2주차 : Insertion & Merge sort

알 고 리 즘

2015. 9. 11.

충남대학교 컴퓨터공학과 임베디드 시스템 연구실
조교 권진세

Overview

- **알고리즘의 수행 시간**
  1) **시간 복잡도**
  2) O-, Ω-, and Θ-notation
- **두 가지 정렬 방법 소개 및 Time Complexity 계산**
  1) **Insertion sort**
  2) **Merge sort**
- **실습 / 과제**

  파일 입출력을 사용한 Insertion & Merge sort 구현

Time Complexity

▶ Time Complexity (시간 복잡도)

알고리즘을 구성하는 모든 명령어들에 대해서
각각의 **[수행에 필요한 Cost x 수행 횟수] 의 총합**

Notation

▶ **O-notation (최악의 경우) : f(n) = O(g(n))**

모든 $n \geq n_0$에 대해 **$0 \leq f(n) \leq cg(n)$**인 양의 상수 n, c이 존재할 때

e.g. $2n^2 = O(n^3)$ ($c=1$, $n_0=2$)

▶ **Ω-notation (최상의 경우) : f(n) = Ω(g(n))**

모든 $n \geq n_0$에 대해 **$0 \leq cg(n) \leq f(n)$**인 양의 상수 n, c이 존재할 때

e.g. $\sqrt{n} = \Omega(\lg n)$ ($c=1$, $n_0=16$)

▶ **Θ-notation (평균인 경우) : f(n) = Θ(g(n))**

$\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$ 일 때

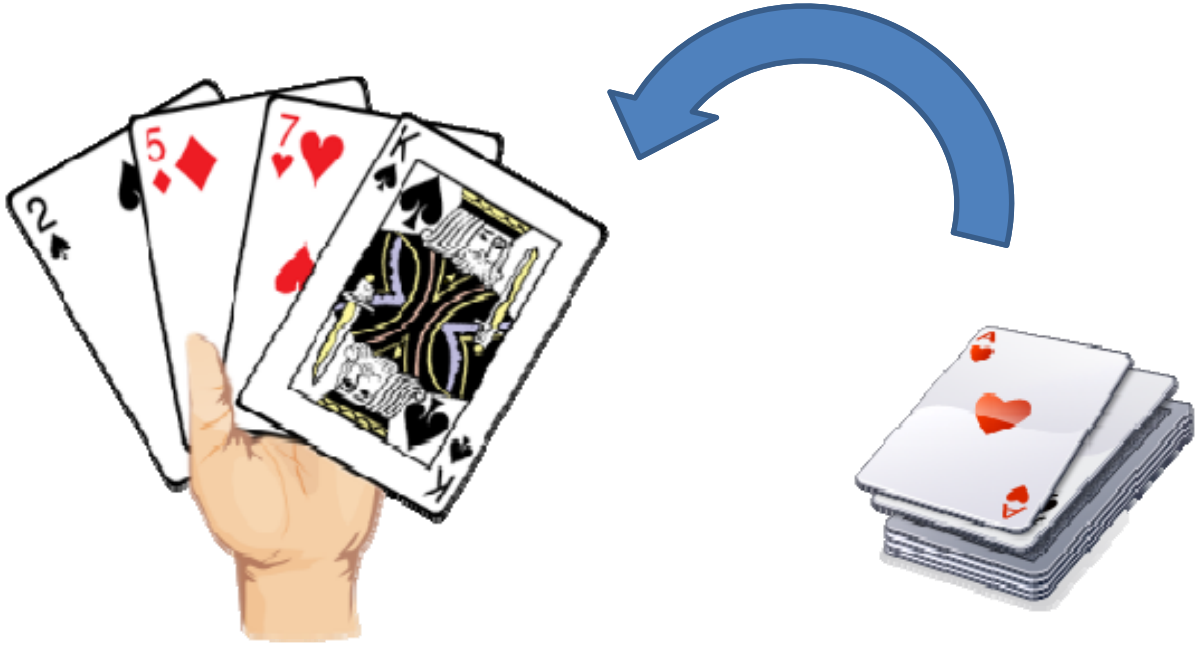
e.g. $\frac{1}{2}n^2 - 2n = \Theta(n^2)$

※ 양의 상수 n과 c, 계산 방법에 따라 여러 가지 g(n)을 구할 수 있다.
단, 일반적으로 가장 근접한 값을 찾도록 한다.

Insertion Sort

▶ Insertion Sort (삽입 정렬)

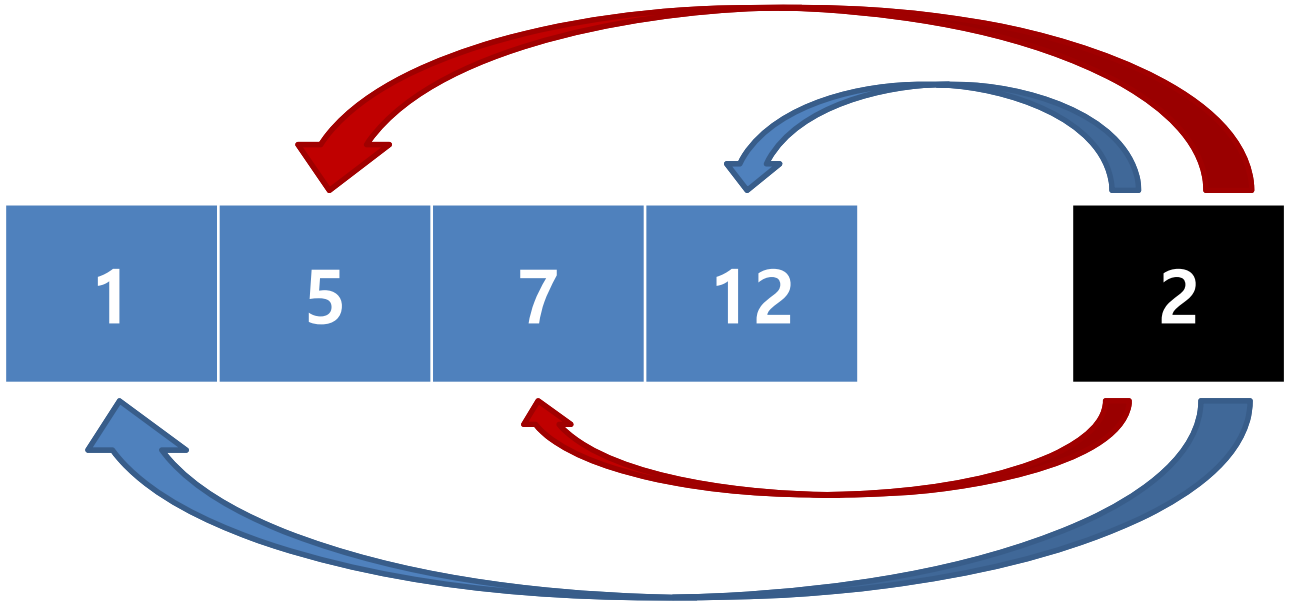
정렬되지 않은 배열로부터 **데이터를 하나씩 꺼내**어
정렬되어 있는 배열의 알맞은 위치에 삽입하는 정렬 방법



