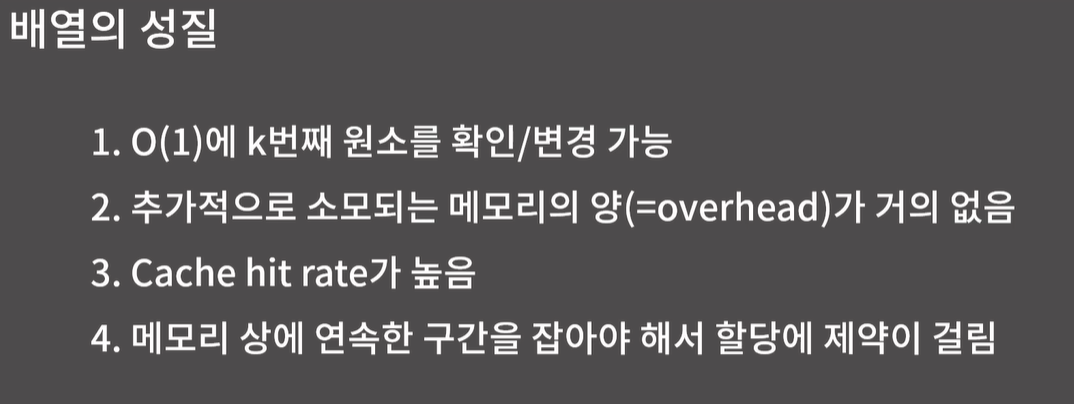
Chap(3).배열

배열 – 메모리 상에 원소를 연속하게 배치한 자료구조

배열의 성질

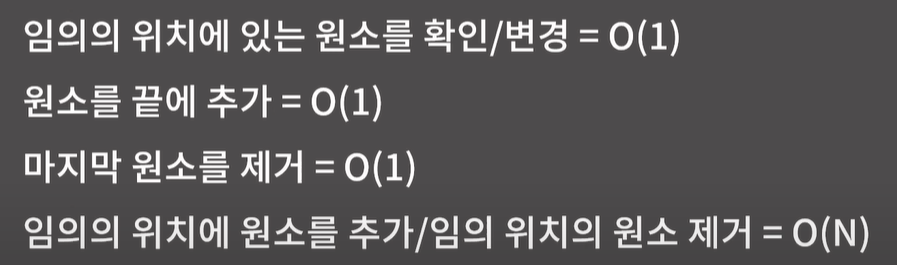


1. 🡪 인덱스를 통해 원소를 찾아갈 수 있기 때문에 한 번의 연산으로 원소 확인 가능

3.--> CPU기 메인 메모리에서 한 개의 데이터를 접근하게 되면 그 근처의 연속된 데이터들도 다음에 빨리 찾을 수 있도록 cache에 저장하게 되는데 그렇기 때문에 배열의 원소들을 접근할 때는 메인메모리까지 접근하지 않고 cache에서 hit가 되는 빈도가 높다

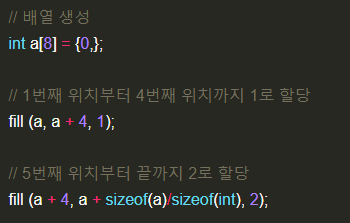
임의의 위치에 원소를 추가/제거 ( O(N) )

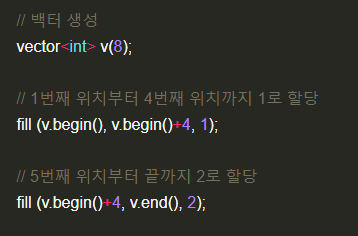
* 추가/제거한 위치 이후의 원소들을 하나씩 땡기거나 밀어야함



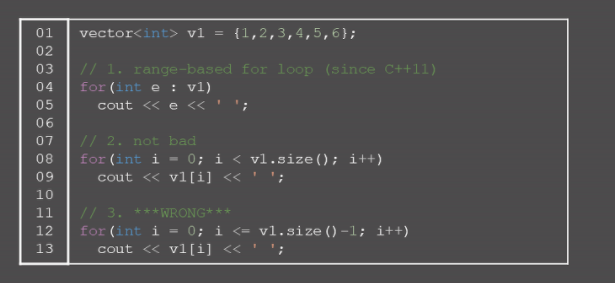
* Fill 함수
* Void fill(first, last, val);
* 헤더: #include<algorithm>
* Fisrt : 채우고자하는 자료구조의 시작위치
* Lase : 채우고자하는 자료구조의 마지막 위치(last는 포함 X)
* Val : First부터 Last 이전까지 채우고자 하는 값

<https://twpower.github.io/121-usage-of-fill-function>

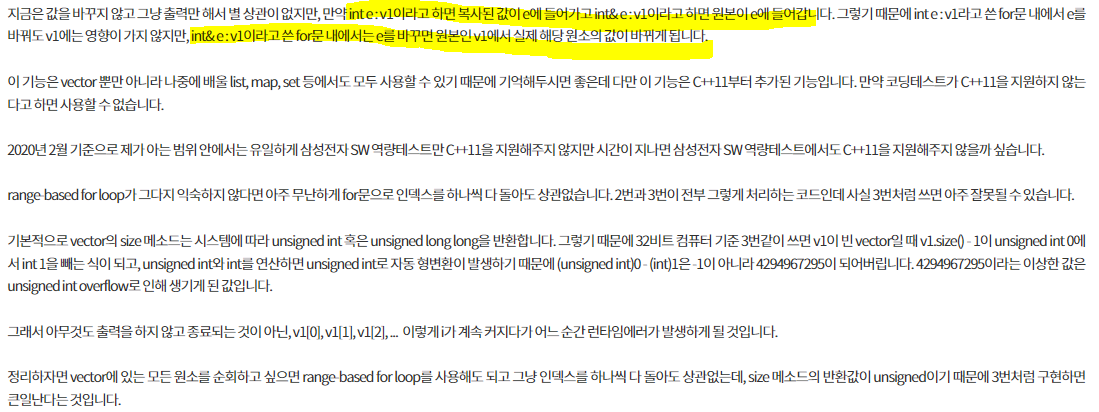




* Vector



04줄 for문 = range-based for loop

e에 v1의 원소들이 하나씩 들어가는 for문

**- string 문자열(c++)도 배열과 같이 인덱스로 접근이 가능하다!**

**- 문자&아스키 코드 비교 (no.10808)**

char a = 'a';

if(a == 97)

cout << true;

==> true 출력