



# D.COM ALGORITHM STUDY

# 9강. 정렬

# 오늘의 목표

정렬을 알아봅시다.  
단, 직접 구현해보는 것이 아니라  
STL을 사용합니다!

#쉬어 가는 단위입니다...!

#정렬을 직접 구현하는 법은 추후 강의에서 다룹니다!

가볍게,  
읽을거리



링크

## 일일 커밋의 효용성

“결과가 눈에 보이는 지표를 찾자”

—  
정렬

Sort

# I 정렬

데이터를 원하는 순서대로 정리하는 것을 의미합니다.  
정렬이 왜 필요할까요?

# I 정렬

데이터를 정렬해두면, 보다 쉽고 빠르게  
원하는 작업을 수행할 수 있기 때문입니다.

물건 정리를 하는 이유와 같다고 보면 됩니다!

# I 정렬

버블 정렬, 퀵 정렬 등 수 많은 정렬 알고리즘이 있습니다.  
이들을 직접 구현하는 것은 추후 강의에서 다룹니다!  
이번 시간에는 C++ STL을 사용합니다.



단순 정렬

# Sort

STL을 사용하여  
간단하게 정렬을  
할 수 있습니다.

```
#include <algorithm>

int main()
{
    int a[1000];
    ...
    sort(a, a+n);
    ...
}
```

배열 sort

```
#include <algorithm>

int main()
{
    int a[1000];
    ...
    sort(a, a+n);
    ...
}
```

벡터 sort



**연습 문제**  
2751번 수 정렬하기 2

<https://www.acmicpc.net/problem/2751>

sort를 사용하면 매우 간단하게 구현할 수 있습니다!

좌표 정렬



연습 문제  
11650번 좌표 정렬하기

<https://www.acmicpc.net/problem/11650>

우선 문제를 읽어봅시다!  
(x,y)를 어떻게 정렬할 수 있을까요?  
다음 슬라이드에서 알아보시다!

이차원 좌표를 어떻게 정렬할 수 있을까요?  
두 가지 방법이 있습니다.

1. Pair 이용
2. 구조체 이용

# 1. Pair 이용

첫 번째 방법으로,  
pair에 값을 담은 뒤  
정렬을 하면 됩니다!



```
vector<pair<int, int>> v;  
...  
sort(a.begin(), a.end())  
...
```

## 2. 구조체 이용

두 번째 방법으로  
구조체와 비교함수를 만듭니다.

sort()의 세번째 인자 값으로 앞서 만든  
비교함수를 넣게 되면, 해당 함수의 반환 값에  
맞춰 정렬이 동작합니다.

```
struct Point {  
    int x, y;  
};  
  
bool compare (const Point &a, const Point &b) {  
    return a.x != b.x ? a.x < b.x : a.y < b.y;  
}  
  
...  
  
sort(a.begin(), a.end(), compare);
```

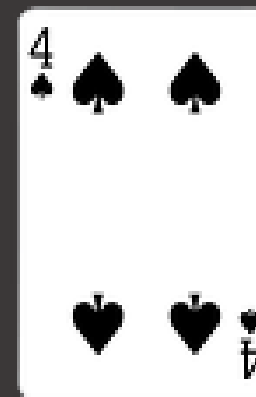
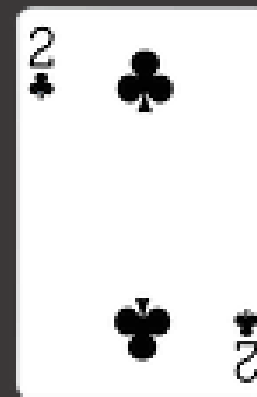
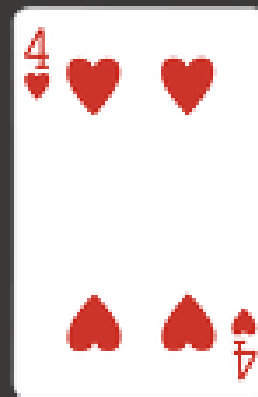
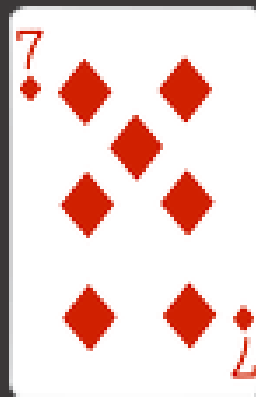


