STL

- vector -

STL(Standard Template Library)이란?

C++ 라이브러리의 일부분으로,

프로그램에 필요한 자료구조와 알고리즘을 템플릿으로 제공하는 라이브러리

STL의 구성 요소

컨테이너(Container)

- **객체를 저장하는 객체로 컬렉션 혹은 자료구조**

  - 시퀀스 컨테이너: vector, deque, list
  - 연관 컨테이너: set, multiset, map, multimap
  - 배열 기반 컨테이너: vector, deque
  - 노드 기반 컨테이너: list, set, multiset, map, multimap

STL의 구성 요소

알고리즘(Algorithm)

- **정렬, 삭제, 검색, 연산 등을 해결하는 일반화된 방법을 제공하는 함수 템플릿**

  - 원소를 읽기만 하는 알고리즘: adjacent_find(b,e), count(b,e,x), find(b,e,x) …
  - 원소를 수정하는 알고리즘: copy(b,e,t), fill(b,e,x), generate(b,e,f), iter_swap(p,q) …
  - 제거 알고리즘: remove(b,e,x), unique(b,e), unique_copy(b,e,t)…
  - 변경 알고리즘: next_permutation(b,e), partition(b,e,f), random_shuffle(b,e,f) …
  - 정렬 알고리즘: make_heap(b,e), sort(b,e), nth_element(b,m,e)…
  - 정렬된 범위 알고리즘: binary_search(b,e,x), includes(b,e,b2,e2), upper_bound(b,e,x) …
  - 수치 알고리즘: accumulate(b,e,x), inner_product(b,e,b2,x) …

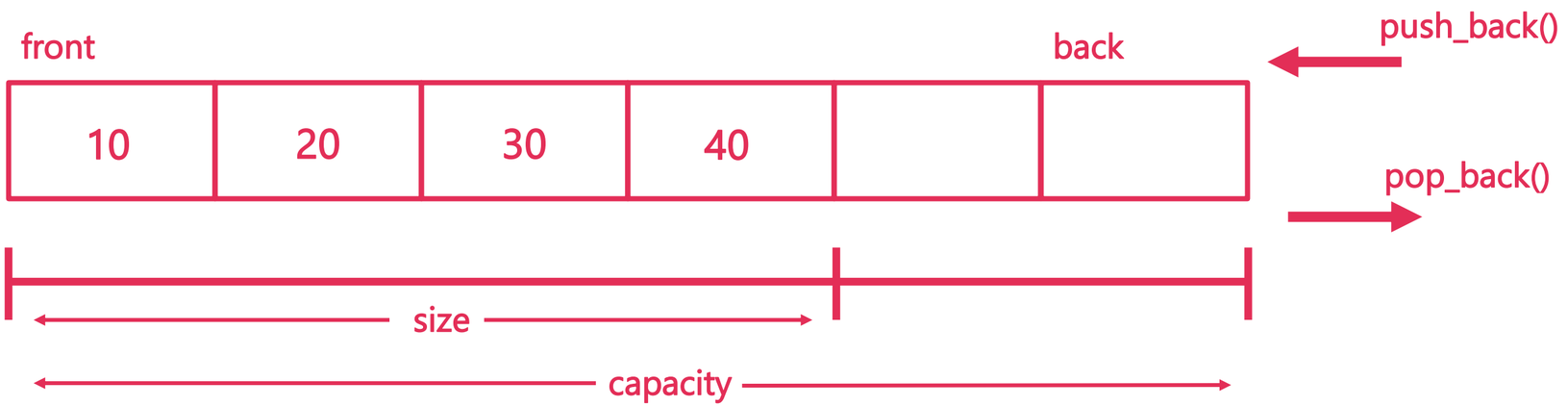
STL의 구성 요소

반복자(iterator)

- 포인터와 비슷한 개념으로 컨테이너의 원소를 가리키고,
  가리키는 원소에 접근하여 다음 원소를 가리키게 하는 기능

vector란?