Insertion Sort

- **프로그램으로 구현 시 달라지는 점**

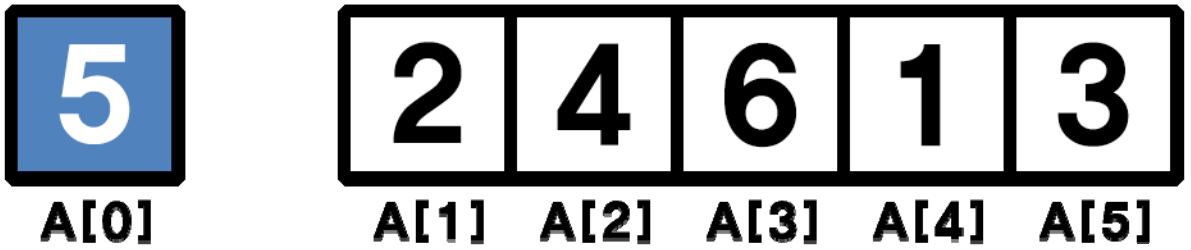
  알맞은 위치에 데이터를 삽입하기 위해
  배열에 저장된 값들을 하나씩 **순서대로 비교**해 보아야 함



Insertion Sort

▶ 정렬 방법 (1/7)

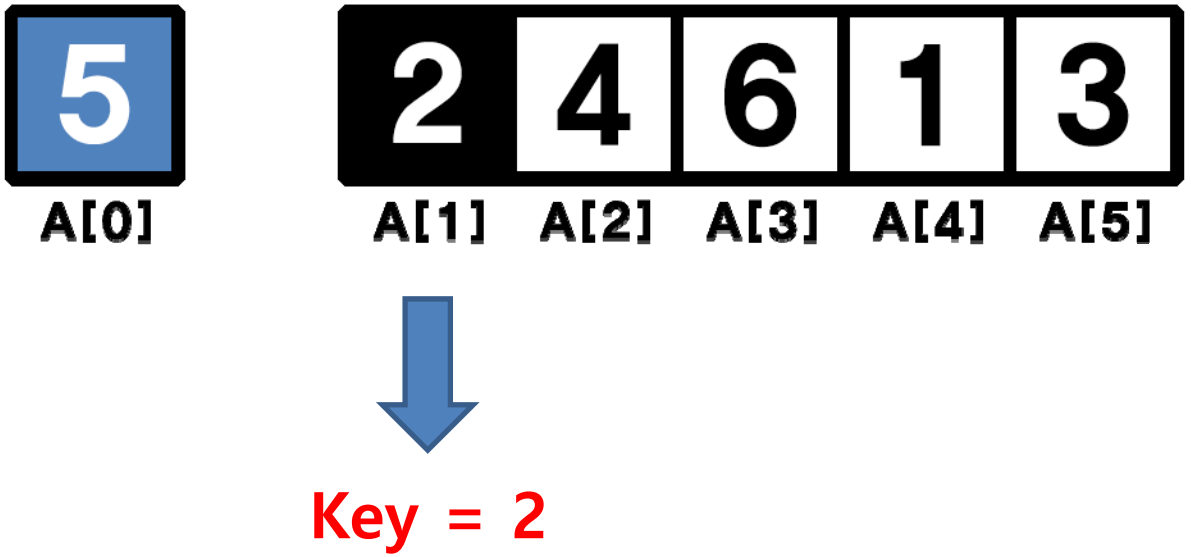
배열의 첫 번째 데이터를 정렬된 배열,
나머지 데이터를 정렬되지 않은 배열로 나누어 생각한다



Insertion Sort

▶ 정렬 방법 (2/7)

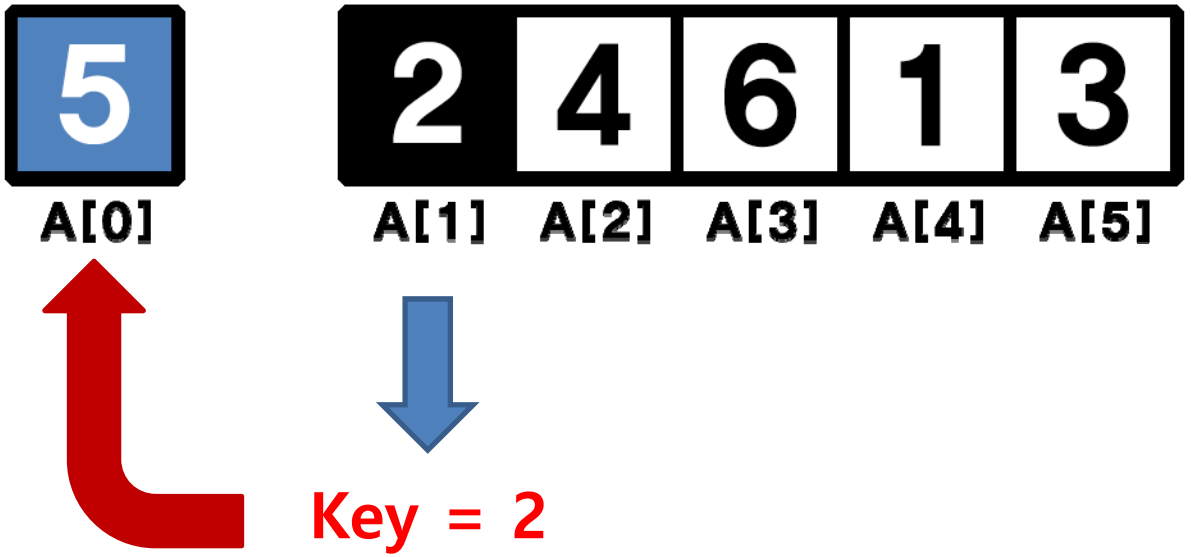
우측 배열의 첫 번째 데이터 값을 변수 **Key**에 복사한다



Insertion Sort

▶ 정렬 방법 (3/7)

