2주차

목차

- 공부 잘 해오셨나요?
  - String
- Sort

String (클래스임)

- 헤더파일 <string> 필요
- 선언 방법
  - String str(“FUCK STRING”);
  - String str;
    - str=“FUCK STRING”;
- C언어 문자열 맨 끝에 있는 NULL은 String에는 없다.

String 활용

- string Str = “Hello”; 라고 선언함.
- Str[0] -> H / Str.at(0) -> H : I번째 원소값 접근 (배열처럼 생각)
- Str.size()/str.length() -> 5 : 크기 반환
- Str.empty() -> 비어있으면 true/false 반환
  - If(str.empty() == true) 이런식으로 활용가능
- cout << Str; 가능
- Str+=‘A’ 하면 뒤에 붙혀짐
- Str.substr(start,end) [start,end)범위의 string 반환
- 문자열 분석-10820,알파벳 찾기 - 10809

반응 재미없으면 넘어갈 슬라이드

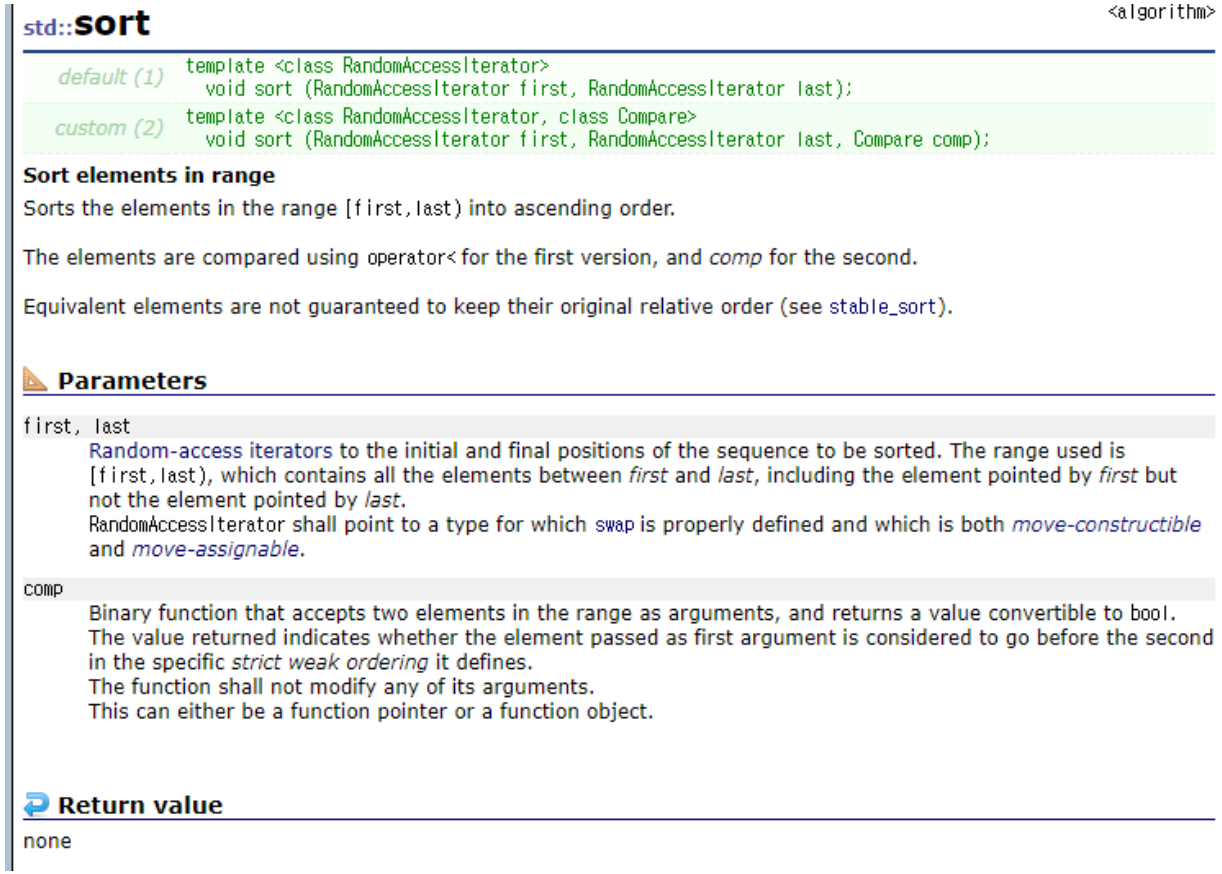
- 왜 컴퓨터는 [],() 범위가 아닌 [ )범위일까? + 왜 0부터 시작?
  - 다익스트라 라는 사람의 노트를 참고하면 : http://www.cs.utexas.edu/users/EWD/ewd08xx/EWD831.PDF
- 보기가 좋다