배열 기반, 시퀀스 컨테이너로 객체(원소)들을 한 메모리 단위에 순차적으로 저장하는 가변길이 배열



vector 사용법

vector의 초기화(생성자)

- vector <type> v
  : 빈 컨테이너

- vector <type> v(n)
  : 0으로 초기화된 n개의 원소를 가진 컨테이너

- vector <type> v(n,x)
  : x값으로 초기화된 n개의 원소를 가진 컨테이너

- vector <type> v(v2)
  : v는 v2 컨테이너의 복사본

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main()
{
    // 빈 컨테이너
    vector<int> v1;

    // v2[0]=0, v2[1]=0, v2[2]=0
    vector<int> v2(3);

    // v3[0]=1, v3[1]=1, v3[2]=1
    vector<int> v3(3,1);

    // v4[0]=1, v4[1]=1, v4[2]=1
    vector<int> v4(v3);

    return 0;
}
```

vector 사용법

vector 요소 접근

- v.front()
  : v의 첫 번째 원소를 참조한다.

- v.back()
  : v의 마지막 원소를 참조한다.

- v.at(i)
  : v의 i번째 원소를 참조한다.

- v[i]
  : v의 i번째 원소를 참조한다.

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main()
{
    // v[0]=1, v[1]=1, v[2]=1
    vector<int> v(3,1)

    // v[0]=100
    v[0]=100;

    // 100 출력
    cout<<v.at(0)<<endl;
    cout<<v.front()<<endl;

    // v[2]=300
    v[2]=300;

    // 300 출력
    cout<<v.back()<<endl;

    return 0;
}
```

vector 사용법

vector 요소 접근

- v.begin()
  : v의 첫 원소를 가리키는 반복자

- v.end()
  : v의 끝을 가리키는 반복자

- *연산자로 원소에 접근할 수 있다.

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main()
{
    // v[0]=1, v[1]=1, v[2]=1
    vector<int> v(3,1);

    // v[0]=100
    v[0]=100;
    v[1]=200;
    v[2]=300;

    // 반복자 선언
    vector<int>::iterator iter=v.begin();
    vector<int>::iterator iter2=v.end();

    // 100 출력
    cout<<*iter<<endl;

    // 300 출력
    iter2=iter2-1;
    cout<<*iter2<<endl;

    // 200 출력
    ++iter;
    cout<<*iter<<endl;

    return 0;
}
```

vector 사용법

vector 요소 삽입 및 삭제

- v.push_back(x)
  : v의 끝에 x를 추가한다.

- v.pop_back()
  : v의 마지막 원소를 제거한다.

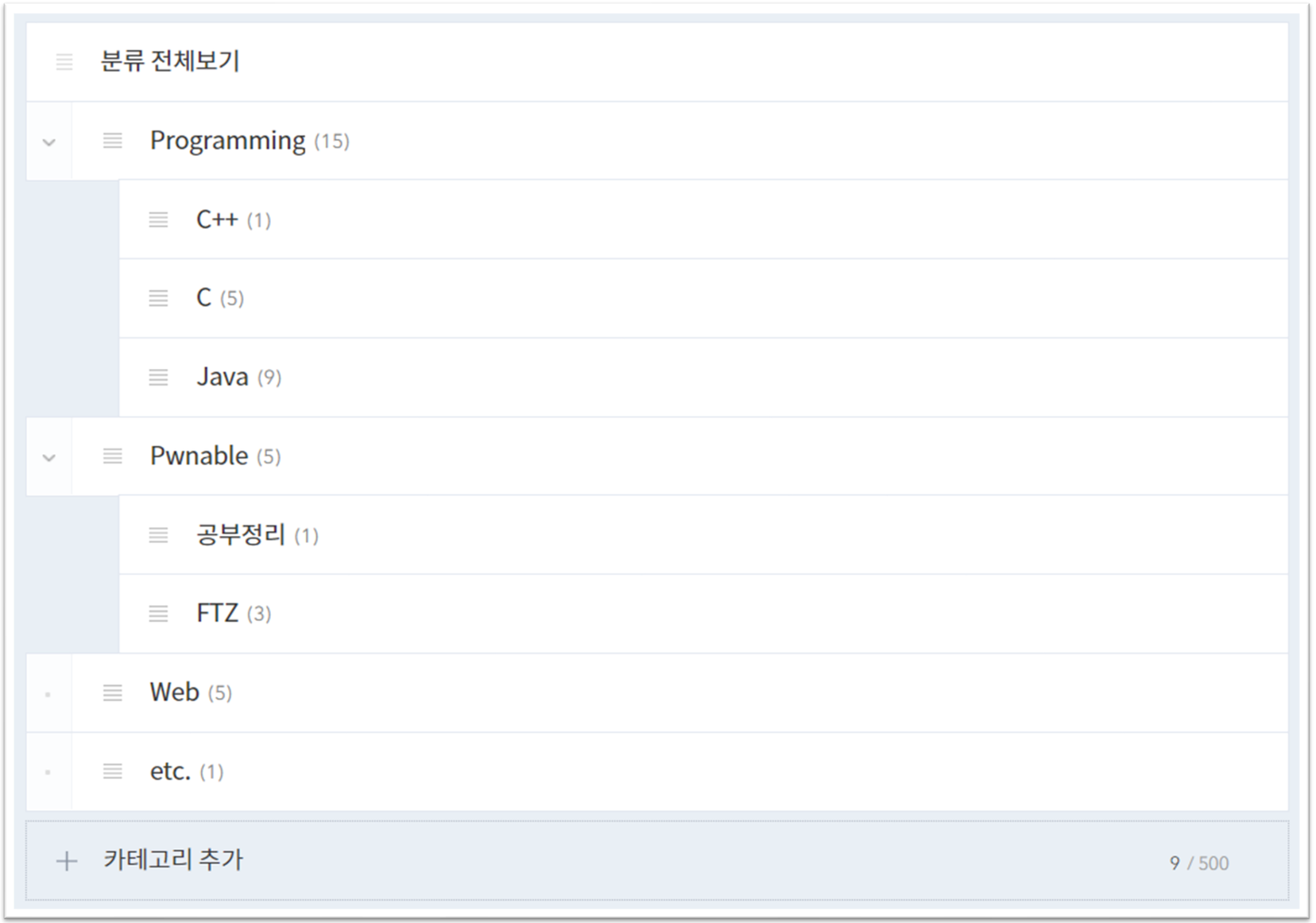
- v.insert(p, x)
  : p가 가리키는 위치에 x 값을 삽입한다.

- v.erase(p)
  : p가 가리키는 원소를 제거한다.

- v.clear()
  : v의 모든 원소를 제거한다.

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main()
{
    vector<int> v;

    // v[0]=100, v[1]=200
    v.push_back(100);
    v.push_back(200);

    // 반복자 선언
    vector<int>::iterator iter=v.begin()+1;

    // v[0]=100, v[1]=300. v[2]=200
    iter2=v.insert(iter, 300);

    // v[0]=100, v[1]=300
    v.pop_back();

    // v[0]=100
    v.erase(iter2);

    v.clear();

    return 0;
}
```