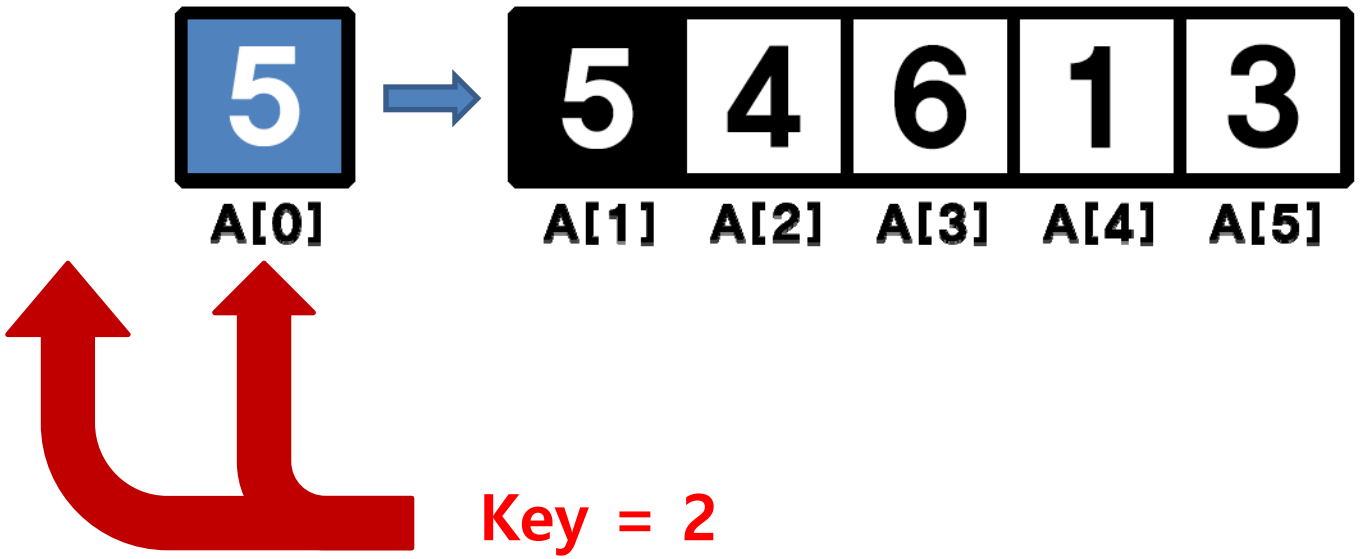
복사한 Key 값을 좌측 배열에 저장된 수 중
가장 마지막(오른쪽) 배열에 저장된 값과 비교한다



Insertion Sort

▶ 정렬 방법 (4/7)

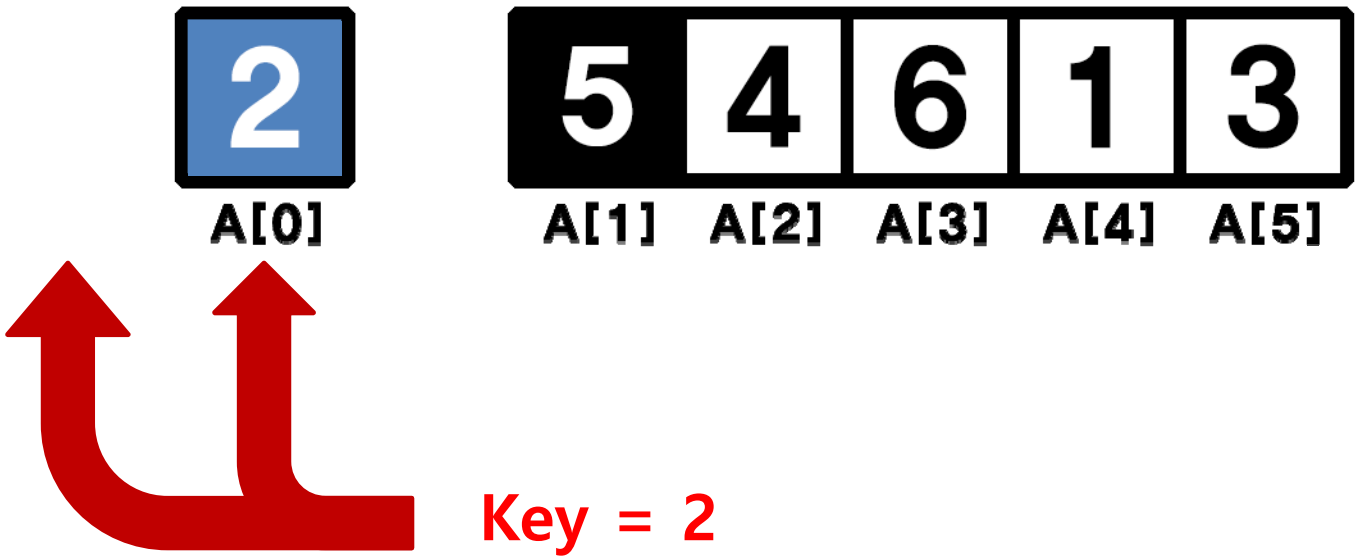
만약 비교한 값이 Key보다 크면
해당 값을 오른쪽 인덱스에 복사하고 그 왼쪽 값을 비교한다



Insertion Sort

▶ 정렬 방법 (5/7)

