**NA-HW #02**  **-** 2015004693\_양상헌

**Read Chapter 1, Numerical recipes in C, and summarize­**

* **How to use pointers for memory allocation**
* **How to use pointers to function**

**Solve the Problems: 3.6, 3.7, 4.2, 4.5, 4.12**

-**Error, Accuracy, and Stability.**

컴퓨터는 한정된 비트로 무한대에 가까운 실수를 표현하기 때문에 실제 나타내고자 하는 숫자와 컴퓨터가 나타낼 수 있는 숫자가 항상 같을 수 없다. 0에서 1사이의 숫자를 나타낼 때 표현 가능한 수의 갯수와 1에서 무한대 사이의 숫자를 나타낼 때 표현 가능한 수의 갯수가 같기 떄문에 0에서 1사이의 숫자들끼리의 간격은 매우 촘촘하고, 1에서 무한대 사이의 숫자 끼리의 간격은 상대적으로 매우 크다.

한정된 비트로 모든 숫자를 표현하기에 실제로는 continuous한 숫자들을 discrete 하게 표현 할 수 밖에 없고 이로 인해 특정 숫자를 표현할 때 그 숫자가 정확이 discrete한 표현범위에 없는 숫자일 경우 반올림 혹은 올림, 버림 을 통해 discrete 한 표현할 수 있는 숫자로 나타내기 때문에 여기서 발생하는 오차가 있고 이를 RoundOff Error라고 한다. 일반적으로 RoundOff Error는 연산의 횟수가 많아질 수록 오차가 커지는 성질을 가진다.

또한 RoundOff error 와 대비되는 개념인 Truncation error가 있는데, 이는 미분계산이나 급수 계산같은 경우에서 계산의 차수가 무한대의 가까워질 수록 실제 답에 가깝게 되는데 컴퓨터는 무한대 계산을 할 수 없으므로 계산 차수를 어느정도 숫자에서 멈춤으로서 생기는 오차를 의미한다. 일반적으로 Truncation error는 계산 횟수 (차수) 가 많아질 수록 실제 값에 가까워져 오차가 줄어든다.

RoundOff Error 와 Truncation Error는 서로 Trade-Off 관계이기 때문에 RoundOff error 와 Truncation Error의 합이 최소가 되도록 하는것이 오차를 최소화 할 수 있는 방법이고 가장 바람직한 방법이다.

* **How to use pointers for memory allocation**

[Vectors and one-dimensional arrays]

