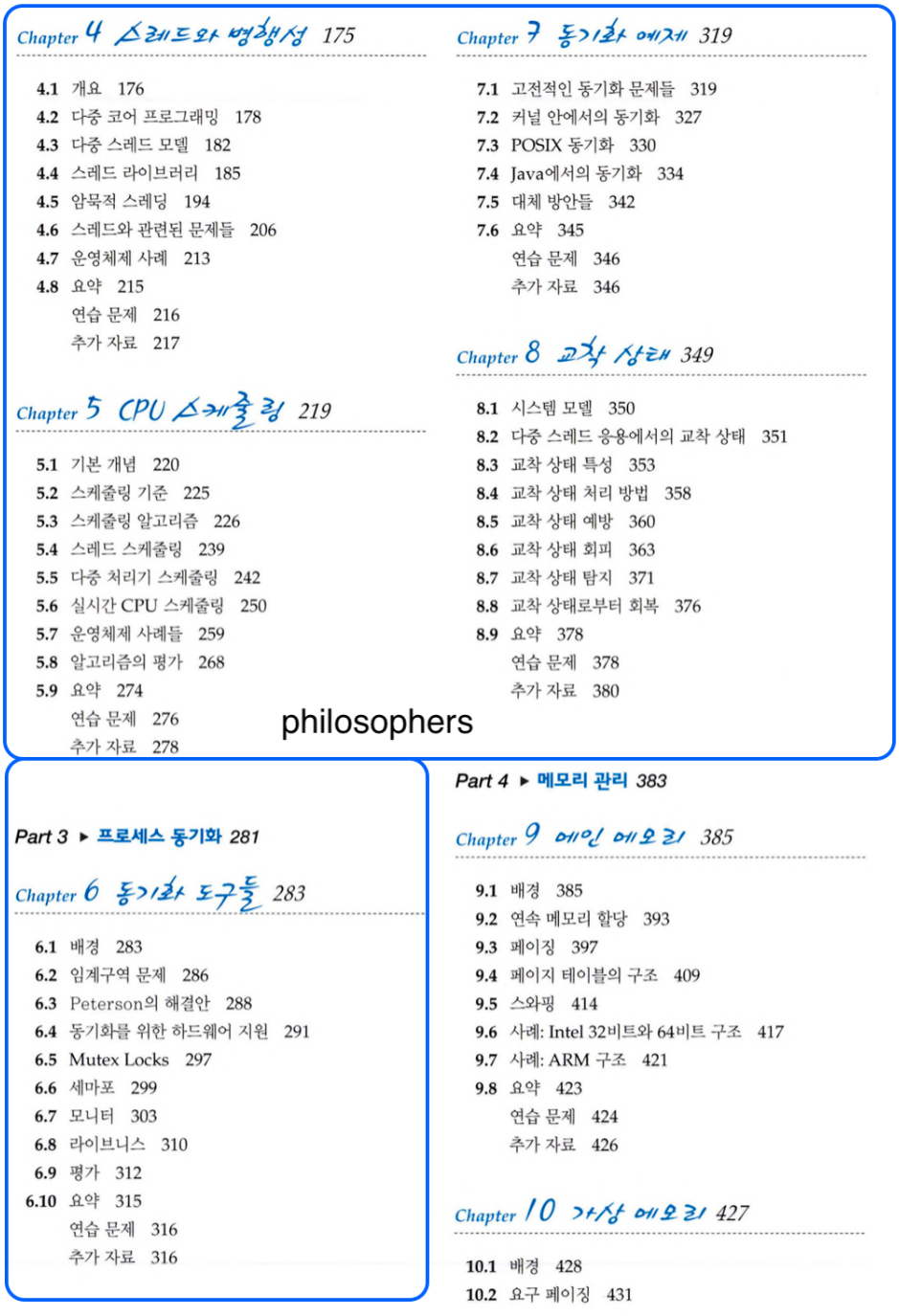
# Contents

# Chapter 1

## 주제 선정 이유

운영체제 도서(Operating System Concepts, 일명 공룡책)의 목차를 통해 운영체제 전반에서 Inner Circle에서 명시적으로 다루지 않은 주제를 찾았다.



Part 7 ▸ 보안과 보호 679

Chapter 16 보안 681

16.1 보안 문제 681
16.2 프로그램 위협 685
16.3 시스템과 네트워크 위협 696
16.4 보안 도구로서 암호 기법 699
16.5 사용자 인증 712
16.6 보안 방어의 구현 717
16.7 예: Windows 10 727
16.8 요약 730
추가 자료 731

Born2beroot

Chapter 17 보호 733

17.1 보호의 목표 733
17.2 보호의 원칙 734
17.3 보호 링 735
17.4 보호의 영역 738
17.5 접근 행렬 742
17.6 접근 행렬의 구현 746
17.7 접근 권한의 취소 749
17.8 역할 기반 액세스 제어 751
17.9 강제적 접근 제어 752
17.10 자격-기반 시스템 753
17.11 기타 보호 개선 방법 755
17.12 언어 기반의 보호 758
17.13 요약 764
추가 자료 765