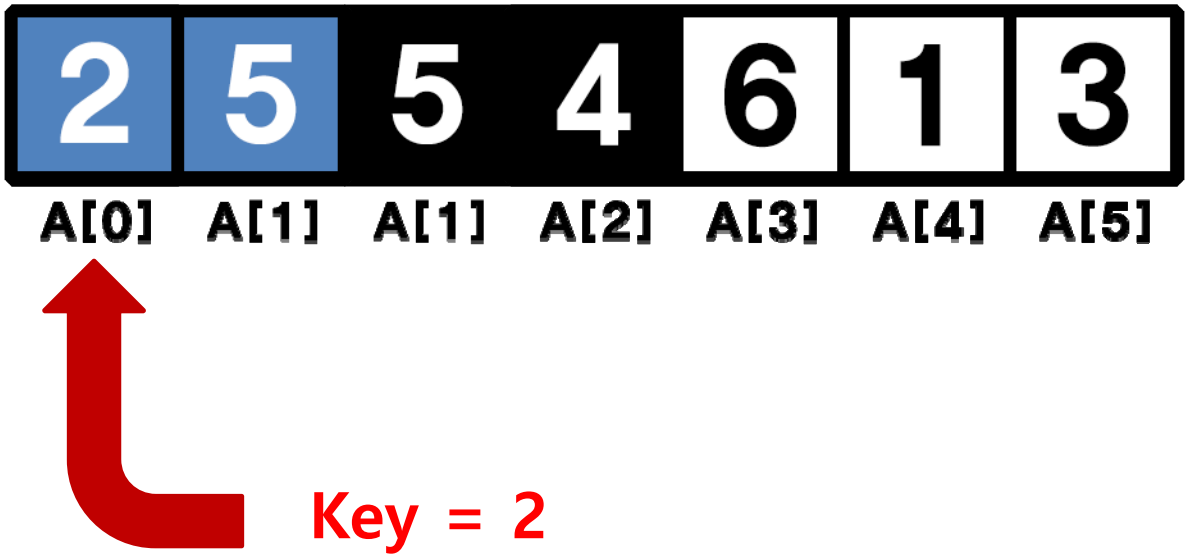
더 비교할 값이 없거나, 비교한 값이 Key보다 작거나 같다면
비교했던 위치의 바로 오른쪽 인덱스에 Key 값을 복사한다



Insertion Sort

▶ 정렬 방법 (6/7)

1개 데이터에 대한 삽입 정렬이 완료되었다
남은 A[2] ~ A[5]의 데이터에도 같은 작업을 반복한다



Insertion Sort

▶ **정렬 방법 (7/7)**

완료

Insertion Sort

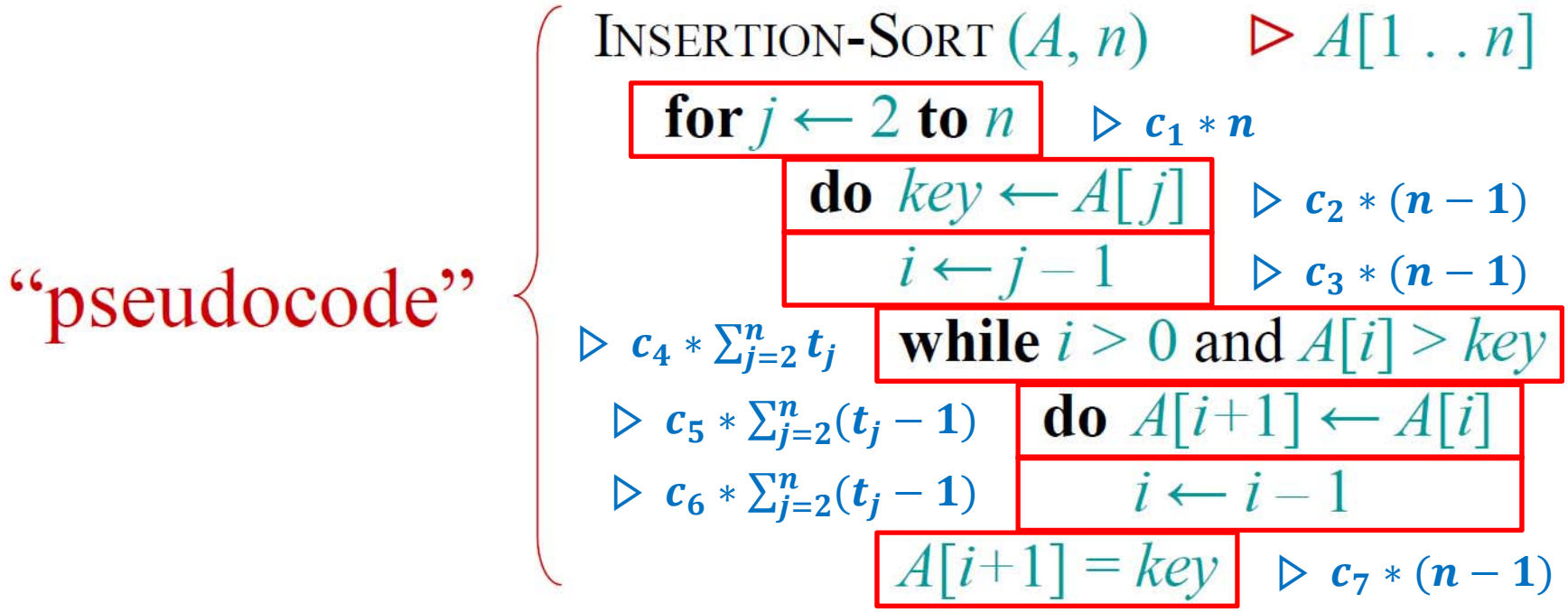
▶ **동영상 자료 시청**

삽입 정렬

6 5 3 1 8 7 2 4

Insertion Sort

▶ **pseudo-code (의사 코드)**



※ 반복문의 루프가 종료될 때, 한 번 더 검사를 수행하는 점에 유의한다.
또한 $j = 2\ to\ n$ 일 때의 t_j 값은, $best\ case = 1$, $worst\ case = j$이다.

