**운영체제**

**Spring 2021**

**Final Exam**

**(50점 만점)**

**1. 정해진 시간 내에 답안을 작성 & 제출 해야함.**

**2. 불가피한 상황 (인터넷 다운등) 인 경우 답안의 사진 제출은 가능**

**(사진의 시간 정보 필수)**

**3. 정해진 시간내 미체출시 1분당 10%, 20%, 30%, 40% 페널티 부여 (4분 후 제출은 0점)**

**.**

**1.답안지의 해석은 전적으로 채점자에게 달려있음.**

**2.타인과 같은 오답 제출시 판단은 전적으로 채점자에게 의뢰함.**

**(2.5 +2.5+2.5+2.5= 10점) 이번 학기에 강의를 얼마나 성실하게 시청하였는지에 대한 평가**

이번 학기의 강의 제목은 아래와 같다

**1.1 프로세서 관리 부분**

**1) 사용자의 관점에서 예를 들어 클릭을 할 때, 어떤 인터럽트가 발생하여서 CPU가 어떤 식으로 움직이고 코딩을 할 때, 함수를 사용하면서 쓸모 없는 걸 include하면 커널에서 어떠한 일이 일어나는지 알아가면서 효율적으로 활용할 수 있다고 말씀하셨던 기억이 납니다.**

**2) 효율성과 편리성 중 효율성에 더 해당이 된다고 생각합니다. CPU의 스케줄링에 있어서 CPU의 활용을 극대화함으로써 효율적으로 일처리가 되기 때문입니다.**

**1.2 메모리 관리 부분**

**1) 메모리도 기본적으로 커널이라는 것이 차지하는 공간과 우리가 disk에 있는 모든 것을 끌어 올리는 것이 아닌 필요한 부분을 메모리 용량에 맞춰서 수시로 올렸다가 내렸다가를 하는 것을 배우게 되고 큰 용량의 파일들이 어떻게 움직이는지 알게 되거라고 하셨습니다.**

**2) 효율성에 더 가깝다고 생각합니다. 메모리의 양은 한정이 되어있기에 OS가 유동적으로 메모리의 공간을 관리하면서 프로그램이 잘 돌아가게 관리해주기 때문입니다.**

**1.3 파일 관리 부분**

**1) Logical과 Physical을 굉장히 강조하셨고 그럼으로써 우리가 쉽게 파일을 Logical적으로 사용하지만 내부에서는 Physical적으로 어떻게 동작하는지 알게 될 것이라고 하셨습니다.**

**2) 편리성에 더 가깝다고 생각합니다. 우리가 파일을 관리할 때 간편하게 드래그 하거나 파일의 용량도 상관없이 저장하지만 List의 형태로 파일이 항상 continuous하지 않게 저장되어도 저희는 의식하지 않고 Logical의 형태로써 사용하기 때문입니다.**

**1.4 입출력 관리 부분**

**1) 우리가 클릭, 모니터의 화면, 키보드 와 같이 간단하게 사용하지만 각각의 디바이스가 컴퓨터에 연결하기만 하면 작동하는 것은 디바이스 마다의 controller를 통해서 간편하게 이용된다고 하셨고 버스가 어떻게 이동하면서 데이터를 주고받는지 알게도리 것이라고 설명하셨습니다.**

**2) 편리성에 더 가깝다고 생각합니다. 입출력 하드웨어들이 운영체제에 맞춰서 Device들이 우리가 원하는대로 움직이도록 controller를 통해서 도와주기 때문에 더욱 편리하게 이용할 수 있다고 생각합니다.**

**각 부분에서 강의 중에 강조한 내용을 아래의 형식에 맞추어서 기술하라.**

**(아래의 형식에 맞추어서 답변하지 않으면 0점 처리)**

1) 왜 이 부분을 배워야 하는지

2) 운영체제의 2가지 주요 기능에 어느 부분에 해당되는지

등등 주요한 내용을 **상기해서 작성**하면 됨

**(8점) 간단한 단답형 문제**

1. (1+1) 일반적으로 scheduling 이란 무엇이며 언제 scheduling 이 발생하는가? (CPU 스케줄링, I/O 스케줄링을 구분하여 각각 제시해야 함)

