스토리지 클래스

한국기술교육대학교
온라인평생교육원
OLEI

학습내용

- 기초 이해
- 정적 변수

학습목표

- 스토리지 클래스의 종류에 대해 설명할 수 있다.
- 정적 변수의 특징에 대해 설명할 수 있다.

기초 이해

1 기억부류(스토리지 클래스)

변수나 함수를 선언할 때 사용되는 키워드

1. 변수나 함수의 저장 위치와 사용 범위를 결정
2. 변수나 함수 선언 시 맨 앞에 지정
3. 변수의 디폴트 기억 부류는 auto이고, 함수의 디폴트 기억 부류는 extern

형식

- 기억부류 데이터형 변수명;
- 기억부류 리턴형 함수형(매개변수 리스트);

예제

auto	int num = 0;
register	int i;
static	int count;
extern	int global;
static	void f(void);

기억부류

기초 이해

2 auto 변수

지역 변수는 디폴트로 auto 기억 부류를 사용

➡ auto 지역 변수는 선언된 위치에서 자동으로 생성되고, 선언된 블록을 빠져나갈 때 자동으로 해제됨

전역 변수에는 auto 지정 불가

```
int main(void)
{
    auto int x = 10;
    int y = 20;
}
```

auto 변수를 선언함

auto를 생략해도 y는 auto 변수임

for 루프에서 선언 시 루프 내에서만 통용

```
for(int i=0 ; i < 5 ; i++)
    printf("%d", i );
```

기초 이해

3 register 변수

1. 변수를 메모리에 할당하는 대신 CPU의 레지스터에 할당
2. 변수를 레지스터에 할당하면 변수에 좀 더 빠르게 접근
3. 보통 루프 제어 변수를 레지스터 변수로 선언

```
register int i ;        ← 레지스터 변수를 선언함
for ( i = 0 ; i < 10000 ; i ++ )
    sum + = i ;
```

4. register 변수로 선언해도 변수가 레지스터에 할당되지 않을 수 있음
5. 레지스터 변수에 대해서는 주소 구하기 연산자를 사용할 수 없음

```
register int i;
printf( "%p", &i ) ;    ← 레지스터 변수의 주소는 구할 수 없으므로 컴파일 에러
```

기초 이해

4 global 변수

01 다른 곳에 선언된 변수에 대하여 별도의 메모리 할당 없이 해당 변수를 사용

02 전역 변수를 코드 중간에 선언하면, 전역 변수가 선언된 뒤쪽에 정의된 함수에서만 전역 변수를 사용 가능

전역 변수의 선언 위치

```
void TestGlobal(void)
{
        global = 10;
}

int global = 0;

int main(void)
{
        global = 20;
}
```

