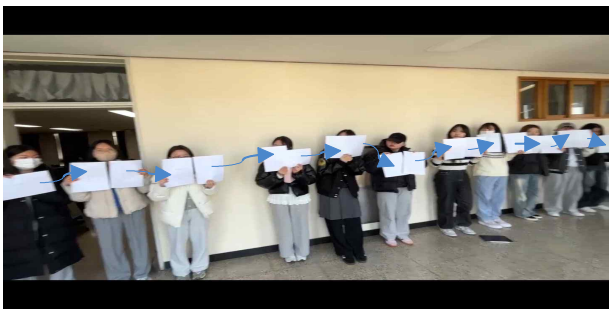
Data Structures - 2025 1st semester 20230523 김난영
Assignment 02

4번 연산 작동 과정

제일 처음 아무런 변화가 없을 때의 연결리스트.

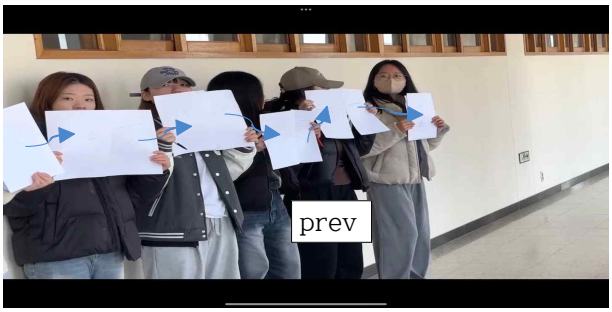
왼쪽에서 오른쪽 방향의 순서대로 헤드-> 첫 번째노드 -> 두 번째 노드...이렇게 연결된다.

a. prev가 첫 번째 노드를 가리키고, 두 번째 노드는 pop하니 빠진 뒤에 첫 번째 노드의 next는 세 번째 노드와 연결된다.



b. 헤드는 prev가 불가능하니 첫 번째 위치에 방금 삭제한 두 번째 노드를 insert하기 위해 두 번째 노드의 next에 첫 번째 노드를 연결하고, 헤드가 두 번째 노드를 가리키게 연결한다.

c. 첫 번째 위치에 두 번째 노드가 들어가니 두 번째 노드가 첫 번째 노드가 된다. 이번에도 헤드는 prev가 불가능하니 첫 번째 노드를 pop하기 위해 헤드는 두 번째 노드에 연결되고, 첫 번째 노드는 pop되어 빠진다.

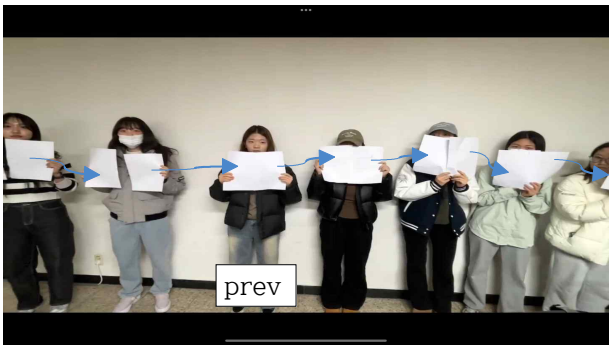


d. append는 연결 리스트의 맨 뒤에 추가하는 작업이니 prev가 제일 마지막 노드를 가리키고, 방금 삭제했던 첫 번째 노드가 마지막 노드의 뒤로 연결된다.

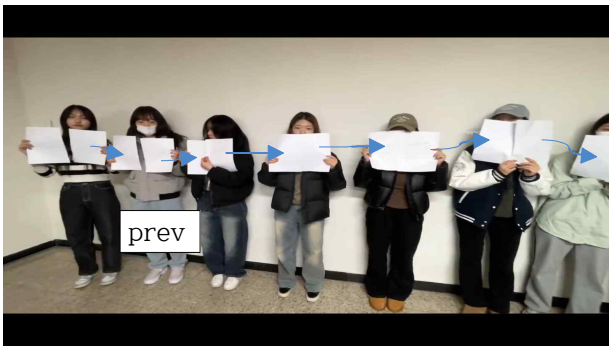
7번 연산 작동 과정

제일 처음 아무런 변화가 없을 때의 연결리스트.

