11. 메모리 관리

1. 페이지와 세그먼트

기억공간 크기를 나누는 방식에 따라 다음과 같은 용어를 사용한다.

페이지	블록의 크기를 **동일하게 분할**한 경우에 '페이지'라 한다.**(고정분할)**
세그먼트	블록의 크기를 **서로 다르게 분할**한 경우에 '세그먼트'라 한다.**(가변분할)**

① 현재 대부분의 다중 프로그래밍 운영체제는 가상메모리 기법으로 구현된다.
② 가상메모리 구현은 **페이징** 또는 **세그먼테이션** 기법을 이용하여 구현한다.

2. 단편화(fragmentation)

내부 단편화	• 프로세스가 분할된 특정 메모리 빈 영역에 적재될 때, 남은 자투리 공간이다.
외부 단편화	• 프로세스가 적재될 수 없는 경우이다. • 공백 영역이 있어도 크기가 너무 작으면, 큰 프로세스는 적재될 수 없다. • 이때 발생되는 단편화를 외부단편화라 한다.

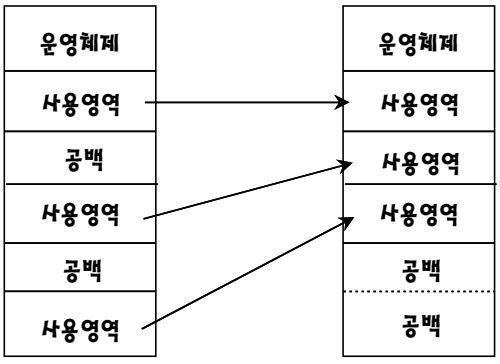
운영체제
사용영역
공백(50k)
공백(70k)

← 크기 **30**k인 프로세스 적재(내부단편화 발생 : 50k − 30k = 20k)

← 크기 **90**k인 프로세스는 적재 불가**(외부단편화 발생 : 70k)**

3. 압축(집약; compaction)

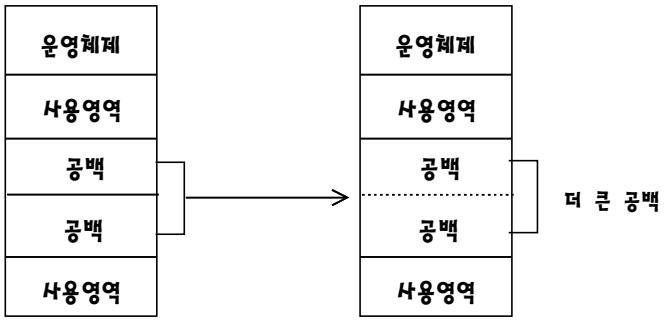
압축은 비어있는 분할 영역을 **한 곳으로 합치는 작업**이다.
수행중인 프로그램은 재배치되어야 하며, 압축되는 동안 시스템은 다른 작업을 할 수 없다.



공백을 **자유영역**이라고도 한다.

4. 통합(coalescing)

비어있는 영역이 인접하게 되면 **두 영역을 하나로 통합**하여 더 큰 영역으로 만든다.



• 외부단편화를 해결하기 위해서는 **'압축이나 통합'**이 필요하다.

5. 기억장치 배치 전략(placement strategy)

새로 반입되는 프로그램을 '메모리 빈 영역 어디에 배치할 것인가?'를 결정하는 것이다.

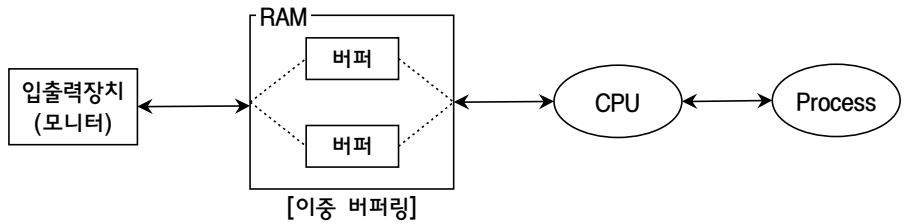
• 최초적합(first fit) : 수용 가능하면 첫 번째 빈 공간에 무조건 배치
• 최적적합(best fit) : 배치 후에 빈 공간이 가장 적게 남는 영역에 배치
• 최악적합(worst fit) : 가장 큰 빈 공간에 배치

[예] 새로 반입되는 프로그램의 크기가 20k인 경우

운영체제	
공백(30k)	→ **최초적합**(내부단편화 : 30k − 20k = 10k)
사용영역	
공백(70k)	→ **최악적합**(내부단편화 : 70k − 20k = 50k)
사용영역	
공백(15k)	
사용영역	
공백(25k)	→ **최적적합**(내부단편화 : 25k − 20k = 5k)