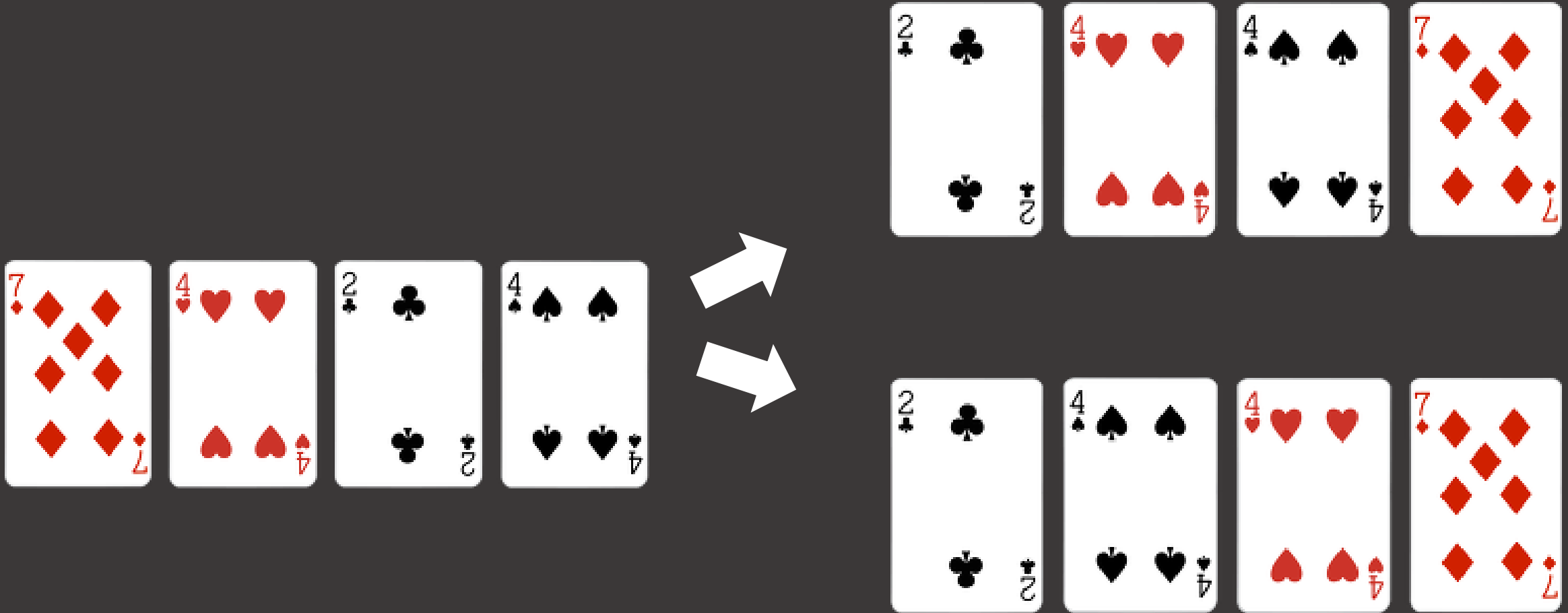
# 안정 정렬

# Stable Sort

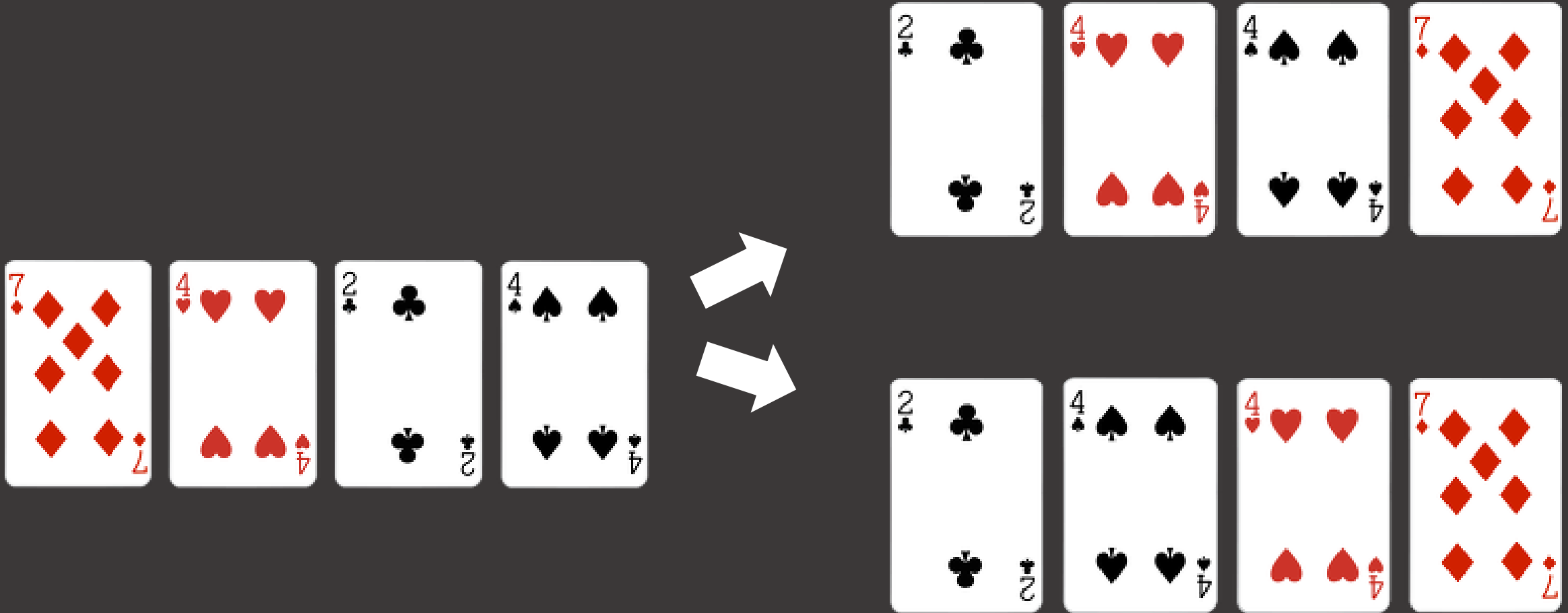
**안정 정렬이란,**  
**같은 키 값을 가지는 2개 이상의 데이터를 정렬할 때**  
**그 순서가 뒤바뀌지 않는 정렬을 의미합니다.**  
**다음 슬라이드에서 예시로 알아보시다!**

다음 카드를 숫자를 내림차 순으로  
정렬하고 싶습니다.  
카드에는 숫자와 문자 두 가지 값이  
있습니다.

