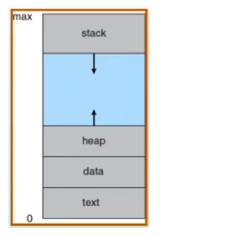
Chap24 프로세스 구조와 컴퓨터 구조

**1. 프로세스와 컨텍스트 스위칭**

* 프로세스 (process)는 일반적으로 어떻게 구성되어 있을까?
* Text(CODE):코드
* Data:변수/초기화된 데이터
* Stack: 임시 데이터(함수 호출, 로컬 변수등)
* Heap: 코드에서 동적으로 만들어지는 데이터

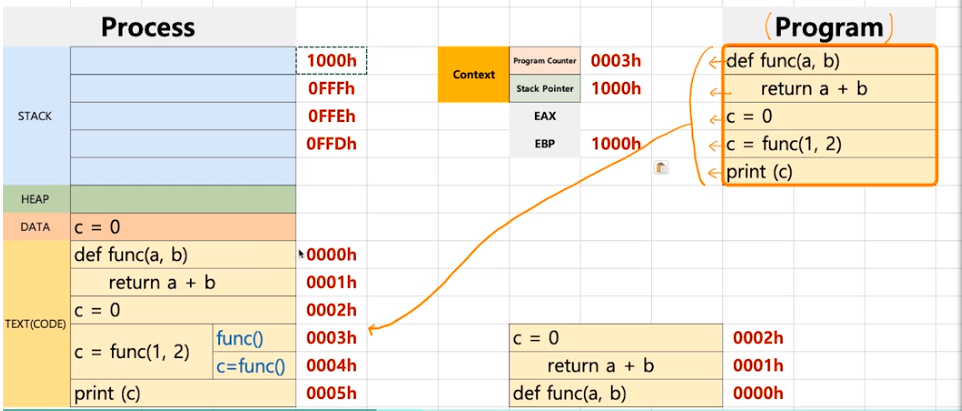
Process의 구조



주소값 process를 구성하는 4가지

**2. 프로세스와 컴퓨터 구조 복습**

- process는 sp(stack pointer)+pc(program counter)의 값을 지니는데 밑의 설명으로 이해합시다.



파이썬 프로그램 구문을 process로 돌려본다고 해보자. >>>중앙 밑단의 부분이 실제 실행 순서입니다.

Context를 보면 0002h라고 적혀있다.

Stack pointer, EBP는 최상단 값이 기억되는데, 우선 지정된 stack크기의 최대값인 1000h이 박힌다.

실행될때마다 program counter의 주소값이 그 주소값으로 변화한다. 0003h 0004h 이때

Stack에 저장되는 값들이 존재한다.

0000h > 0001h > 0002h 0003h일 때 0004h실행시점에서 함수작동으로 인해 stack메모리가 사용된다.

stack에는 처음 대용량 1000h , ret = 0004h, a=1,b=2 사용되었다.

Stackpointer, 변화한다. 위에 찼던 값들 제외한 최상단값이 적용된다.

fun성공시 ebp의 값도 변화한다. 그 시점에서 stack메모리의 최상단 값이 적용된다.



이 ebp가 의미가 있는데 프로그래밍 작업을 할 때

Fun1 fun2 fun4 진행시 fun2까지는 성공인데 fun4실패한 상황이라고 보자

이때 ebp는 fun2의 주소값을 지니고 있었기 때문에 ebp를 보면 어디부분이 잘못인지 발견할수 있다.

Return 부분으로 가면 위의 stack메모리가 전부 초기화되며

eax값으로 최종결과가 저장된다.

작업종료