Part 8 ▸ 진보된 주제 767

Chapter 18 가상 머신 769

18.1 개요 769
18.2 역사 771
18.3 장점 및 특징 772
18.4 빌딩 블록 775
18.5 VM 유형 및 구현 781
18.6 가상화와 운영체제 구성요소 789
18.7 사례 796
18.8 가상화 연구 798
18.9 요약 799
추가 자료 801

Chapter 19 네트워크 및 분산 시스템 803

19.1 분산 시스템의 장점 803
19.2 네트워크 구조 805
19.3 통신 구조 808
19.4 네트워크 및 분산 운영체제 820
19.5 분산 시스템의 설계 문제 824
19.6 분산 파일 시스템 829
19.7 DFS 명명 및 투명성 833
19.8 원격 파일 액세스 836
19.9 분산 파일 시스템에 대한 최종 생각 840
19.10 요약 841
연습 문제 842
추가 자료 843

Netpractice

minishell

Part 9 ▸ 사례 검토 845

Chapter 20 Linux 시스템 847

20.1 Linux 역사 847
20.2 설계 원칙 853
20.3 커널 모듈 856
20.4 프로세스 관리 860
20.5 스케줄링 864
20.6 메모리 관리 870
20.7 파일 시스템 880
20.8 입/출력 887
20.9 프로세스 간 통신 890
20.10 네트워크 구조 891
20.11 보안 894
20.12 요약 897
연습문제 898
추가 자료 898

Chapter 21 윈도우 10 901

21.1 역사 901
21.2 설계 원칙 906
21.3 시스템 구성요소 920
21.4 터미널 서비스와 빠른 사용자 교체 960
21.5 파일 시스템 961
21.6 네트워킹 968
21.7 프로그래머 인터페이스 973

'Part 4.메모리 관리'가 언급되지 않은 것으로 확인되었고, 그 중에서도 dash의 취지에 맞는 서브젝트를 구상할 수 있는 페이지 교체 알고리즘 구현을 주제로 선정했다.

# Chapter II

## 학습하고자 하는 개념

첫째, 페이징 메모리 관리법
둘째, Page Fault
셋째, 페이지 교체 알고리즘

# 1. 페이징 메모리 관리법

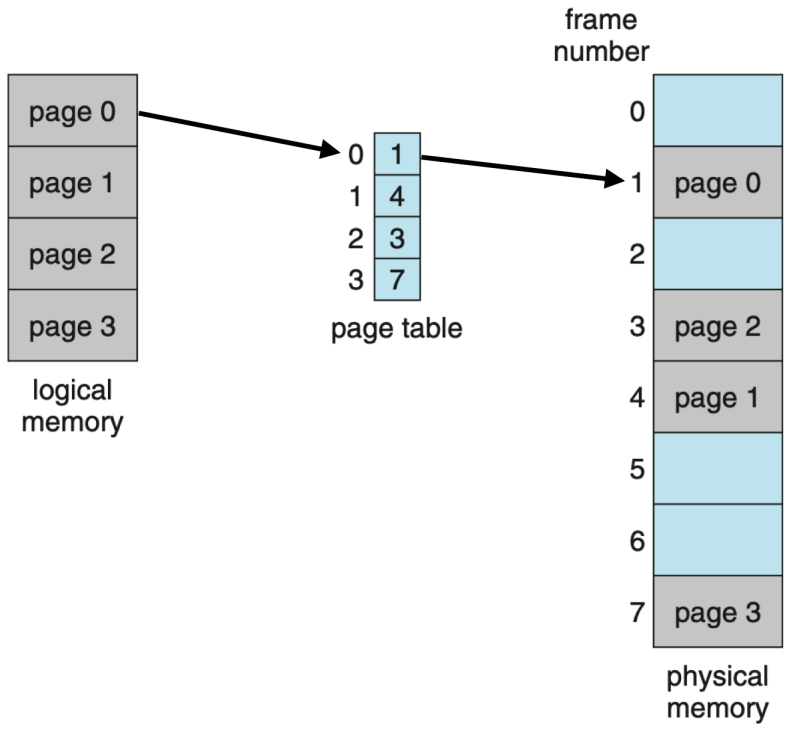



**Figure 9.9** Paging model of logical and physical memory.

그림을 통해 페이징 모델을 설명하고자 한다. 논리 메모리의 단위가 page이며 물리 메모리(메인 메모리)는 frame이라는 단위로 나뉜다. 현재 하나의 프로세스가 논리 메모리에 저장되어있고 메인 메모리에 올라올 때 연속된 물리 메모리를 할당받지 못할 수 있다. 이때 페이징이 page table을 참고하여 논리 메모리의 주소를 물리 메모리의 주소로 변환하여 물리 메모리에 흩어진 페이지들에 접근할 수 있도록 한다. page table은 위와 같이 frame number를 갖고 있으며 이 frame number들의 배열을 page frame이라고 한다.