Scheduling은 만약 일이 들어온 순서대로 일어난다면 예를 들어 1초면 끝나는 일이 1분이 걸리는 일 뒤에 들어왔다면 1분을 기다려야되는 수고가 있어야되고 짧은 일만 먼저한다면 1분이 걸리는 일은 계속 끝나지 않을 수도 있습니다. 그러기에 효율적인 scheduling을 통해서 각각의 일을 적절한 시기에 수행하는 것이 scheduling입니다.

1) CPU 스케줄링

CPU가 하나의 프로세스 작업이 끝난 후 다음 작업에 대해서 어떠한 프로세스를 처리할지에 대해서 선택할 때 CPU Scheduling이 일어납니다.

2) I/O 스케줄링

운영체제가 여러 프로세스한테서 I/O를 요청받을 때 그에 대한 우선순위를 선택할 때 스케줄링이 발생합니다.

1. (1) 멀티 프로그래밍이란 무엇이고, 이것이 가능하려면 운영체제는 무엇을 제공해야 하는가

멀티프로그래밍이란 CPU가 하나의 일만을 수행하는 것이 아닌 다른 프로세스도 처리할 수 있는 것입니다. 운영체제는 이를 위해서 CPU가 만일 하나의 입출력에 대해서 완료될 때까지 기다리고 있는 것이 아닌 완료될 때까지 다른 일을 할 수 있도록 하게 해주어야합니다.

1. (1+1) 컴퓨터 시스템은 하드웨어적 / 소프트웨어적으로 어떻게 구성되어 있는가? 각각 구성에 대해서 중요 요소를 거론하고 간단히 설명하라.
2. (1+1) 2000K 의 텍스트 파일이 있다. 블록 사이즈를 120K 이라고 가정 할 때 인덱스 와 링크 리스트 방법으로 파일을 구성할 경우 필요한 전체 사이즈를 각각 제시하라. (인덱스 크기는 120K, 포인터 크기는 10K)

링크 리스트 방법

포인터 크기 10K이므로 110K에 저장 2000/110 = 18.18~ 이므로 19개의 block이 필요합니다. 120K \* 19 = 2280K

인덱스 방법

인덱스로써 120K한개를 쓰고 나머지 블록은 저장을 하니 2000/120 = 16.666~ 으로 17개가 필요함으로 총 18개가 필요

120K \* 18 = 2160K

1. (1) disk access time 이란? 이것은 무엇에 의해서 지배를 받으며 시스템 효율에 어떠한 영향을 미치는지 상세히 논의하라.

disk에서 원하는 내용에 접근하는데 걸리는 시간으로써 디스크랑 만일 하드웨어적으로 멀리 떨어져있는 거리에 의해서 상대적으로 영향을 받습니다. 자주 함께 쓰이는 정보들이 disk의 track에 있어서 상대적으로 멀리 떨어져있다면 효율성에 있어서도 굉장히 ᄄᅠᆯ어질 것입니다.

**강의 이해 관련 문제**

1. (1.5점+1.5점) 운영체제 중 메모리 관리 모듈의 요소를 하드웨어 및 소프트웨어 관점에서 제시하고, 이 운영체제가 가상 메모리를 가능하게 하기 위해서 필요한 요소를 하드웨어 및 소프트웨어 관점에서 구체적으로 제시하라.

메모리 관리를 위해서 소프트웨어적으로는 Logical하게 가상주소를 만들어 CPU가 만들어낸 주소를 통해서 메모리를 관리합니다. 하지만 이러한 것을 하드웨어적인 Physical 주소는 RAM의 크기와 같은 공간 밖에 없기 때문에 MMU를 통해서 Logical 주소를 Physical주소로써 변환을 하여서 사용하게 됩니다.