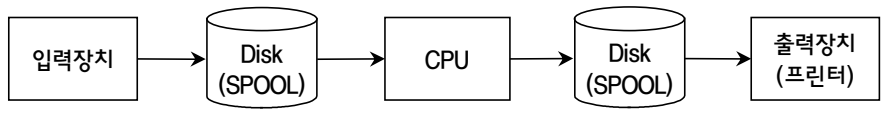
6. 버퍼링(buffering)

다중 프로그래밍에서 버퍼링은 운영체제의 효율성과 각 프로세스의 원활한 활동을 유지하게 하는 중요한 요소가 된다.



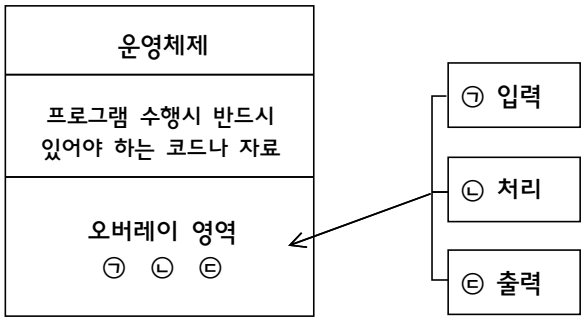
- 버퍼링은 하나의 프로그램 내에서 입출력과 자신의 연산처리를 중복시키는 방법이다.
  → CPU가 이전에 읽은 자료를 처리할 때, 입출력장치는 다음 자료를 입출력한다.
- 버퍼링은 **단일 사용자 방식**이다.
- 버퍼링은 **하드웨어 방식**이다. (전용 장치 버퍼)
- RAM을 제조할 때 RAM의 일부**(8M, 16M 등)**를 Buffer 공간으로 배정한다.
- 버퍼 수는 시스템에 따라 다르다.(2중 버퍼링, 3중 버퍼링 등)
- 버퍼링 사용 예 : **Streaming 방식의 동영상 파일 처리, 인터넷 방송** 등

7. SPOOLing(Simultaneous Peripheral Operation On Line)



- 스풀링은 어떤 작업의 입출력과 다른 작업의 연산을 병행(중복) 처리하는 기법이다.
- 스풀링은 **프린터** 처리 속도 및 프린터 공유를 보완하기 위해 많이 사용된다.
- 스풀링은 다양한 입출력 작업을 디스크 같은 대용량의 저장매체에 보관해 두었다가 편리한 시점에 처리될 수 있도록 하는 기법이다.
- 스풀링은 **디스크**의 일부분을 버퍼처럼 사용하는 것이다.
- 스풀링은 **다중 사용자** 방식이다.
- 스풀링(디스크)은 버퍼링(RAM)에 비해 보다 많은 입출력 작업을 임시 보관할 수 있다.
- 스풀링은 **소프트웨어 방식**이다. - 운영체제의 스풀러(spooler)에 의한 입출력 작업 관리

8. 오버레이(overlay)



① 주기억장치 용량보다 **더 큰 프로그램을 실행**시키기 위한 기법이다.
→ 프로그램을 적절한 크기로 분할하여 현재 작업 수행에 꼭 필요한 것만 적재시킨다.

② 오버레이 영역에 '㉠ ㉡ ㉢' 중에서 어느 하나가 필요에 따라 적재되고,
당장 필요 없는 부분은 디스크에 보관해 둔다.

③ 가상메모리 구현에서는 오버레이는 저절로 해결된다.

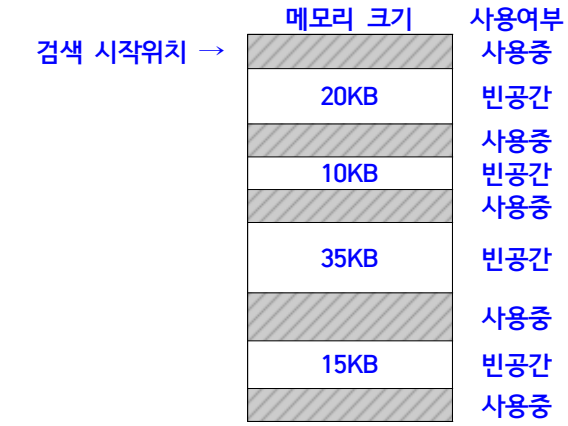
9. 페이징(paging) 기법

순수 페이징 (pure paging)	① 주기억장치를 균등 크기의 프레임으로 분할하고, 적재될 프로세스도 프레임과 같은 크기의 페이지로 균등하게 분할한다. ② 프로세스의 **모든 페이지를 주기억장치에 적재**시킨 후에 실행하는 방식이다.
요구 페이징 (demand paging)	① 프로세스 수행 도중에 필요한(요구) 페이지가 있을 때, **요구된 페이지를 주기억장치에 적재**한다. ② 다중 프로그래밍 정도를 높일 수 있다. ③ 복잡한 메모리 관리를 해야 하므로 오버헤드가 크다.