Review : Sequence

▶ **등차 수열 공식**

$\{a_n\} : a_1,\ a_2,\ \dots,\ a_n$ (공차 : d)

$a_n = a_1 + (n-1)d \quad (n = 1, 2, 3, \dots)$

$S_n = \frac{n(a_1+a_n)}{2} = \frac{n\{2a_1+(n-1)d\}}{2}$

▶ **등비 수열 공식**

$\{a_n\} : a_1,\ a_2,\ \dots,\ a_n$ (공비 : r)

$a_n = a_1 r^{n-1} \quad (n = 1, 2, 3, \dots)$

$S_n = \frac{a_1(1-r^n)}{1-r} = \frac{a_1(r^n-1)}{r-1} \quad (r \neq 1) \qquad S_n = na \quad (r = 1)$

Merge Sort

▶ **Merge Sort (합병 정렬)**

[분할] – [정복] – [결합] 과정을 **재귀적으로 반복**하는 정렬 방법

(1) 분할 (Divide) → mergeSort()

배열의 크기가 1이 될 때까지 계속하여 **배열을 둘로 나눈다**

(2) 정복 (Conquer) → merge()

나눠진 데이터를 **2개 배열씩 비교하여 재귀적으로 정렬**한다

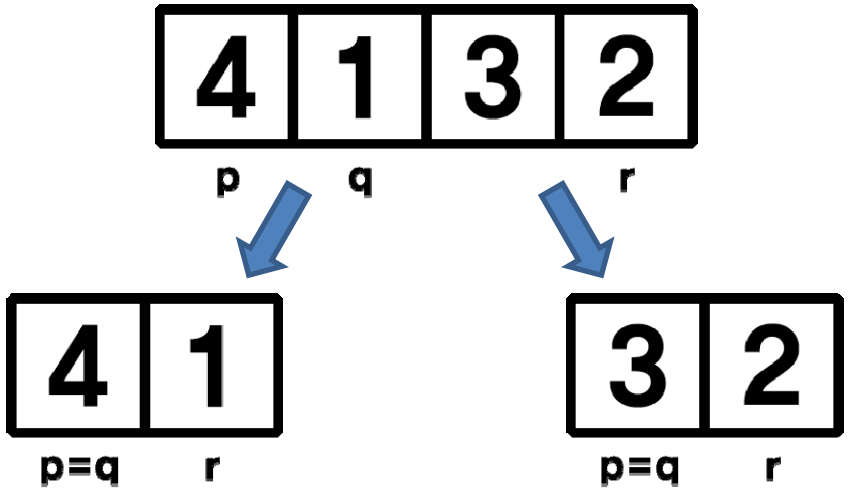
(3) 결합 (Combine) → merge()

정렬된 두 개의 배열을 병합해 하나의 정렬된 배열로 만든다

Merge Sort

▶ 정렬 방법 (1/4) – 분할 (1/2)

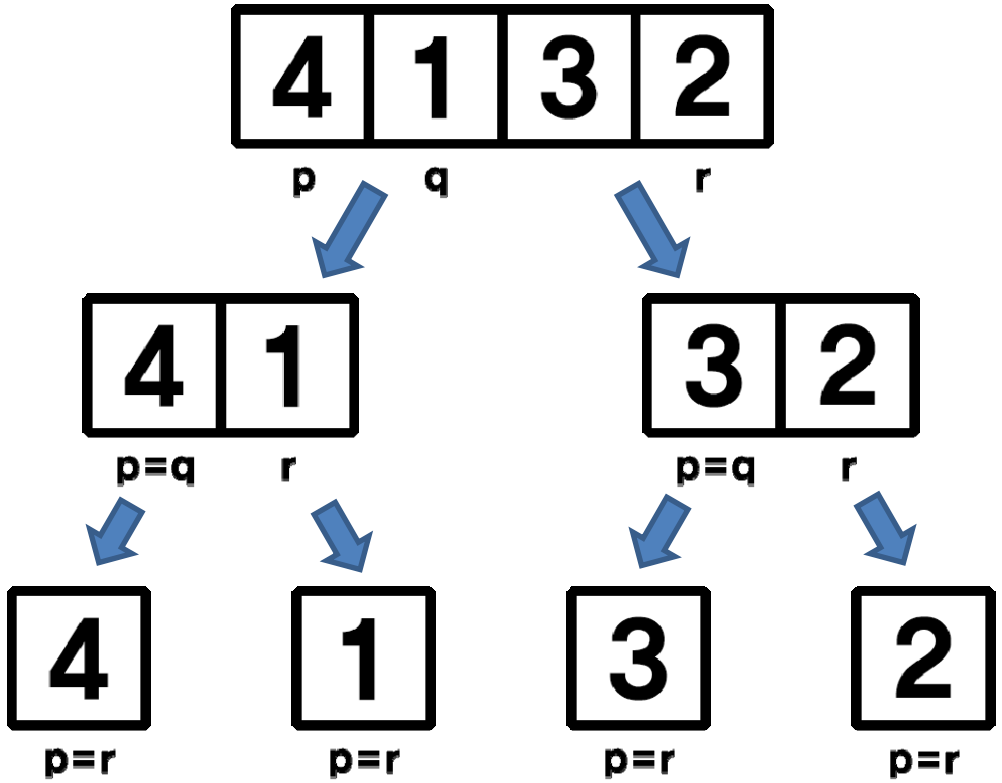
배열의 처음과 마지막 인덱스 넘버를 p, r이라 하고
가운데 인덱스 넘버를 q라 하여, 이를 기준으로 배열을 나눈다