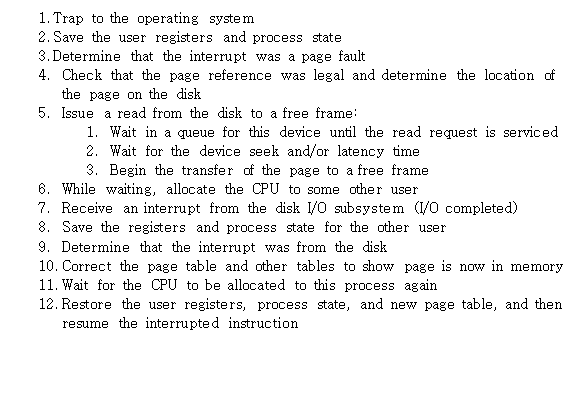
가상메모리가 가능한 이유는 Paging을 예로 들어서 설명하면 처음에 만약 데이터를 가져온다면 Page Fault가 발생하여서 OS에게 요청을 하고 소프트웨어적으로 정보를 가져오게 됩니다. 하지만 그 후 부터는 하드웨어적으로 Page table에 저장되어 있기 때문에 빠르게 정보를 가져올 수 있게 됨으로써 가상메모리를 사용할 수 있게 되는 것입니다.

1. (2점) 라운드 로빈 스케줄러는 일반적으로 정확히 한번 나타나는 각 프로세스들과 함께 모든 수행 중인 프로세스들의 리스트를 유지한다. 만약 프로세스가 리스트에서 두 번 나타난다면 어떻게 되는가? 이것을 허용한 이유는 어떤 장점을 얻고자 함인가? 단점은 무엇인가?

리스트에서 두 번 등장했다는 것은 Time Quantum을 넘기는 프로세스였다는 것이고 그로인해서 마치지 못했기에 다시 등장한 것이다. 이것을 허용하는 이유는 만약 이 프로세스가 너무나도 길었다면 그 뒤에오는 프로세스들은 한정없이 기다려야되는 상황에 놓이게 될 것이고 그렇게되면 일을 효율적으로 처리할 수 없게 됩니다. 타임 퀀텀을 둠으로써 정해놓은 시간보다 지체가 된다면 그 다음의 프로세스를 처리함으로써 효율적이게 할 수 있는 장점이 있습니다.

단점으로써는 만약 타임퀀텀이 프로세스들에 비해서 너무 짧다면 계속적으로 프로세스를 변경해줘야되고 그렇다면 너무 많은 오버헤드가 발생하여서 오히려 비효율적으로 변해버리게 됩니다.

1. (1+1+1 = 3점) 아래는 페이지 부재 시 발생하는 일련의 과정이다. 이 때 아래 문제에 대해 적절한 답변을 서술하시오



1. 2번의 과정에서 해당 프로세스의 상태가 변화하는 원인과 과정을 서술하시오(프로세스 상태도를 고려)

Page Fault가 발생했기에 OS를 통해서 그에대한 정보를 가져와야되기에 CPU에서 실행중인 상태인 프로세스를 다른 일을 해야되기에 중단 상태로 바꾸어 주어야합니다. 그래서 프로세스를 실행에서 중단으로 변경하고 징행사항을 PCB에 저장을 해주는 과정입니다.

1. 위의 과정에서 Context Switch가 발생하는 과정 번호와 발생하는 원인, 위의 과정에서 Context Switch가 발생하는 총 횟수를 서술하시오.