전역 변수 선언

아직 선언되기 전이므로 컴파일 에러가 발생함

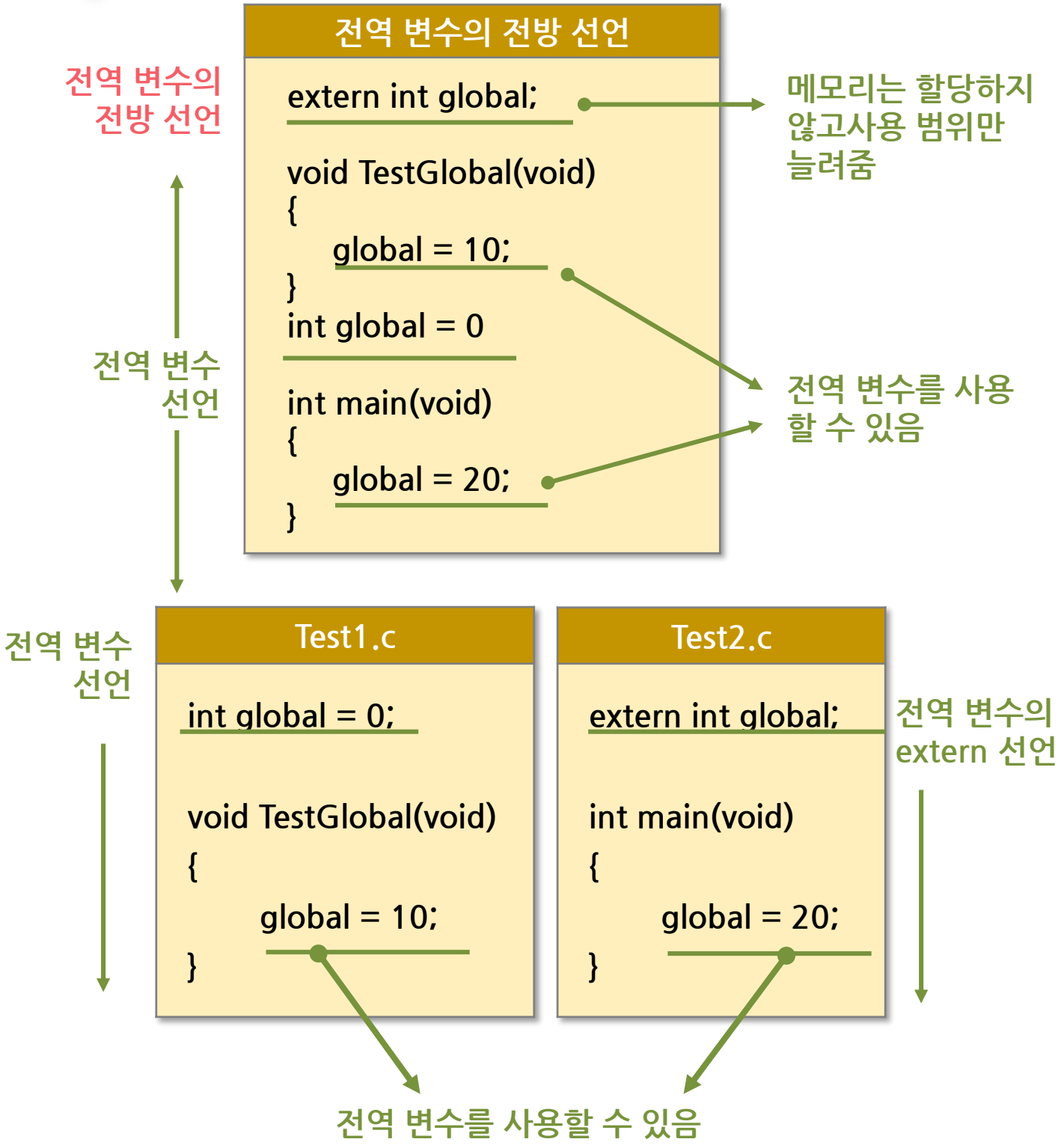
전역 변수를 선언한 이후에 사용할 수 있음

전방 선언(Forward Declaration)

전역 변수에 대한 extern 선언이 있으면
전역 변수가 선언된 위치와 관계없이 전역 변수 사용 가능

기초 이해

4 global 변수



정적 변수

1 정적 지역 변수

키워드 : static

1. 번역 과정의 마지막 단계인 적재 시간에 기억 장소를 할당받는 변수
2. 전체 프로그램의 시작부터 종료까지를 생존 기간으로 하며 동일 기억 장소를 유지
3. 초기화는 기억 공간 할당 시 한 번만 초기화됨
4. 정적 지역 변수는 전역 변수처럼 프로그램이 시작할 때 메모리에 할당되고 프로그램이 종료할 때 해제
5. 전역 변수와는 달리 정적 지역 변수는 선언된 함수 안에서만 사용
6. 정적 지역 변수는 함수가 리턴하더라도 해제되지 않고 남아있다가 그 다음 함수 호출 시 그대로 다시 이용

정적 변수

1 정적 지역 변수

```
#include <stdio.h>

void func() {
        static int x = 0;
        int y=0;
        x++;
        y++;
        printf("%d   %d₩n", x,  y);
}
int main() {
        func();
        func();
        func();

        return 0;
}
```

정적 변수

1 정적 지역 변수

```
char* Reverse(const char *str)
{
    char result[80];
    int i;
    int len = strlen(str);
    for( i = 0 ; i < len ; i ++ )
        result[i] = str[len - i - 1];
    result[i] = '\0';
    return result ;
}
```

주소(char*)를 리턴하는 함수

result 배열을 지역 변수로 선언함

포인터를 리턴하는 함수를 정의할 때는 함수 안에 선언된 지역 변수의 주소를 리턴하지 않도록 주의해야 함

result 배열명이므로 지역 변수인 배열의 주소를 리턴함

```
char* Reverse(const char *str)
{
    static char result[80];
    int i;
    int len = strlen(str);
    for( i = 0 ; i < len ; i ++ )
        result[i] = str[len - i - 1];
    result[i] = '\0';
    return result ;
}
```