vector?


분류 전체보기
Programming (15)
C++ (1)
C (5)
Java (9)
Pwnable (5)
공부정리 (1)
FTZ (3)
Web (5)
etc. (1)
카테고리 추가
9 / 500

vector?

<사용 전>

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main(){
    int i, size=3;
    char ** arr = (char **)malloc(size*sizeof(char*));
    for(i=0; i<size; i++){
        arr[i] = (char*)malloc(20*sizeof(char));
    }
    arr[0]="Programming";
    arr[1]="Pwnable";
    arr[2]="Web";

    for(i=0; i<size; i++){
        printf("%s\n",arr[i]);
    }

    int num, j=0;
    char menu[20];
    scanf("%d %s", &num, menu);
    size++;
    arr = (char **)realloc(arr, size*sizeof(char*));
    while(1) {
        j++;
        if (j==num+1) break;
        arr[size-j]=arr[size-(j+1)];
    }
    arr[size-j]=menu;
}
```

// 새 카테고리 추가

<사용 후>

```c
#include <iostream>
#include <vector>
using namespace std;

int main(){
    vector<string> v;
    v.push_back("Programming");
    v.push_back("Pwnable");
    v.push_back("Web");

    vector<string>::iterator iter;

    int num;
    string menu;
    cin>>num>>menu;
    vector<string>::iterator iter2=v.begin()+num-1;
    v.insert(iter2, menu);
}
```

vector?

// 마지막 카테고리 삭제

<사용 전>

```
31      size--;
32      arr = (char **)realloc(arr, size*sizeof(char*));
33
34      for(int i=0; i<size; i++){
35          printf("%s", arr[i]);
36      }
37  }
```

<사용 후>

```
27      v.pop_back();
```

vector?

// 모든 카테고리 출력

<사용 전>

```
34          for(int i=0; i<size; i++){
35              printf("%s", arr[i]);
36          }
37      }
```

<사용 후>

```
28
29          for(iter=v.begin(); iter!=v.end(); iter++){
30              cout<<*iter<<endl;
31          }
32      }
33
```