2번) 그 전에 진행중이었던 프로세스를 PCB에 저장하고 새로운 일을 시작하기 위해서

6번) Waiting을 하는 동안, CPU가 놀면 안되니 other user의 프로세스를 진행하기위해서

8번) Waiting이 종료되었기 때문에 other user의 프로세스를 중단하고 저장하고 다시 기다 기다리던 일을 실행하기 위해서

12번) 모든 Page Fault 과정이 종료되었음으로 그전에 수행하던일을 지속하기 위해서 PCB에서 저장되어 있던 정보를 가져와서 다시 진행하기 위해서

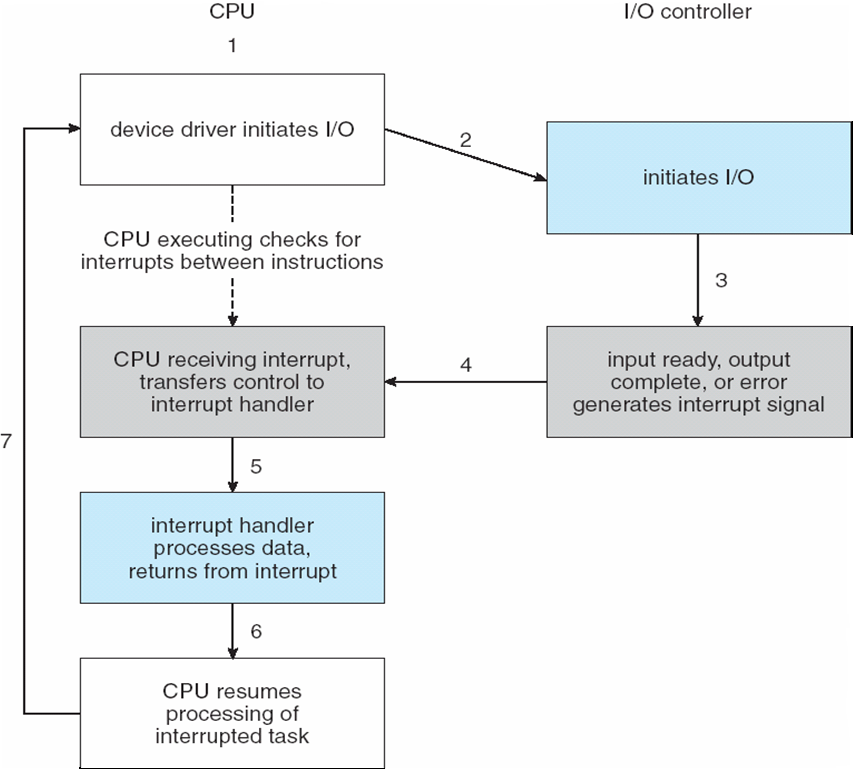
이렇게 총 4번의 Context Switch 가 발생합니다.

1. 필요한 페이지가 메모리에 적재된 후 10번 과정에서 페이지 테이블에 어떤 변화가 생기는지 서술하시오.

페이지 테이블에서 invalid 였던 것이 valid가 됨으로써 정보를 가져왔다는 것을

알 수 있게됩니다.

1. (1.5+1.5 = 3점) 프로그램에서 file open statement를 수행할 때 일어나는 interrupt-driven I/O의 과정을 나타낸 것이다.



여러분의 프로그램이 현재 파일을 읽고 있는 과정이고 그다음 step은 연산을 하는 statement 가 있다고 가정하고, 현재의 step 과 그 다음 연산을 하는 step 사이에 일어나는 일을 상세히 설명하라. 여러분의 프로세서 번호가 P1번이고, 다른 사용자의 번호가 P2-P9번, 그리고 OS가 P0 이다. (각 스텝 번호에서 어느 프로세서가 어떤 수행을 하는지 상세 제시해야 함, 그렇지 않으면 0점 처리).

1. Kernel I/O system에서 device driver에게 file을 open해 달라고 요청을 함

2. device driver가 I/O controller에게 file을 open의 명령을 전달하게 됩니다. 이렇게 한

후, CPU는 P1이 아닌 P2~P9의 프로세스를 진행하게 됩니다.