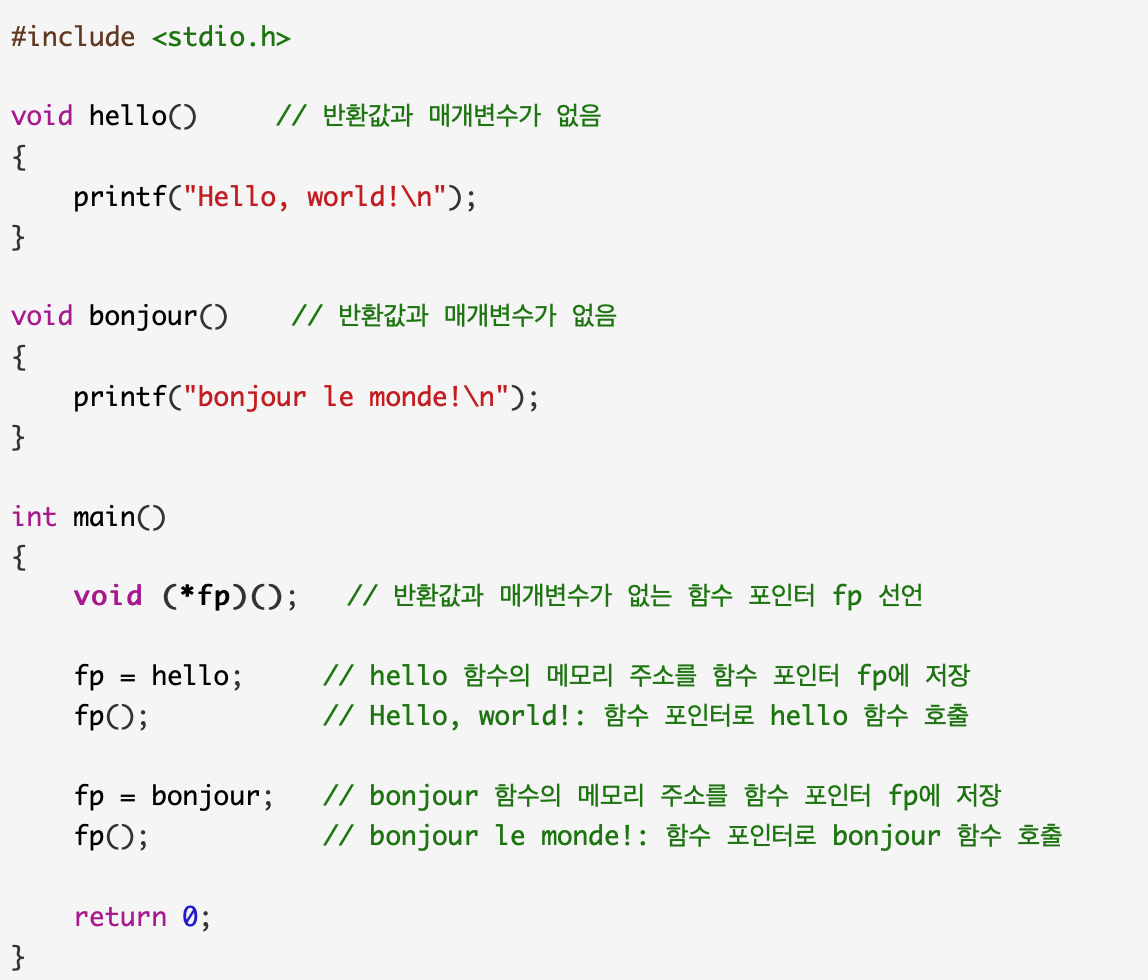
C 에서는 Pointer 와 array의 관계에 주목해야한다. 1차원 배열의 경우 a[j] 은 a 배열의 j index의 값을 나타내는데 이는 포인터로 표현시 \*((a)+(j)) 로 표현할 수 있다. Pointer a의 주소값에 j를 offset으로 더하는 것이다. Array a[0]의 주소와 pointer a 의 주소가 같기 때문이다.

[Matrices and two-dimensional arrays]

2차원 배열 (혹은 행렬) 역시 1차원 배열과 비슷하게 두개의 포인터로 나타낼 수 있다. a[j][k] 는 사실상 표현은 보기 쉽게 2차원으로 하지만 실제 메모리에 저장 될 때는 긴 1차원 배열로 저장이 된다. a[j][k]는 \*((a)+(k\*j)+(k)) 인 것이다. 이중포인터를 사용할 때 a[][] 를 \*\*a 로 표현하는데, \*\*a 는 a[0][0]의 주소값을 의미하고, \*a[0]는 a[0][0] 의 주소, \*a[1]은 a[1][0] 의 주소 와 대응된다.

* **How to use pointers to function**

C언어 에서는 함수를 다른 함수의 인자로 바로 사용하는 것이 불가능 하기 때문에 함수 포인터를 사용해두어야 한다. 일단 아래의 사진과 같이 함수를 void 형으로 선언을 해준 후, 메인 함수 안에서



사진출처: <https://dojang.io/mod/page/view.php?id=592>

함수를 가리키는 포인터 변수를 선언해준 다음 포인터 변수에 함수를 대응시키고 함수를 직접 호출하지 않고 포인터 변수를 호출해주면 함수가 실행되는 결과를 볼 수 있다. 이 방법을 이용하여 하나 또는 여러 개의 함수를 인자로 필요로 하는 또다른 함수를 실행 시킬 때 함수 내부를 편집하지 않고도 함수를 인자로 사용할 수 있다.