2023년 2학기 스터디그룹 주간학습보고서

그룹 명	MaveOS		
날 짜	10월 11일	**시 간**	12시 00분 ~ 14시 45분
스터디장소	50주년 기념관 516호	**회차**	4 회차 모임
수강과목	**운영체제**	**담당교수**	강지훈
스터디 참석자	**리더: 한아림**		
	그룹원: 손효림, 이경화		
학습문제			

<CPU 스케줄링>

- 기본 개념
- 스케줄링 기준
- 스케줄링 알고리즘
- 스레드 스케줄링
- 다중 프로세스 스케줄링
- 실시간 CPU 스케줄링
- 운영체제의 예
- 알고리즘 평가

학습문제 해결과정

<CPU 스케줄링>

1. 프로세스 간 CPU 전환
    - 단위 : 프로세스가 아닌 커널 수준 스레드
    - 스케줄링 개념 : 프로세스 스케줄링
    - thread-specific 아이디어 : 스레드 스케줄링

2. 단일 코어 시스템 vs 멀티 코어 시스템
    - 단일 코어 시스템 : 동시에 하나의 프로세스만 실행. 다른 프로세스는 CPU 코어가 비워지고 재스케줄링 될 때까지 대기
    - 멀티 코어 시스템 : CPU를 항상 바쁘게 유지하는 컨셉을 모든 코어로 확장

3. 멀티 프로그래밍의 목적
    - 스케줄링은 OS 설계의 핵심이자 근본적인 기능
    - CPU 사용률을 최대화하기 위해 프로세스를 항상 실행
    - 프로세스는 I/O 요청이 완료될 때까지 대기(대기 기간동안 CPU가 유휴 상태가 되는 자원 낭비 발생)
    - 대기시간을 생산적으로 활용(동시에 여러 프로세스를 메모리에 유지. 하나의 프로세스가 대기할 때마다 다른 프로세스가 CPU 대신 사용)

4. CPU-I/O Burst Cycle
    - 프로세스 실행은 CPU 버스트로 시작되고 뒤이어 I/O 버스트가 발생
    - 사이클이 반복되고 마지막 실행 종료를 위한 시스템 콜과 함께 끝남

5. CPU 버스트
    - 짧은 버스트가 많고 긴 버스트는 적다
    - I/O 집약적 프로그램 : 짧은 버스트를 많이 가짐
    - CPU 집약적 프로그램 : 긴 버스트를 가질 수 있음

6. CPU 스케줄러
    - CPU가 유휴상태가 될 때마다 OS는 ready queue에 있는 프로세스 중 하나를 선택
    - 스케줄러는 실행 준비된 메모리 내 프로세스 중 하나를 선택해 CPU 할당
    - ready queue는 FIFO, 우선순위 큐, 트리, 단순 연결 리스트로 구현(큐에 있는 레코드들은 일반적으로 프로세스의 PCB)

7. 선점 스케줄링 vs 비선점 스케줄링
    - 선점 스케줄링
        > 프로세스를 어떻게 처리할 지 선택의 여지가 있음
        > 사용 중인 프로세스를 중지시키고 CPU를 해제할 수 있음
        > 현재 OS 대부분이 사용하는 알고리즘
        > 공유 데이터에 대한 액세스 선점 과정에서 경쟁 상태를 초래하거나 일관성을 깨뜨릴 수 있음
        > 운영체제 커널에 영향을 미침
        > 프로세스가 대기 상태에서 준비 완료 상태로 전환될 때, 프로세스가 실행 상태에서 준비 완료 상태로 전환될 때 스위치
    - 비선점 스케줄링
        > 새로운 프로세스가 반드시 선택되어야 함
        > CPU가 할당되면 프로세스가 종료되거나 자기 스스로 CPU를 방출할 때까지 대기
        > 프로세스가 실행 상태에서 대기 상태로 전환될 때, 프로세스가 종료될 때 스위치

8. 인터럽트
    - 인터럽트에 영향을 받는 코드 부분은 반드시 동시 사용으로부터 보호되어야 함
    - 인터럽트 불능화는 자주 발생해서는 안되고 아주 적은 수의 명령어들만 포함해야 함

9. Dispatcher
- 스케줄러가 선택한 프로세스에게 CPU 코어의 제어를 넘겨주는 모듈
- 모든 프로세스의 컨텍스트 스위칭 시 호출되기 때문에 빠르게 수행되어야 함
- dispatch latency: 프로세스를 정지하고 다른 프로세스 수행 시작까지 소요되는 시간

10.디스패처 기능
- 한 프로세스에서 다른 프로세스로 컨텍스트 스위칭
- 사용자 모드로 전환
- 프로그램을 다시 시작하기 위해 사용자 프로그램의 적절한 위치로 이동하는 일