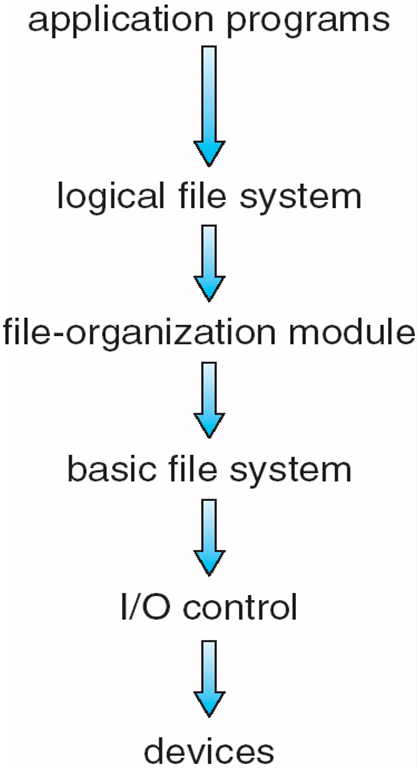
3. initiater I/O가 끝났다는 것을 의미합니다.

4. 인터럽트를 발생시키고 CPU는 P2 ~ P9의 프로세스를 진행하다가 인터럽트를 받게 됩 니다.

5. 인터럽트가 발생했으므로 interrupt handler라는 프로그램을 OS(즉, P0)에게 넘기게 됩 니다.

6. 인터럽트에 내용에 대해서 file open을 실행할 준비가 끝났다는 상황이고 Context swtiching을 발생시킵니다.

7. 인터럽트가 발생한 일 P1,에 대해서 실행을 하게 됨으로써 현재의 step을 마무리 짓게 됩니다.

1. (0.8+0.8+0.8+0.8+0.8 = 4점) 여러분의 파일이 실제 디스크에 저장되는 과정을 아래의 그림을 참조해서 상세히 설명하라. applicatio programs 부터 devices 까지 각 모듈 간의 맵핑 방법을 제시해야 하고 그렇지 않으면 0점 처리.
2. 

application programs 에서는 logical한 파일의 name을 넘기게 됩니다.

logical file system은 해당 파일의 logical한 주소에 대한 정보를 file-organization에게 넘겨주는 역할을 합니다.

file-organization module은 logical 주소를 physical하게 바꿔주는 역할을 합니다.

basic file system에서는 I/O control에게 block123가 같은 형식으로써 블락의 위치를 전해주게 됩니다.

I/O Controller 는 devices가 disk에서 몇 번 실린더, 몇 번 트랙, 몇 번 섹터등의 상세한 주소로 커맨드를 바꿔주는 역할을 합니다. 그러면 devices에서 해당 위치로 가서 저장을 진행하게 됩니다.

**(각 3점) 주관식 문제**

1. 10, 22, 20, 2, 40, 6, 38 실린더와 같은 디스크 요청들이 Disk driver로부터 들어왔다. 실린더 1을 움직이는데 걸리는 탐색시간은 6msec이다. 다음 알고리즘들을 사용하면 얼마나 많은 탐색시간이 필요한가? 모든 경우에 arm은 cylinder20에서 시작한다고 가정하자. (Cylinders range : 0~50) **(Answer w/o equation is penalty of -1.5)**
   1. First Come, First Service

20, 10, 22, 20, 2, 40, 6, 38 의 순서로 움직이므로

10+12+2+18+38+34+32 = 146\*6msec이 필요합니다.

* 1. Elevator(SCAN) algorithm (initially moved upward)

20,22,38,40 을 찍은다음 50까지 갔다가 다시 내려오면서 10, 6,2를 해결함으로

30+48 = 78\*6msec가 필요합니다.

1. 현재 컴퓨터 내에서 A 프로그램과 B 프로그램이 동시(time sharing)에 수행되고 있다고 가정하자. A 프로그램이 Data read instruction (I/O instruction) 을 수행하고 있고 B 프로그램이 수학적 계산 프로그램 (CPU instruction)을 수행하고 있는 경우 **OPTIMAL** 한 Operating System 이 하는 일을 I/O 관련 사항 (device, controller, driver)을 포함하여 매우 상세히 설명하라.

