20191031 실습 6주차

박 건 호

KITRI BoB 8th

5422 RTDCS

devgunho.github.io

Review

STL?

STL 은 Standard Template Library 라고 불리우며
C++ 에서 사용할 수 있는 컨테이너(container) class 와
알고리즘 일반화 시켜서 사용할 수 있는 자료구조 등을 포함하는 라이브러리의 모음이다.
STL 은 흔히 generic library(일반화된 라이브러리) 라고 불리우며,
이러한 일반화를 구현하기 위해서 C++ 에서 제공하는 Template(템플릿) 을 사용하고 있다.

STL 알고 사용하기

장점

소스코드의 크기를 줄일 수 있다. 이미 STL에는 자주 사용되는 50여개 정도의 알고리즘과 다양한 데이터 구조들을 가지고 있다. 우리는 이러한 STL에서 사용하는 자료구조와 알고리즘을 이용해서 코드의 크기를 획기적으로 줄일 수 있다.

STL 알고리즘과 컨테이너들은 기존의 C/C++ 의 포인터와 배열에도 사용할 수 있으며, 변환 가능하므로 유연하게 사용이 가능하다.
그리고 컨테이너와 알고리즘이 서로 분리되어 있음으로(많은 경우), 자신이 알고리즘을 만들어서 컨테이너에 적용시키는 작업이 가능하다.

대부분의 알고리즘과 자료구조들은 충분히 테스트되고 최적화 되어있다.

STL 알고 사용하기

단점

우선 디버깅이 어렵다. STL을 잘못 사용할 경우 발생되는 컴파일시의 에러코드는
템플릿 깊은 곳에 있는 내부함수들의 에러들을 출력시킨다.
이러한 에러가 발생하면 일단 에러메시지의 양들도 많을 뿐더러,
이해자체가 어려운 메시지들이다.

STL이 유연성을 구현하기 위한 템플릿의 경우, 프로그램의 크기를 매우 크게 만들 수 있다.
컴파일러가 템플릿을 그리 효율적으로 처리하지 못하기 때문이다.
역시 비용을 잘 고려하여서 효율적으로 STL의 알고리즘과 컨테이너들을 사용해야 한다.

반복자(일종의 포인터)와 컨테이너가 분리되어서 존재한다.
이는 Thread Programming 시 문제를 발생할 소지를 가지고있다.

Introduction to std::array (since C++11)

고정 배열(fixed array)과 동적 배열(dynamic array)을 배웠다.
두 가지 배열 모두 C++에 내장되어 있지만,
포인터로 형 변환되었을 시 배열 길이 정보가 손실되고,
동적 배열은 지저분한 할당 해제 문제가 있다.

이러한 문제를 해결하기 위해 C++ 표준라이브러리는 배열 관리를 쉽게 해주는
std::array 와 std::vector가 있다.

C++ 11에서 소개된 std::array는 함수에 전달할 때
포인터로 형 변환되지 않는 고정 길이 배열이다.
std:: array는 <array> 헤더의 std 네임 스페이스 내부에 정의되어 있다.

https://en.cppreference.com/w/cpp/container/array

Introduction to std::vector (since C++98)

std::vector는 자체 메모리 관리를 처리하는 동적 배열 기능을 제공한다.
즉, new와 delete를 사용하여 메모리를 동적으로 할당, 해제하지 않고도
런타임에 길이가 설정된 배열을 만들 수 있다.

std::vector는 <vector> 헤더에 정의되어 있다.

std::vector는 요청에 따라 해당 내용에 대해 동적으로 메모리를 할당하기 때문에
초기화에 상관없이 컴파일 타임에 배열 길이를 명시적으로 설정할 필요가 없다.

초기화 리스트(initializer-list)를 사용해 std::vector에 값을 할당할 수 있다.

```
array = { 0, 1, 2, 3, 4 }; // array length is 5
array = { 9, 8, 7 }; // array length is 3
```

http://www.cplusplus.com/reference/vector/vector/vector/

1. ModifiedFibo (10)

- 입력 : n input1 input2
  - n번째 값 찾기 (n>=3) / 0이상의 자연수 2개를 입력 받아 벡터에 저장
- 출력 :

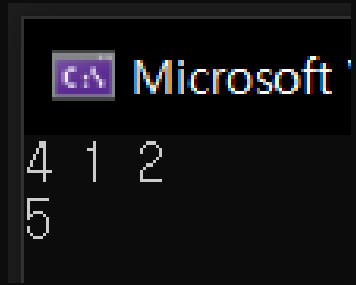
  vector의 1번째 요소 [index=0] 에는 input1,

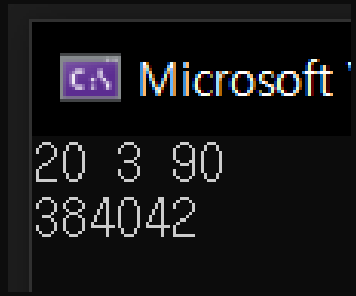
  vector의 2번째 요소 [index=1] 에는 input2,

  vector의 3번째 요소에는 input1+input2 이 저장된다.

  vector의 4번째 요소에는 input2+input3 이 저장된다.