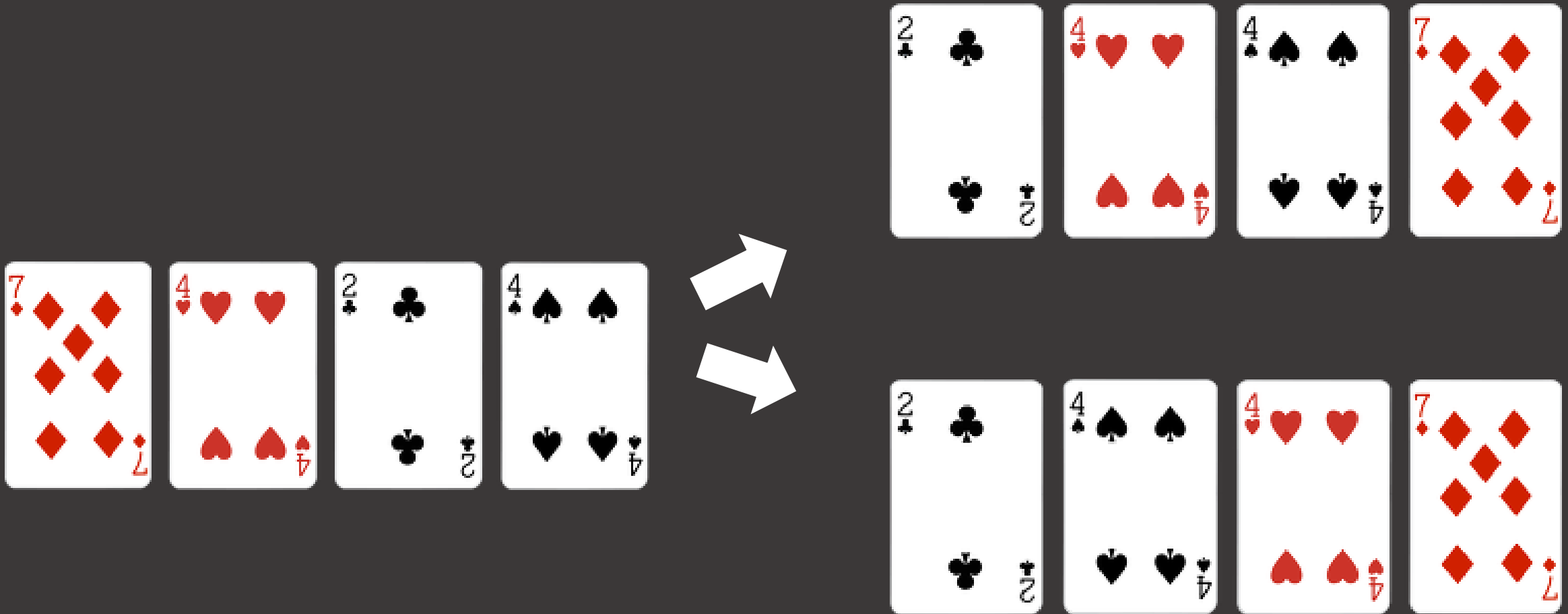




위와 같이 두가지 경우로 나눌 수 있습니다.  
주목해야 할 부분은 숫자 4 카드들 입니다.




위의 경우 카드의 그림의 순서가 유지 되었고,  
아래의 경우 카드의 그림의 순서가 유지 되지 않았습니다.



위의 경우를 안정 정렬(Stable sort)  
아래의 경우를 불안정 정렬(Unstable sort)라고 합니다.

삽입 정렬, 합병 정렬 등이 안정 정렬(Stable sort)이며,  
힙 정렬, 퀵 정렬 등이 불안정 정렬입니다.

STL에서는 `stable_sort()`를 제공합니다.



```
struct Data
{
    int age;
    string name;
};
bool cmp(const Data &u, const Data &v)
{
    return u.age < v.age;
}
...
stable_sort(d.begin(), d.end(), cmp);
...
```



연습 문제  
10814번 나이순정렬

<https://www.acmicpc.net/problem/10814>

왜 안정 정렬을 사용해야 할까요?  
위 문제를 통해 알아보시다!



읽을거리



링크

## 정렬 알고리즘 요약 정리

여러가지 정렬 알고리즘들을 정리해둔 글입니다.  
앞으로 알고리즘 문제나 구현에서 수도 없이 정렬을 마주치게 됩니다.  
몇가지 정렬 알고리즘 정도는 구현할 수 있어야  
큰 도움이 됩니다.

# 정리

이번 시간에는  
**정렬**에 대해 알아보았습니다!

1. 기본 정렬
2. 좌표 정렬
3. 안정 정렬

정렬은 컴퓨터 과학에서 매우 중요한 개념입니다.  
직접 구현하는 방법은 추후 강의자료에서도 지겹도록 다룰 예정이니,  
이번 시간에는 STL의 정렬 사용법을 알아두면 됩니다!

# The End

수고하셨습니다!