11.자발적 컨텍스트 스위칭 vs 비자발적 컨텍스트 스위칭
- 자발적 컨텍스트 스위칭 : 프로세스가 현대 사용 불가능한 자원을 요청하여 프로세스가 CPU 제어권을 포기한 경우
- 비자발적 컨텍스트 스위칭 : 타임 슬라이스가 만료되었거나 우선순위가 더 높은 프로세스에 의해 선점된 경우

12.스케줄링 기준
- (스케줄링 알고리즘 특성을 고려) CPU 사용량 최대화 I 처리량 최대화 I 처리 시간 최소화 I 대기 시간 최소화 I 응답 시간 최소화

13.스케줄링 알고리즘
- 레디 큐에 있는 프로세스 중 어떤 프로세스에서 CPU를 할당할지 결정

14.FCFS(First-Come, First-Saved Scheduling)
- 가장 간단한 CPU 스케줄링 알고리즘
- CPU 요청을 먼저 요청한 프로세스에게 CPU 먼저 할당
- FIFO 레디 큐로 관리
- 비선점 스케줄링 -> 대화형 시스템에서 비효율적 작동
- convoy effect: 하나의 긴 프로세스가 CPU 양도하길 기다림 -> 장치 이용률 저하

15.SJF(Shortest Job First Scheduling)
- CPU 이용이 가능해지면 다음 CPU 버스트가 가장 짧은 프로세스에게 CPU 할당
- 동일한 CPU 버스트 길이를 가진 프로세스가 여러 개면 FCFS 사용
- 짧은 프로세스를 긴 프로세스보다 먼저 스케줄링하면 평균 대기 시간 감소
- 다음 CPU 버스트 시간을 알 수 없지만 이전 길이 평균으로 근사값 예측
- 선점 또는 비선점 스케줄링이 될 수 있음

학습성찰	
학습내용 이해도	**93** %
학습활동 돌아보기 **(좋았던 점, 아쉬운 점** **활동모습 사진 추가)**	- 한아림 이번 학습에서는 컴퓨터의 핵심 사항인 CPU의 스케줄링에 대한 내용을 학습하였다. CPU 스케줄링은 다중 프로그래밍 기능을 지원하는 운영체제의 기초가 되는 요소로, 현대 시대에서 사용하는 운영체제의 작동 기제에 있어 핵심 기능이라는 것이 인상깊었다. 이번 주차 학습을 진행하기 전 CPU 코어와 다른 프로세스 간의 대략적인 내용만 이해했을 뿐 자세한 내용은 알지 못하였는데, 이번 학습을 진행한 후에는 단일 코어 시스템과 다중 코에 시스템에서의 CPU 작업에 대한 내용까지도 학습할 수 있어 정말 유익했다. 기회가 된다면 프로세스를 직접 생성하고 CPU를 트래킹해보는 실습도 진행해보았으면 좋겠다는 생각이 들었다. - 손효림 오늘은 CPU 스케줄링에 대한 내용을 학습했다. CPU 스케줄링의 개념, 기준, 알고리즘과 스레드, 다중 프로세스의 스케줄링과 실시간 CPU 스케줄링 그리고 운영체제의 예와 알고리즘 평가에 대한 내용이다. 수업시간에 진도를 많이 나가지 않아 피피티를 제작하기 조금 힘들었다. 하지만 예습도 상당히 유익했다. 스터디가 학습에 상당한 도움이 되고 있기 때문에 중간고사에서 좋은 점수를 받을 수 있을 것 같은 기분이 든다. - 이경화 이번 주차에는 CPU 스케줄링을 주제로 스터디를 진행하였다. CPU는 프로세스를 처리하는 중앙 처리 장치인데, 운영체제는 이 자원을 여러 프로세스들이 효율적으로 활용할 수 있도록 스케줄링하는 역할을 한다. 이러한 CPU 스케줄링의 개념을 이해하는 데 있어서 CPU 버스트와 I/O 버스트라는 개념을 짚고 넘어가는 것이 도움이 많이 되었다. 또한 선점 스케줄링과 비선점 스케줄링을 비교하고, 자발적 컨텍스트 스위칭과 비자발적 컨텍스트 스위칭을 비교하는 것도 CPU 스케줄링에 대해 구체적으로 이해하는 데에 도움이 되었다. 특히 교안으로 학습 내용을 정리하여 스터디원의 관점에서 이해한 내용을 한번 더 들을 수 있어서 학습한 내용을 입체적으로 이해할 수 있어 좋았다. 아쉬웠던 점은 특별히 없었다. 
다음 학습계획	**일정 : 10월 18일 12:00~14:45 (50주년기념관 516호)**