● process를 간단히 설명해라

▶ 프로세스(process)란 실행중인 프로그램(program in execution)을 뜻합니다. 즉, 실행파일 형태로 존재하던 program이 RAM memory에 적재되어 CPU를 할당 받아 CPU에 의해 실행, 즉, 연산되는 것을 process라고 합니다.

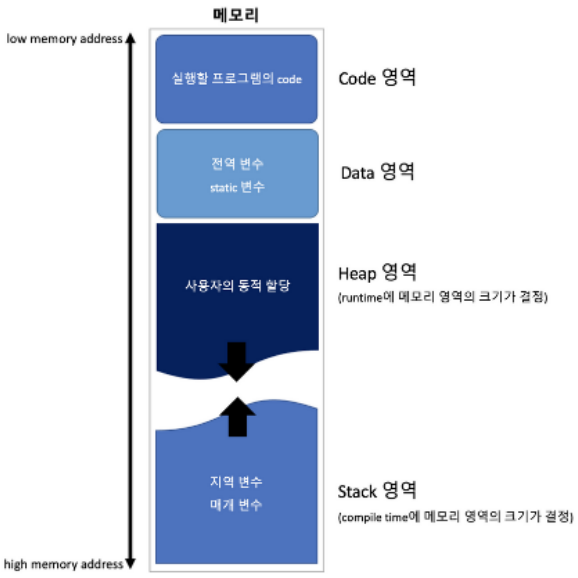
→ 프로그램은 단순히 명령어 리스트를 포함하는 파일입니다.

→ 운영체제를 관통하는 핵심적인 단어 하나를 뽑는다면 그건 바로 process입니다. 운영체제가 작동하는 다양한 원리들이 바로 process를 위해 존재하는 것입니다.

→ Ex) 코딩을 해서 컴파일 > 하드 디스크에 저장(HDD): 프로그램 > 프로그램을 실행 - 하드 디스크에 저장된 파일을 CPU가 직접 읽을 수는 없어서 RAM 메모리에 올리고(RAM 메모리에 적재하고) CPU를 할당 받는다 > CPU가 연산하며 프로그램이 process가 된다

● Memory에 적재

▶ memory(RAM)는 CPU가 직접 접근할 수 있는 컴퓨터 내부의 기억장치 입니다. program이 CPU에서 실행되려면 해당 내용이 memory에 적재된 상태여야만 합니다. 프로세스에 할당되는 memory 공간은 Code, Data, Heap, Stack 4개의 영역으로 이루어져 있으며, 각 process마다 독립적으로 할당을 받습니다.



Code 영역	실행한 프로그램의 코드가 저장되는 메모리 영역 (기계어가 저장된 부분)
Data 영역	프로그램의 전역 변수와 static 변수가 저장되는 메모리 영역
Heap 영역	프로그래머가 직접 공간을 할당(malloc)/해제(free) 하는, 즉, 동적으로 메모리를 할당하는 메모리 영역. runtime에 메모리 영역의 크기가 결정된다.
Stack 영역	함수 호출 시 생성되는 지역 변수와 매개 변수가 저장되는 임시 메모리 영역 (함수가 반환될 때 모두 반환된다). compile time에 메모리 영역의 크기가 결정된다.

● CPU의 연산과 PC register

▶ 프로그램의 코드를 토대로 CPU가 실제로 연산을 해야만 프로그램이 실행된다고 볼 수 있습니다. 어떤 코드를 읽어야 하는가(어떤 것을 연산해야 하는가)를 정하는 것은 CPU 내부에 있는 PC(Program counter) register에 저장되어 있습니다. PC register에는 다음에 실행될 코드(즉, 명령어, instruction)의 주소값이 저장되어 있습니다. 즉, memory에 적재되어있는 process code 영역의 명령어 중 다음번 연산에서 읽어야할 명령어의 주소값을 PC register가 순차적으로 가리키게 되고, 해당 명령어를 읽어와서 CPU가 연산을 하게 되면 process가 실행이 되는 것입니다.

● process의 memory영역(code, data, stack, heap)에 대해서 설명해라.
▶ 프로세스가 운영체제에서 할당받는 메모리 공간은 code, data, heap, stack 영역으로 구분됩니다. code 영역은 실행한 프로그램의 코드가 저장되는 메모리 영역입니다. data 영역은 프로그램의 전역 변수와 static 변수가 저장되는 메모리 영역입니다. heap 영역은 프로그래머가 직접 공간을 할당(malloc)/해제(free) 하는, 즉, 사용자가 동적으로 할당을 하는 메모리 영역입니다. stack 영역은 함수 호출 시 생성되는 지역 변수와 매개 변수가 저장되는 임시 메모리 영역입니다.

Resources: inflearn 개발남 노씨