Merge Sort

▶ **정렬 방법 (2/4) – 분할 (2/2)**

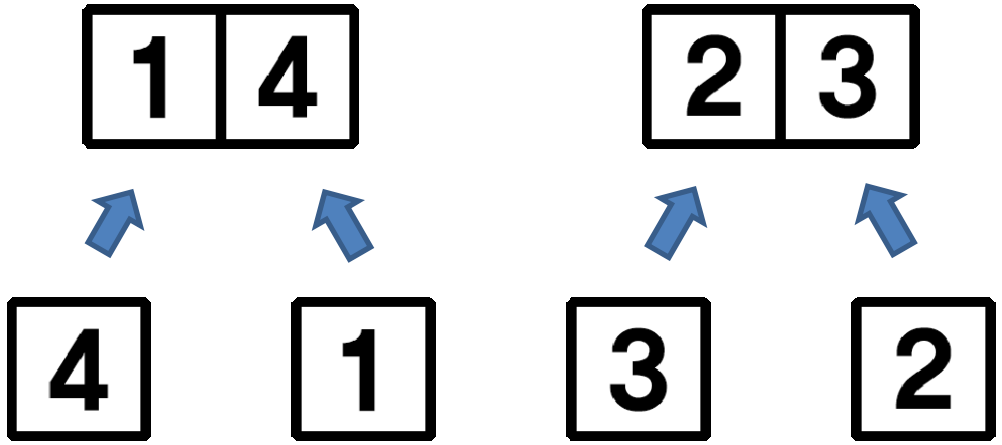
배열의 크기가 1이 될 때까지 계속하여 배열을 둘로 나눈다



Merge Sort

▶ **정렬 방법 (3/4) – 정렬 & 결합 (1/2)**

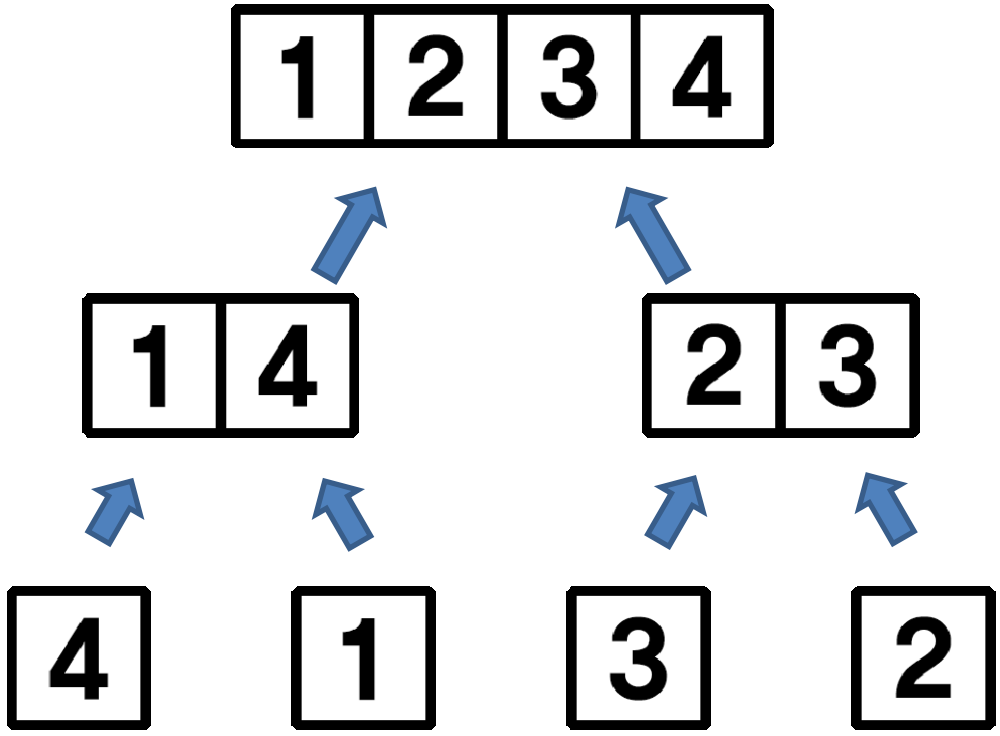
두 배열의 가장 앞 데이터를 비교하여
더 작은 값부터 차례대로 뽑아내어 정렬 및 결합한다



Merge Sort

▶ **정렬 방법 (4/4) – 정렬 & 결합 (2/2)**

원래의 크기가 될 때까지 계속하여 정렬 및 결합하면 완료



Merge Sort

▶ **동영상 자료 시청**

합병 정렬

6 5 3 1 8 7 2 4

Merge Sort

MERGE-SORT $A[1 \,..\, n]$

1. If $n = 1$, done.
2. Recursively sort $A[\,1 \,..\, \lceil n/2 \rceil\,]$ and $A[\,\lceil n/2 \rceil + 1 \,..\, n\,]$.
3. "***Merge***" the 2 sorted lists.

Key subroutine: **MERGE**

APPENDIX 1. File I/O

1. 파일 입출력 방법

```
FILE* fp;          //fp : input file pointer
FILE* fop;         //fop : input file pointer

//파일 이름은 "00_201500000_insertion.txt"
//입출력 파일은 *.c 소스파일과 같은 폴더에 있어야 한다.
fp = fopen(FILENAME,"rt");              //입력 파일 열기
fop = fopen(FILENAME2,"wt");            //출력 파일 열기

if (fp == NULL) {
        printf("**** Input File open error ****₩n");
        exit(1);
} //파일 없을 경우 예외처리로 프로그램 종료
```

APPENDIX 1. File I/O (계속)

```
while(!feof(fp)){
} //파일의 끝날때 까지 반복

fscanf(fp,"%d, ", &변수);        // ex) 123, 456,
                                 // int 값만 추출함 ‘, ’는 제외됨

fprintf(fop, "%d", 출력할 값);  // 파일 출력 시 사용

fclose(fp);
```

APPENDIX 2.

Q & A.

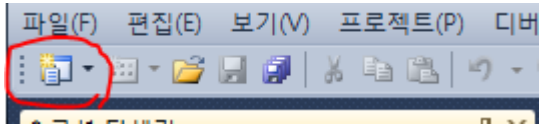
포인터로 받은 배열의 크기를 구하는 방법?

- 포인터가 가르키는 배열의 크기를 구하는 방법은 없다.
- 모든 Standard 함수가 배열의 크기도 함께 arg(인자로)로 받는다.
- 따라서 포인터로 배열을 다룰 때, 배열의 크기를 함께 인자로 넘기는 코딩을 하도록 하자.