페이징으로 메모리를 관리하면 필요한 페이지만 메인 메모리에 로드할 수 있다. 하지만 페이징이 물리 메모리 주소를 찾을 수 있는 페이지 테이블에서 찾을 수 없는 페이지에 접근하는 경우 page fault가 발생한다. page fault는 메모리의 용량이 고정된 이상 발생할 수밖에 없으며 이를 해결하기 위한 방법 또한 존재한다.

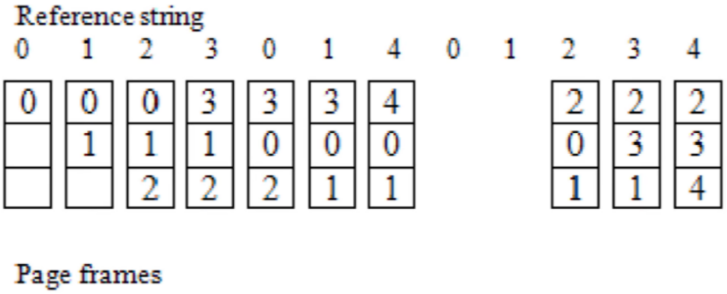
## 둘째, Page Fault

page fault가 발생하면 해당 페이지를 페이지 테이블에 적재하기 위해 희생자(victim)프레임을 선택해야한다. 이때 페이지 교체 알고리즘이 동작하여 희생자 프레임을 빼고 새로운 페이지를 넣는다. 알고리즘마다 각기 다른 최선의 희생자 프레임을 선택한다.

## 셋째, 페이지 교체 알고리즘

페이지 교체 동작이 어떤 방식으로 이루어지는 지 이해하기 위해 총 4가지의 알고리즘을 간단하게 설명한다.

가. FIFO(First Input First Out)




희생자 프레임 : 메모리에 올라온지 가장 오래된 페이지

나. LRU(Least Recently Used)

희생자 프레임 : 가장 오랫동안 사용되지 않은 페이지

다. LFU(Least Frequently Used)

희생자 프레임 : 참조 횟수가 가장 작은 페이지

라. OPT(Optional Page Replacement)

희생자 프레임 : 앞으로 가장 오랫동안 사용되지 않을 페이지

이때, 알고리즘의 성격이 다른 하나가 존재한다. 바로 OPT알고리즘이다. 다른 알고리즘들은 과거의 정보를 통해 희생자 프레임을 선택하는데 OPT는 이후 있을 작업을 미리 알고 있다. 이는 가장 낮은 page fault율을 보장한다. 하지만 프로세스가 앞으로 어떤 메모리를 참조할지 예측하기 어렵기 때문에 OPT는 실제로 구현되기 어려우며, 최적의 알고리즘으로써 다른 페이지 교체 알고리즘의 기준이된다.

# Chapter III

## 서브젝트 가이드라인

서브젝트 상단에 위키피디아 주소를 첨부하였고 특히 해당 해설지에서는 알고리즘까지 설명하였다.

Mandatory는 위에서 공부한 지식을 바탕으로 최적의 페이지 교체 알고리즘을 구현한다.

입력과 출력을 서브젝트에 '맞춰' 구현하고 제공된 corrector 실행파일을 통해 자신의 알고리즘이 최적의 알고리즘인지 확인한다. corrrector는 출력값을 비교하여 OK / KO의 결과를 보여준다.

최적의 알고리즘이 과연 실제 컴퓨터에서 최적으로 작동하는가? 이에 대한 답변은 학습을 통해 깨닫길 바란다.

Bonus는 학습을 위한 확장 버전이므로 시도해보면 좋다. 입력을 넣을 때마다 페이지 프레임 상태를 출력하여 미래에 어떤 페이지가 참조될지 모르는 상황에서의 페이지 교체 알고리즘을 구현한다. Mandatory와 달리 체커가 존재하지 않는다.

Mandatory와 Bonus학습을 통해 최적의 페이지 교체 알고리즘과 실제로 사용되는 페이지 교체 알고리즘을 구현할 수 있다.

테스터기 사용법 안내

1) vpdlwldkfrhflwma 실행파일을 만든다.
2) 해당 실행파일을 주어진 corrector 프로그램과 test_cake.sh 쉘 스크립트와 같은 디렉토리에 둔다.
3) test_cake.sh 스크립트를 실행시킨다.

# Chapter V

## 마치며

우선 42서울 에듀톤의 개최를 추진한 모든 관계자 분들에게 감사의 인사를 전합니다. 1차 발표 전에 dash의 해설지를 만드는 막바지에 이르렀습니다. 긴 시간동안 같은 공간에서 함께 카뎃들이 있었기에 같이 열정을 불태울 수 있었으므로 또한 감사를 전합니다.

시작 단계부터 익숙하지 않은 일이었기에 부족한 점도 많음을 느꼈습니다. 그러므로 이 해설지를 읽고 평가에 참여해주는 모든 분들의 피드백을 깊이 받아들일 것입니다.

해설지가 이해되지 않는 부분이나 서브젝트에 대한 질문은 언제나 환영입니다. 에듀톤의 마지막 1분까지 다들 힘내시길 바랍니다!