I/O instruction을 먼저 수행하고있는다고 하면 I/O read가 발생을 한다면 read가 일어나서 file을 read해달라고 요청을 해야되기에 OS의 커널 I/O system에서 I/O controller에게 요청을 하게됩니다. 그렇다면 CPU는 기다리는 것이 아닌 time sharing을 수행해야되기에 OS가 B프로그램을 수행하도록 하는 것입니다. Controller는 device에서 해당 정보에 대해 얻고 read할 준비가 완료가 된다면 인터럽트를 발생시키게 될 것입니다. 그렇다면 OS는 B프로그램을 PCB에 저장하고 다시 A프로그램에 대한 read를 수행함으로써 인터럽트를 수행하게 되고 이렇게 time sharing이 이루어지면서 여러 프로세스를 동시에 처리하는 것처럼 보이게 될 것 입니다.

1. 아래의 instruction 에서 address binding 은 compile time, load time, 그리고 execution time 등 3가지 경우에 발생할 수 있다. 세 가지가 발생할 때의 기계어 코드의 내용 변환 (if any) 을 쓰고, 이 세 가지중 가장 효율적이라고 생각되는 방법과 가장 비효율적이라고 생각되는 방법을 한 가지씩 선택하고 그 이유를 간략히 설명하라.

Load R7, abc : 메모리 abc 의 내용을 레지스터 7번에 저장

1010 0111 XXXXXXXX

(1010 : load, 0111 : register 7, XXXXXXXX : the rest 8bits are Memory address)

가장 비효율적인 것은 compile time 입니다. 왜냐하면 compile이 되자마자

해당 변수를 저장 해버리기 때문에 프로그램의 내부 주소인 Logical주소가 아닌 Physical주소로써 사용을 해서 같아져버리는 것입니다. 그러기에 가장 비효율적인 겁니다.

가장 효율적인 방식은 execution time 방식입니다. 실행이 될 때 주소를 바꾸어주는 방식으로써 예를 들어 abc의 내용이 8천 주소의 번지에 있다면 8천을 그대로 쓰는 것이 아닌 load가 되었을때의 시작점이 10000번지에서 시작을 한다면 실제로 abc 내용은 시작을 더해준 18000으로 찾아가는 방식입니다. 이러한 것은 MMU 하드웨어적으로 처리되기에 효율적이고 logical 주소를 사용하기에 더 편리합니다.

**(각 4점) 디자인 문제**

**다음의 서로 다른 성격의 프로그램이 있다(below there are two different programs)**

(1) 실시간으로 주어진 수학식을 계산하는 프로그램(A program that calculates the math equation in real time)

(2) 월별로 부서별 직원 봉급을 계산하는 프로그램(A program that calculates the employee salary for each departments monthly)

아래의 2단계는 운영체제의 발전과정을 나타낸다.(below two steps define the process of O/S development)

(i). No multiprogramming O/S

(ii). Multiprogramming support O/S

각각의 단계에서 운영체제가 제공해야 할 기능들(프로세서, 메모리, I/O)을 구체적으로 기술하고 (i)->(ii)로 upgrade 할 때 필수적인 것을 거론하라. (두 개의 서로 다른 프로그램이 수행되어야 한다는 점을 상기하기 바람)  
(describe concretely the function about processor, memory, I/O which is provided by O/S and essential factor to be upgraded in initial O/S. Remind that above two different programs should be executed.)

**(i) No Multiprogramming O/S**

A. Processor management

B. Memory management

C. File management

D. I/O management

**(ii) Multiprogramming support O/S (many program execute at the same time)**

A. 전 운영체제에서 필수적으로 upgrade 되어야 할 것(essential upgrading factor in initial O/S)

B. Processor management

C. Memory management

D. File Management

E. I/O management