  …





2. ExerciseVector (15)

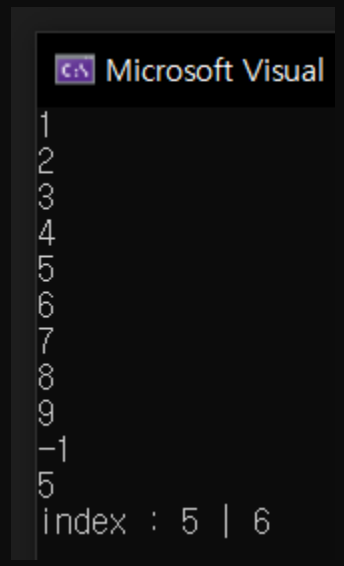
- 입력 : -1이 입력될 때까지 자연수를 입력 받고, 이후 다음 입력값에 따라 결과를 출력한다.
- 출력 :

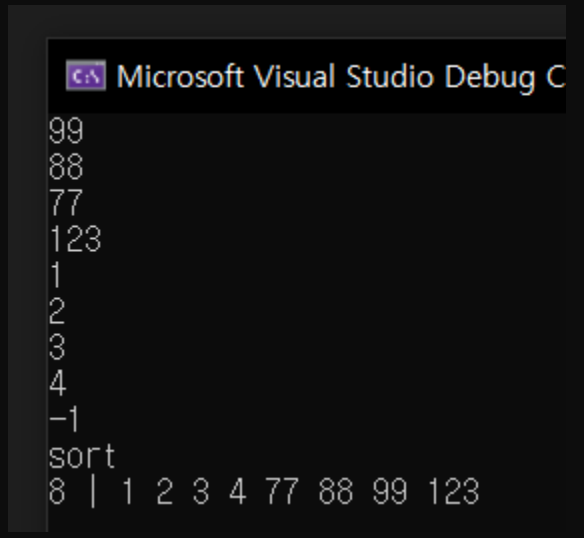
숫자 인 경우, 해당 인덱스의 값을 출력,

문자인 경우,

"pop" > pop 연산 이후 사이즈와 마지막 원소 출력

"sort" > sort 연산 이후 정렬된 원소들을 모두 출력





3. SortString (15)

알파벳 소문자로 이루어진 단어가 한 줄에 하나씩 주어진다.
주어지는 문자열의 길이는 50을 넘지 않는다.
END가 입력되기 전까지 입력받는다.

길이가 짧은 단어부터
길이가 같으면 사전순으로
같은 단어가 여러 번 입력된 경우 한번 씩만 출력하기

```
Microsoft Visua
a
b
c
d
d
asdf
sad
sadf
sadf
sdaf
safd
sadf
sad
fsadf
sadf
asdf
asdf
asdf
asdf
asdf
END
a
b
c
d
sad
asdf
sadf
safd
sdaf
fsadf
```

```
Microsoft V
aaaaa
a
a
a
aaaa
a
aaa
aaba
ab
ab
abbb
END
a
ab
aaa
aaaa
aaba
abbb
aaaaa
```

```
Microsoft
END
```

4. CalTopNPercentage (15)

인원에 따른 백분율을 구하여라 (점수에 따른 백분율이 아님을 주의할 것!)

동점자의 경우 백분율을 다음과 같이 적용.

ex) 97, 96, 95, 94, 93 => 0%, 25%, 50%, 75%, 100%

ex) 97, 95, 95, 95, 93 => 0%, 75%, 75%, 75%, 100%

// 95점으로 동점인 경우 3명 모두 상위 75%

- 입력 (0<=점수<=10000)
  - 사람 수 n
  - 이름1 점수
  - 이름2 점수
  - ...
- 출력 : 이름 점수 백분율(소수 2번째 자리까지)
  - 내림차순 출력

Person class 요구사항

private:

string 이름, int 점수

public:

생성자, string get_name(), int get_score()