами		
процесс мультипрограммирование с переменными разделами и уплотнением памяти.		

1. Свопинг (swapping, перекачка) - метод управления памятью, основанный на том, что все процессы, участвующие в мультипрограммной обработке, хранятся во внешней памяти.

Процесс, которому выделен CPU, временно *перемещается в основную память* (swar in/roll in). В случае прерывания работы процесса он *перемещается обратно во внешнюю память* (swar out/roll out).

Замечание: при свопинге из основной памяти во внешнюю (и обратно) перемещается вся программа, а не ее отдельная часть.

6.5.1. Управление памятью: неvirtуальная

2. Смежное размещение процессов.

Методы размещения процессов в основной памяти по отношению к расположению участков памяти, выделенных для одной и той же программы, делят на два класса.

2.1. метод смежного размещения - в памяти, начиная с некоторого начального адреса, выделяется один непрерывный участок адресного пространства.

Размер загружаемой программы **ограничивается размером ОЗУ.**

Для того чтобы при смежном размещении загружать программы, размеры которых превышают размеры ОЗУ, используют **метод оверлейных сегментов** (overlay segments).

6.5.1. Управление памятью: неvirtуальная

2.2. метод несмежного размещения - программа разбивается на множество частей, которые располагаются в различных, необязательно смежных участках адресного пространства – реализуется через **страничную организацию памяти** (paging).

Достоинство: позволяет свести к минимуму общую фрагментацию за счет полного устранения внешней фрагментации и минимизации внутренней фрагментации.

Суть:

- адресное пространство основной и внешней памяти разбивают на блоки фиксированного размера, называемые *страничные рамки* (frames).
- логическое адресное пространство программы также разбивается на блоки фиксированного размера, называемые *страницами* (pages).
- размеры страничных рамок и страниц совпадают.
- процесс загружается в память постранично, причем каждая страница помещается в любую свободную страничную рамку основной памяти.
- каждый адрес, генерируемый процессором, состоит из двух частей:

Н — номер страницы (page number)

С — смещение в пределах страницы (offset).

Таблица страниц содержит начальные адреса всех f страничных рамок, в которых размещена программа.

Физический адрес определяется путем сложения начального адреса страничной рамки f и смещения **С**.

6.5.1. Управление памятью: виртуальная

Самостоятельная работа: заполните таблицу (стр. 58-63 Партыка, Попов)

Составляющие метода управления виртуальной памятью	Схема	Особенности
<i>метод управления - перемещение страниц по запросу</i>		
<i>страничная недостаточность</i>		
<i>метод замещение страниц.</i>		
<i>алгоритм распределения страничных рамок</i>	Стр. 68	
<i>алгоритм замещения страниц</i>	Стр. 69	

6.5.2. Управление памятью: виртуальная

Все методы управления памятью имеют одну и ту же цель — хранить в памяти мультипрограммную смесь, необходимую для мультипрограммирования.

Рассмотренные ранее методы предполагали, что вся программа перед выполнением должна быть размещена в основной памяти.

Виртуальная память — это технология, которая позволяет выполнять процесс, который может только частично располагаться в основной памяти и позволяет выполнять программы, размеры которых превышают размеры физического адресного пространства.

Чаще всего реализуется на базе страничной организации памяти, совмещенной со свопингом страниц.

Свопингу подвергаются только те страницы, которые необходимы процессору.

6.5.2. Управление памятью: виртуальная

Методы управления:

1. *Перемещение страниц по запросу (demand paging)* означает:

- программа может выполняться CPU, когда часть страниц находится в основной памяти, а часть — во внешней;
- в процессе выполнения новая страница не перемещается в основную память до тех пор, пока в ней не возникла необходимость.
- для учета распределения страниц между внешней и основной памятью каждая строка таблицы страниц дополняется битом местонахождения страницы (**valid/invalid bit**).
- позволяет начать выполнение процесса даже в том случае, когда ни одна страница этого процесса не загружена в основную память.

2. *Замещение страниц.*

- в основной памяти выбирается наименее важная/используемая страница, называется *страница-жертва* (victim page), которая временно перемещается в пространство свопинга, а на ее место загружается страница, вызываемая страничной недостаточностью.

Алгоритмы для использования метода:

- алгоритм распределения страничных рамок (frame allocation algorithm);
- алгоритм замещения страниц (page replacement algorithm).

Мульти -программирование и -процессорность

Мультипрограммирование, или многозадачность (multitasking), — это способ организации вычислительного процесса, при котором на одном процессоре попеременно выполняются сразу несколько программ.

Совместное использование процессора, оперативной и внешней памяти, устройства ввода-вывода, данные и т.д.

Цель мультипрограммирования - эффективность использования вычислительной системы

Критерии эффективности ВС являются:

- пропускная способность — количество задач, выполняемых вычислительной системой в единицу времени;
- удобство работы пользователей - возможность интерактивно работать одновременно с несколькими приложениями на одной машине;
- реактивность системы — способность системы выдерживать заранее заданные (возможно, очень короткие) интервалы времени между запуском программы и получением результата.

Мульти -программирование и -процессорность

Мультипроцессорная обработка — это способ организации вычислительного процесса в системах с несколькими процессорами, при котором несколько задач (процессов, потоков) могут одновременно выполняться на разных процессорах системы.

В зависимости от выбранного критерия эффективности ОС делятся на:

- системы пакетной обработки,
 - системы разделения времени,
 - системы реального времени.
-

Мульти -программирование и -процессорность

Самостоятельная работа. Заполните таблицу:

Виды ОС	Мультипрограммирование					
	1	2	3	4	5	6
системы пакетной обработки						
системы разделения времени						
системы реального времени.						