1. [a,b) 사용할 때 마지막 수가 N이라서 깔끔
   1. |b-a| 값이 N인 것이 반 닫힌구간 이다.
2. 그럼 [a,b) (a,b]중에 어떤 게 더 좋은가?
   1. 0부터 시작할때 (a 인경우는 -1<이렇게 적어야하는데 보기싫음
3. 실제로 Mesa라는 프로그래밍 언어에서 [a,b) 표기법이 좋다는게 증명됨.

코딩하면서 왜 이렇게 하지? 라는 생각이 들면 꼭 다 찾아보자.
면접할 때 물어보면 사랑받을 수 있다.

Sort - #include <algorithm> 필요

- Cplusplus.com 에 sort를 쳐보면 다음과 같이 나온다.



std::**sort** <algorithm>

default (1)
```
template <class RandomAccessIterator>
  void sort (RandomAccessIterator first, RandomAccessIterator last);
```

custom (2)
```
template <class RandomAccessIterator, class Compare>
  void sort (RandomAccessIterator first, RandomAccessIterator last, Compare comp);
```

Sort elements in range

Sorts the elements in the range [first,last) into ascending order.

The elements are compared using operator< for the first version, and *comp* for the second.

Equivalent elements are not guaranteed to keep their original relative order (see stable_sort).

Parameters

first, last
Random-access iterators to the initial and final positions of the sequence to be sorted. The range used is [first,last), which contains all the elements between *first* and *last*, including the element pointed by *first* but not the element pointed by *last*.
RandomAccessIterator shall point to a type for which swap is properly defined and which is both *move-constructible* and *move-assignable*.

comp
Binary function that accepts two elements in the range as arguments, and returns a value convertible to bool. The value returned indicates whether the element passed as first argument is considered to go before the second in the specific *strict weak ordering* it defines.
The function shall not modify any of its arguments.
This can either be a function pointer or a function object.

Return value

none

3줄요약
1.[first,second) 범위까지 오름차순 정렬해줌
2.sort(fist,second)로 쓰면됨
3.Cmp 사용가능 (뒤에 설명)

Sort

```cpp
#include <algorithm>
#include <iostream>
using namespace std;
int main() {
    int arr[10] = {5,4,2,6,1,3,8,7,9,10};
    cout << "Before Sorting \n";
    for(int i : arr) {
        cout << i <<" ";
    }
    cout <<"\n";
    sort(arr, _Last: arr+10);    //배열 크기가 10
    cout << "After Sorting \n";
    for(int i : arr){
        cout << i <<" ";
    }
    return 0;
```

이러한 방식으로
Sort(시작점, 어디까지) 작성함.
예를 들어서 sort(arr+2,arr+10) 이런 식으로도 가능.
배열 이름과 주소 값과 동치니까 ^^ (모르면 반성)

```
C:\Users\Juhyuk\CLionProjects\boj\cmake-build-debug\boj.exe

Before Sorting

5 4 2 6 1 3 8 7 9 10

After Sorting

1 2 3 4 5 6 7 8 9 10

Process finished with exit code 0
```

Sort

- 학교에서 배운 Sorting 방식을 이야기해보자
- (참고)퀵소트는 O(nlogn)이라고 하면 혼난다. N^2 이다.
  - 왜 merge sort는 모든 경우 nlogn인데 애매한 퀵소트를 쓸까?
    - Big-O표기법의 숨겨진 상수항이 꽤 좋아서. 그래서 대부분 퀵소트가 더 빠름
    - 역순일때만 N^2 나머지는 다 빠름
- STL Sort는 nlogn을 보장 해 준다.
  - 문제를 풀어보도록 하자
    - 수 정렬하기1,수 정렬하기 2
    - 단어정렬 – 1181
    - 수 정렬하기 3

Sort

- 이건 코드 작성하는 것을 같이 보고, 집에서 각자 풀어보자
  - 좌표 정렬하기 - 11650