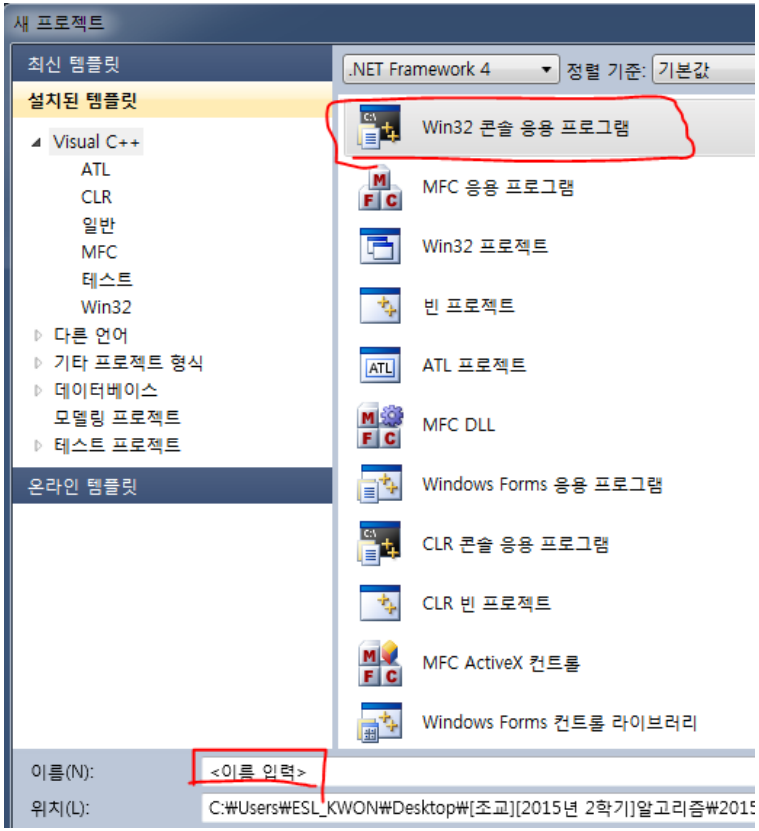
Trouble Shooting

Visual Studio 2010 프로젝트 생성

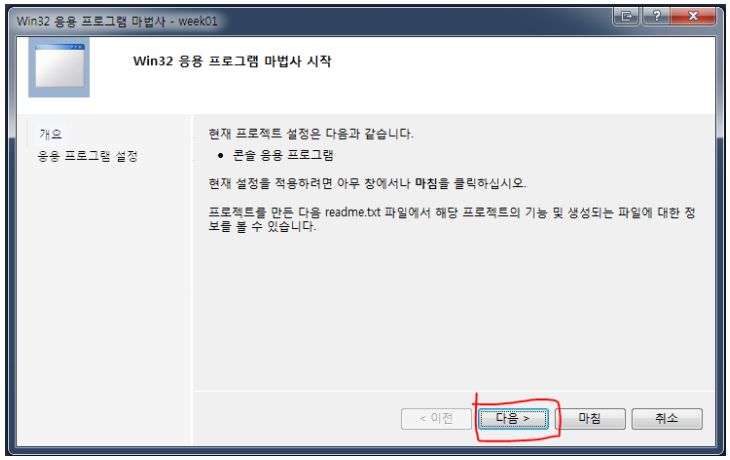
1.



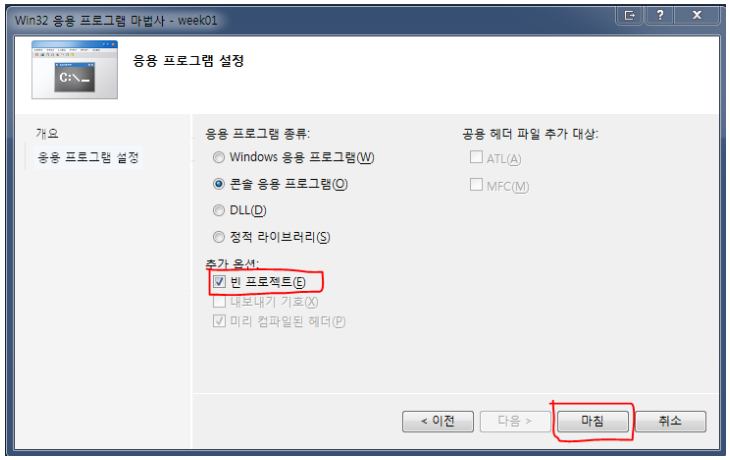
2.



3.



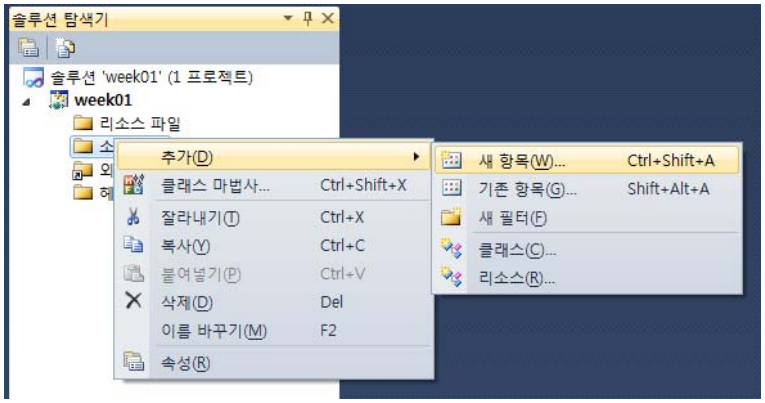
4.



Trouble Shooting

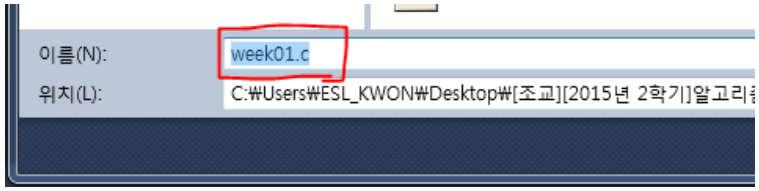
Visual Studio 2010 프로젝트 생성

5.



6.

7.