• 요구 페이징 기법은 **가상메모리**(virtual memory) 구현에 사용된다.

기출문제 분석

1. 크기가 각각 12KB, 30KB, 20KB인 프로세스가 다음과 같은 메모리 공간에 순차적으로 적재 요청될 때, 모든 프로세스를 적재할 수 있는 알고리즘만을 모두 고른 것은? [2016년 지방 9급]

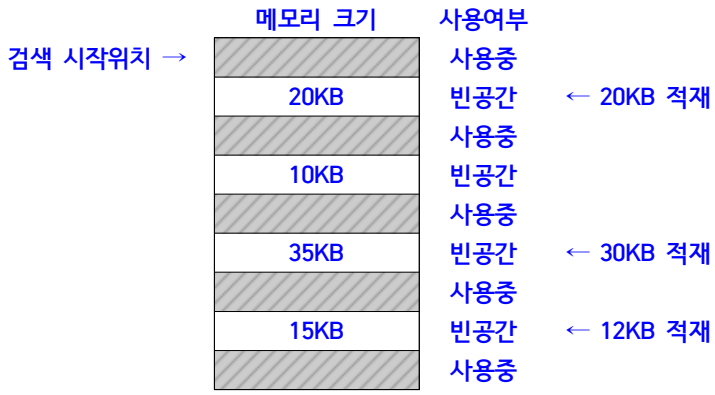


ㄱ. 최초 적합(first-fit) / ㄴ. 최적 적합(best-fit) / ㄷ. 최악 적합(worst-fit)

① ㄱ ② ㄴ ③ ㄱ, ㄴ ④ ㄴ, ㄷ

♤ **메모리 적재**

• 최적 적합(best-fit)



• 최초 적합(first-fit)은 20KB가 적재될 수 없다.
• 최악 적합(worst-fit)은 30KB가 적재될 수 없다.

정답 : ②

2. 다음과 같은 가용 공간을 갖는 주기억장치에 크기가 각각 25KB, 30KB, 15KB, 10KB인 프로세스가 순차적으로 적재 요청된다. 최악적합(worst-fit) 배치전략을 사용할 경우 할당되는 가용 공간 시작주소를 순서대로 나열한 것은? [2017년 지방 9급]

가용 공간 리스트	
시작주소	크기
w	30KB
x	20KB
y	15KB
z	35KB

① w→x→y→z ② x→y→z→w
③ y→z→w→x ④ z→w→x→y

♤ 최악적합(worst-fit) 배치전략

시작주소	크기	최악적합(worst-fit) 배치전략
w	30KB	30KB
x	20KB	15KB
y	15KB	10KB
z	35KB	25KB

• 25KB→30KB→15KB→10KB의 시작주소 : z→w→x→y

정답 : ④

3. 기억장치의 동적분할에서 발생하는 단편화 문제를 해결하기 위한 운영체제의 기법은? [2008년 경찰]

① 압축 ② 체이닝
③ 스풀링 ④ 동기화

♤ 단편화 해결 방안

• 외부단편화를 해결하기 위해서는 '압축이나 통합'이 필요하다.

정답 : ①

4. 메모리 관리 방안에 대한 설명으로 옳은 것만을 모두 고르면? [2022년 국회 9급]

ㄱ. Worst-fit 할당은 가장 큰 공간에 프로세스를 배치하는 방식이다.
ㄴ. Best-fit 할당 방식은 요청하는 메모리 크기에 가장 일치하는 크기의 메모리 블록을 할당함으로써 외부단편화를 최소화 할 수 있다.
ㄷ. First-fit 할당 방식은 탐색 시 요청한 메모리 크기를 할당할 수 있는 메모리 블록을 찾았을 때 바로 할당함으로써 탐색시간을 줄일 수 있다.
ㄹ. Buddy 할당 방식은 메모리를 2의 거듭제곱 단위의 크기로 할당하여 내부단편화를 최소화 할 수 있다.