result 배열을 정적 지역 변수로 선언함

result를 정적 지역 변수로 선언하면 함수가 리턴하더라도 해제되지 않으므로 올바른 실행 결과를 얻을 수 있음

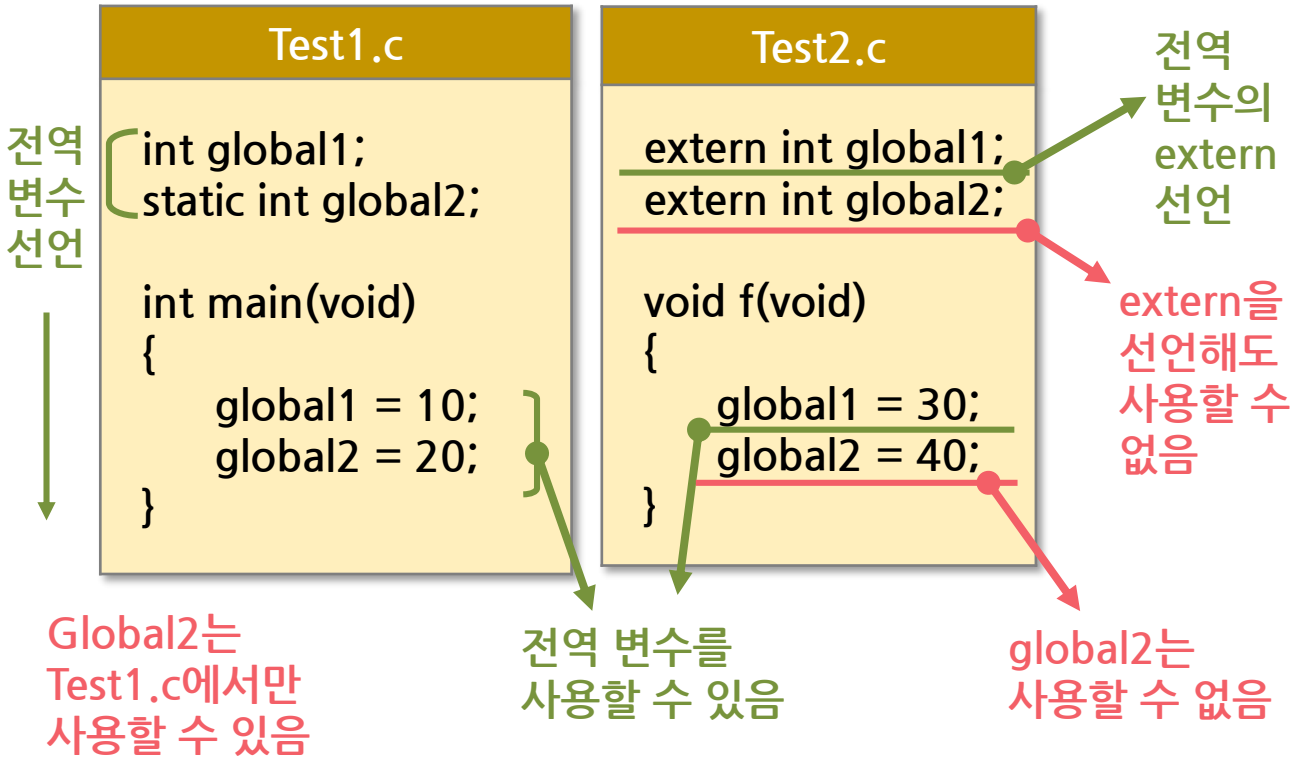
정적 지역 변수의 주소를 리턴함

정적 변수

2 정적 전역 변수

정적 전역 변수는 정적 전역 변수가 선언된 소스 파일에서만 사용

정적 전역 변수는 전역 변수를 다른 소스 파일에서 접근하지 못하도록 제한



정적 변수

3 스토리지 클래스 비교

구분	일반지역변수	정적지역변수	전역변수	정적전역변수
선언 위치	함수 안	함수 안	함수 밖	함수 밖
생성 시점	변수 선언 시	프로그램시작시	프로그램시작시	프로그램시작시
해제 시점	함수 리턴 시	프로그램종료시	프로그램종료시	프로그램종료시
사용 범위	함수 안	함수 안	프로그램 전체	선언된 소스 파일
초기화하지 않았을 때	쓰레기 값	0으로 초기화	0으로 초기화	0으로 초기화

학습정리

1. 기초 이해

- 지역 변수는 자동 변수임
- 자동 변수는 선언된 블록을 빠져나가면 소멸됨
- 레지스터 변수는 CPU의 레지스터에 할당되는 메모리도 빠른 액세스가 가능함
- 전역 변수는 프로그램 전체에서 통용되고 다른 파일에서도 통용 가능함

2. 정적 변수

- 정적 지역 변수는 할당 시 한 번만 초기화되고 선언된 블럭을 빠져나가도 소멸되지 않음
- 정적 전역 변수는 프로그램 전체에서 사용 가능함
- 정적 변수는 할당 시 0으로 초기화됨