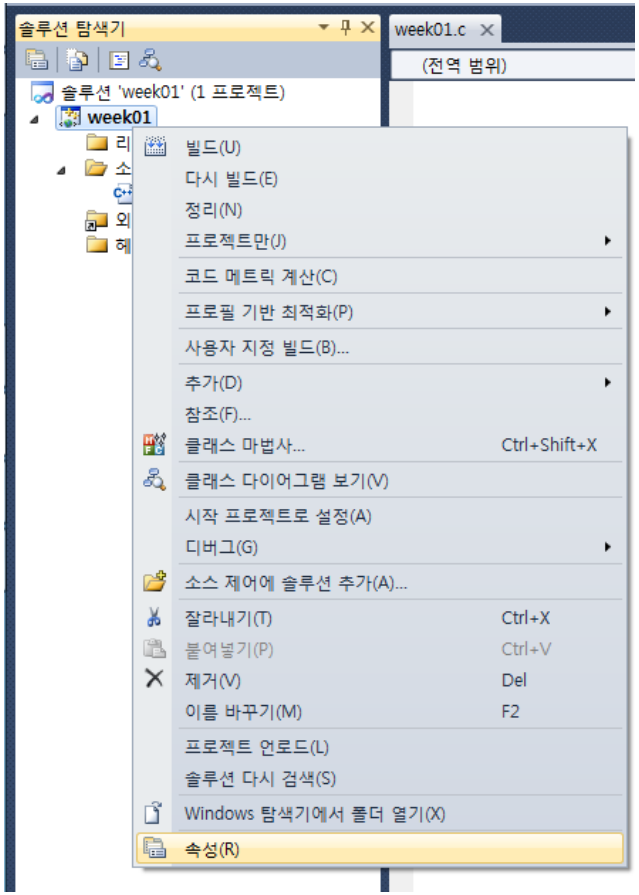
반드시 .c 로 이름 변경!!

Trouble Shooting

Visual Studio 2010 메니페스트 오류 해결

```
1>LINK : fatal error LNK1123: COFF로 변환하는 동안 오류가 발생했습니다. 파일이 잘못되었거나 손상되었습니다.
1>
1>빌드하지 못했습니다.
```

1.

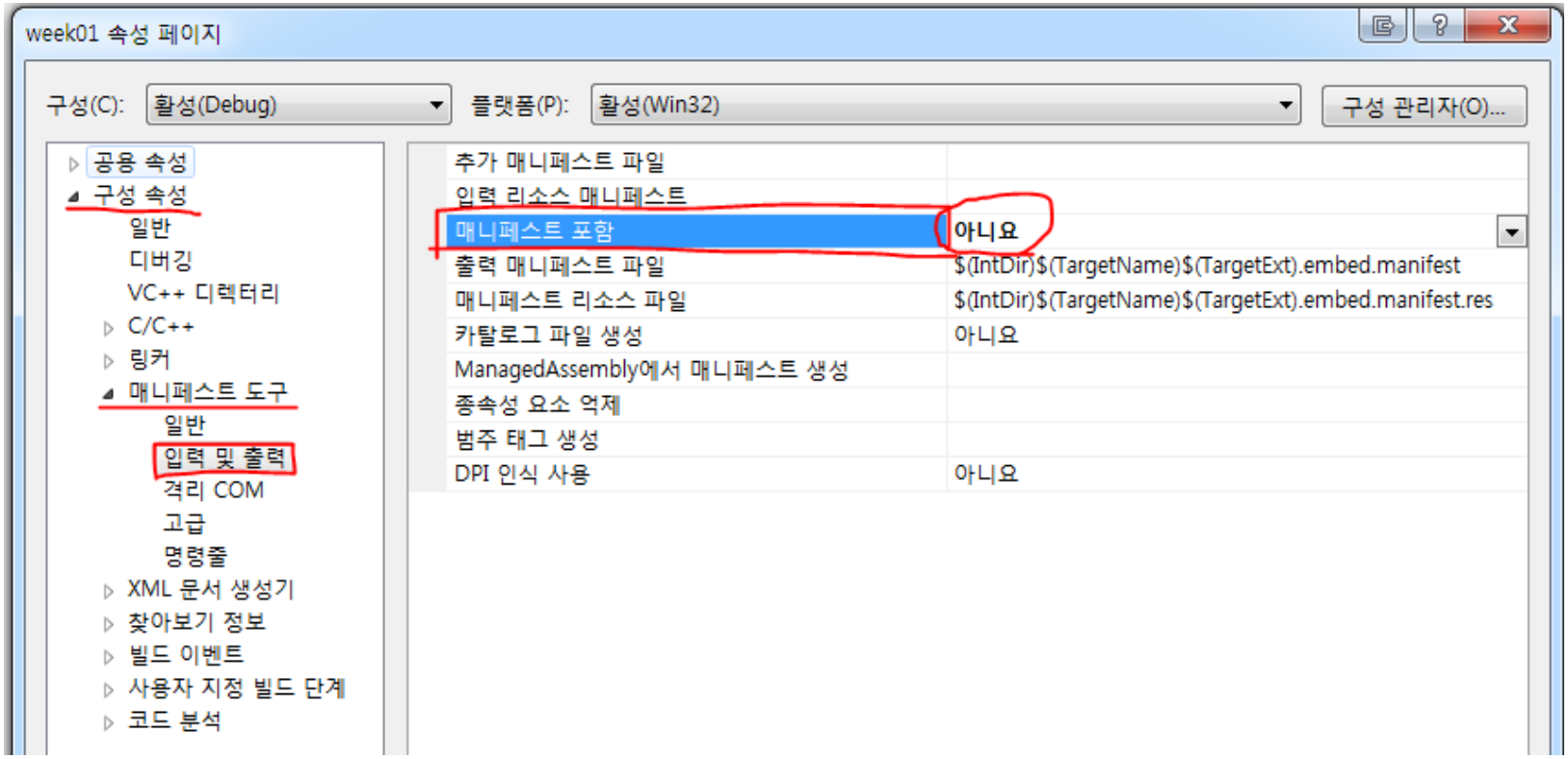


< 프로젝트 속성열기

Trouble Shooting

Visual Studio 2010 메니페스트 오류 해결

2.



구성 속성 -> 매니페스트 도구 -> 입력 및 출력 ->

메니페스트포함 : “아니요 "