---

① ㄱ, ㄷ ② ㄱ, ㄹ ③ ㄴ, ㄹ
④ ㄱ, ㄴ, ㄷ ⑤ ㄴ, ㄷ, ㄹ

♤ **메모리 관리 방안**

ㄴ. Best-fit 할당 방식은 요청하는 메모리 크기에 가장 일치하는 크기의 메모리 블록을 할당함으로써 **외부단편화**를 최소화 할 수 있다.(×)
→ Best-fit 할당 방식은 **내부단편화**를 최소화 할 수 있다.

ㄹ. Buddy 할당 방식은 메모리를 **2의 거듭제곱** 단위의 크기로 할당하여 내부단편화를 최소화할 수 있다.(×)
→ Buddy 할당 방식은 메모리를 **절반** 크기로 분할하면서 최적 할당한다.(내부단편화 최소화)

// 버디 시스템(buddy system) – 버디 메모리 할당(buddy memory allocation)

- 먼저, 영어단어 buddy는 친구, 단짝을 의미한다.
- 버디 시스템은 큰 버퍼들을 반복적으로 **반으로 나누어** 작은 버퍼들을 만든다.
- 메모리 요청에서 나누어진 작은 버퍼 메모리에 최적으로 할당하는 알고리즘이다.
- 즉, 버디 시스템은 메모리 크기를 절반씩 분할하면서 가장 잘 맞는 크기의 메모리를 찾는다.

[예] 버디 시스템에서 1024KB 메모리에 200KB의 프로그램 코드 할당

<table>
<tr><td colspan="8">1024KB</td><td>← 전체 메모리 크기</td></tr>
<tr><td colspan="4">512KB</td><td colspan="4">512KB</td><td>← 1번 분할된 버디</td></tr>
<tr><td colspan="2">256KB</td><td colspan="2">256KB</td><td colspan="2">256KB</td><td colspan="2">256KB</td><td>← 2번 분할된 버디</td></tr>
<tr><td>128KB</td><td>128KB</td><td>128KB</td><td>128KB</td><td>128KB</td><td>128KB</td><td>128KB</td><td>128KB</td><td>← 3번 분할된 버디</td></tr>
</table>

- 200KB의 프로그램 코드는 2번 분할 후에 할당한다.
- 200KB의 프로그램 코드는 3번 분할 후에는 할당이 불가능하므로(128KB〈200KB〈256KB)

정답 : ①

5. 버퍼링(buffering)과 스풀링(spooling)에 대한 설명으로 틀린 것은 모두 몇 개인가? [2010년 경찰 9급]

> **㉠ 버퍼링과 스풀링 모두 저속의 입출력장치와 고속의 CPU장치의 속도 차이를 개선하기 위한 방법이다.**
> **㉡ 버퍼링은 보조기억장치를, 스풀링은 주기억장치를 버퍼로 사용한다.**
> **㉢ 버퍼링은 소프트웨어적 구현 방법이고, 스풀링은 하드웨어적 구현 방법이다.**
> **㉣ 버퍼링은 단일사용자 시스템에 사용되고, 스풀링은 다중사용자 시스템에 사용된다.**

① 1개 ② 2개
③ 3개 ④ 4개

♤ **버퍼링과 스풀링**

㉡ 버퍼링은 보조기억장치를, 스풀링은 주기억장치를 버퍼로 사용한다.
→ 버퍼링은 주기억장치, 스풀링은 보조기억장치를 버퍼로 사용한다.
㉢ 버퍼링은 소프트웨어적 구현 방법이고, 스풀링은 하드웨어적 구현 방법이다.
→ 버퍼링은 하드웨어 구현 방법이고, 스풀링은 소프트웨어적 구현 방법이다.

정답 : ②

6. 버퍼링(buffering)과 스풀링(spooling)에 대한 설명으로 옳지 않은 것은? [2011년 국회]

① 버퍼링은 저속의 입출력 장치와 고속의 CPU 간의 속도 차를 해소하기 위해서 나온 방법이다.
② 스풀링은 어떤 작업의 입출력과 다른 작업의 계산을 병행 처리하는 기법이다.
③ 버퍼링은 한 레코드를 읽어서 CPU가 그것에 대한 작업을 시작함과 동시에 입출력 장치가 필요한 레코드를 미리 읽어 버퍼에 저장해 둔다.
④ 스풀링은 디스크 일부를 매우 큰 버퍼처럼 사용하는 방법이다.
⑤ 버퍼링은 보조기억장치를 버퍼로 사용한다.

♤ **버퍼링과 스풀링**

• 버퍼링은 주기억장치를 버퍼로 사용한다.

정답 : ⑤