**프로세스의 메모리 배치**

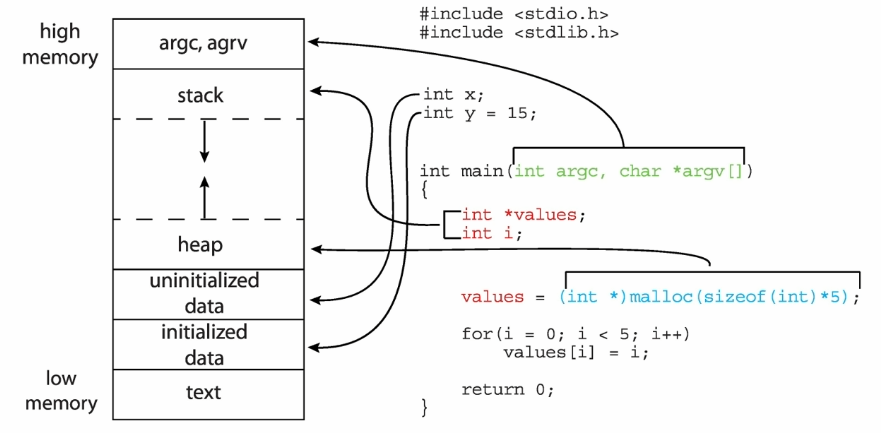
텍스트 섹션 – 실행 코드

데이터 섹션 – 전역 변수

힙 섹션 – 프로그램 실행 중에 동적으로 할당되는 메모리

스택 섹션 – 함수를 호출할 때 임시 데이터 저장장소

**C 프로그램의 메모리 배치**



**프로세스 상태**

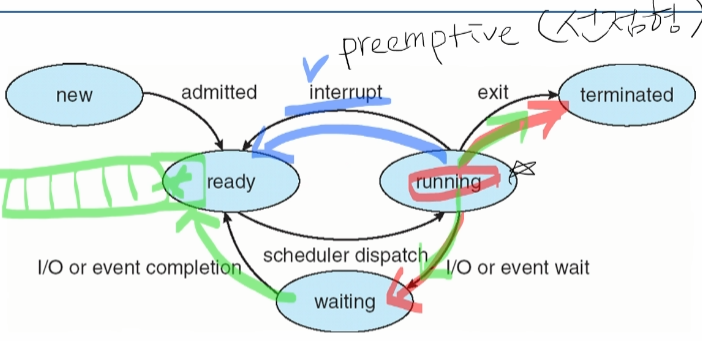
New: 프로세스가 생성 중이다

Ready: 프로세스가 CPU에 할당되기를 기다린다

Running: 명령어들이 실행되고 있다

Waiting: 프로세스가 어떤 이벤트(입출력 완료 또는 신호의 수신 같은)가 일어나기를 기다린다

Terminated: 프로세스의 실행이 종료되었다



**프로세스 제어 블록**: 특정 프로세스와 관련된 여러 정보들을 수록함

프로세스 상태

프로그램 카운터

CPU 레지스터들

CPU 스케줄링 정보

메모리 관리 정보

Accounting 정보

I/O status 정보

프로세스를 시작하거나 다시 시작시키는 데 필요한 모든 데이터를 위한 역할을 함

**다중 프로그래밍**: 현재 메모리에 있는 프로세스 수

**다중 프로그래밍의 목적**: CPU 이용을 최대화하기 위하여 항상 어떤 프로세스가 실행되도록 하는 데 있다

**I/O 바운드 프로세스**: 계산에 소비하는 것보다 I/O에 더 많은 시간을 소비하는 프로세스

**CPU 바운드 프로세스**: 계산에 더 많은 시간을 사용하여 I/O 요청을 자주 생성하지 않는다

**스케줄링 큐**

프로세스가 생성되면 ready 큐에 들어가서 준비 상태가 되어 CPU 코어에서 실행되기를 기다린다 프로세스에 CPU 코어에 할당되고 running 상태가 되면 여러 이벤트 중 하나가 발생할 수 있다

- 프로세스가 I/O 요청을 공표한 다음 I/O wait 큐에 놓일 수 있다

- 프로세스는 새 자식 프로세스를 만든 다음 자식의 종료를 기다리는 동안 wait 큐에 놓일 수 있다

- 인터럽트 또는 타임 슬라이스가 만료되어 프로세스가 코어에서 강제로 제거되어 ready 큐로 돌아갈 수 있다

프로세스가 종료될 때까지 이 주기를 계속한다

**문맥 교환**: CPU가 다른 프로세스로 전환할 때, 이전 프로세스의 상태를 저장하고 저장되어 있던 새 프로세스의 상태를 복원하는 작업

순수한 오버헤드이며, 문맥 교환 시간은 하드웨어의 지원에 크게 좌우된다