**auto 변수**

블록 안에서만 유효. 블록 실행이 끝나면 소멸.

기본적으로 사용되지만 생략되어 있음. 아무것도 표시하지 않은 변수는 auto와 같은 의미를 가짐.

**extern 변수**

외부의 변수를 쓸 수 있게 해주는 변수

선언 시 전역 범위에 선언됨. 프로그램 전체에 서 유효해짐.

Extern을 사용함으로써, 다른 파일에서도 참조하게 해 줌.

초기화를 생략하면 0으로 자동 초기화 됨.

**static 변수(정적변수)**

공간을 빠져나가도 형태를 유지하는 변수.(그냥 공간 대여를 하듯이, 계속 죽치고 있음.   
무궁화 꽃이 피었습니다! 너 움직이지마! 마냥 무궁화 꽃 술래가 보는 시간이 무한정 계속되는 느낌.)

초기화하지 않아도 0으로 초기화됨.

메모리에 항상 상주하도록 설정되기 때문에 가비지 컬렉터에 의해 시작점이 종료되는 것을 막아준다.

//초기화하지 않았다면, 0으로 초기화됨.

static변수는 선언될 때 한번만 초기화가 가능함  
//(선언할 때 초기화하지 않으면 0으로 초기화되기에 이후 대입하는 방법으로 접근이 불가능함.)   
(값을 대입하려고 하면 무시됨. 초기화를 함수 내부에서 하면, 처음 실행될 때 초기화되고 이후 무시됨.)

런타임 이전에 미리 할당됨.(프로그램 시작하자마자 정적 변수인 static 변수가 생성되고, 이후 런타임 중 다른 스택이나 힙 변수가 생성됨.)

**Java는 왜 static을 main 변수에 사용할까?** (static void main)

Main이 끝나도 유지시키기 때문에 static으로 존재함.

**지역변수 자투리**

지역변수는 초기화해줘야 함. 초기화 안해주면 오류남(요즘 컴파일러에선 자동으로 초기화해주는 경우도 있음).

**Resister 변수**

레지스터는 cpu에 있는 빠른 메인메모리

Static, global은 데이터 메모리에 저장되기에 레지스터 변수 선언이 불가능함.

레지스터 변수는 cpu에 존재하는 레지스터 공간에 변수를 생성.(빠름, 때문에 성능향상에 도움이 되어 자주 사용하는 변수에 선언해 성능향상을 노리기 위해 사용함.)

전역변수에는 레지스터 변수를 선언할 수 없음.

하지만 모든 변수를 레지스터에 넣을 수는 없기에, 제한이 있고, 제한을 넘긴다면 레지스터로 변수를 선언해도 메모리 공간에 저장할 수도 있다.

레지스터 변수에는 포인터 변수나 배열을 선언할 수 없다. 주소를 참조할 수 없기 때문이다. 따라서 어느정도 제약사항이 존재한다.

현대 컴파일러에서는 대부분 rsgister키워드를 무시하고 변수의 레지스터 할당을 자동으로 최적화하여 어떤 변수를 레지스터에 저장할지 판단하고 사용하는 경우가 허다함.

**Hip stack**

설명하다가 아무도 모른다니까 그냥 다른수업가서 배워라~ 하심.