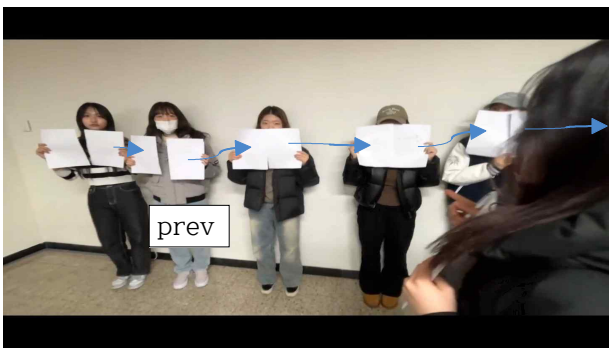
왼쪽에서 오른쪽 방향의 순서대로 헤드-> 더미 헤드 노드-> 첫 번째 노드-> 두 번째 노드-> 세 번째 노드...이렇게 연결된다.



a. prev가 첫 번째 노드를 가리키고, 두 번째 노드는 pop하니 빠진 뒤에 첫 번째 노드의 next는 세 번째 노드와 연결된다.



b. prev가 더미 헤드 노드를 가리키고, 첫 번째 위치에 방금 삭제한 두 번째 노드가 insert된다. 그러니 더미 헤드 노드는 insert된 두 번째 노드와 연결되고, 두 번째 노드의 next는 첫 번째 노드와 연결된다.



c. prev가 더미 헤드 노드를 가리키고, 첫 번째 위치에 들어간 두 번째 노드가 이제 첫 번째 노드가 되었으니 그 노드를 pop해서 빠지게 한 뒤, 더미 헤드 노드는 pop해서 빠진 첫 번째 노드를 건너뛰고 그 다음 노드인 두 번째 노드에 연결된다.

d. prev가 두 번째 위치에 있는 노드를 가리키고, 방금 삭제된 첫 번째 노드가 insert된다. 그러니 두 번째 위치에 있는 노드의 next는 insert된 첫 번째 노드와 연결되고, 첫 번째 노드의 next는 네 번째 위치의 노드와 연결된다.

교재 3장(알고리즘 성능) 연습문제

1.
1) 참 -> 최고차항의 차수가 f(n)과 같다.
2) 거짓 -> 빅 오메가는 최고차항의 차수가 f(n)보다 크거나 같아야하는데 이 문제에선 더 작다.
3) 참 -> n^3보다 작은 차수인 n^2은 빅 오 표기법에서 포함됨.
4) 참 -> ($2n^2$ - 100n)은 결국 n^2의 차수와 동일시함.
5) 참 -> 로그 항은 제곱에 비해 빠르기 때문에 최고차항인 n^2으로 보면 f(n)과 같다.
6) 참 -> n^2은 당연히 n^{100}보다 작다.
7) 참 -> 로그 항은 n^2보다 빠름.
8) 참 -> 최고차항의 차수가 n^2로 동일하다.
9) 참 -> n^2 보다 n log n이 더 빠르다.
10) 거짓 -> 최고차항의 차수가 n으로 n^2과 동일하지 않다.

2. n에 비례하는 수행시간이 걸린다.
-> n이 크면 클수록 돌아가는 루프가 많아지기 때문에 n에 비례하는 시간이 걸린다.

3. ○-표기법 : ○(n^2) Ω-표기법 : Ω(n^2) ⊖-표기법 : ⊖(n^2)
-> 최선, 최악이 모두 같은 차수이기 때문에 세가지 모두 동일하다.

4. ⊖-표기법 : ⊖(n^3)
-> for i에서 n번 반복, for j에서 n번 반복, for k에서 n번 반복되고 이 3개가 다 중첩되어있으니 n^3

5. ○-표기법 : ○(n^2) Ω-표기법 : Ω(n)
-> for i는 무조건 n번 되지만 그 뒤의 if문은 random해서 나온 정수가 50보다 크냐에 따라 실행될지 말지가 정해지기 때문에 만약 if가 실행된다면 내부의 for i루프도 실행되기 때문에 이 경우가 n^2의 최악의 경우이고, if의 내부의 루프가 실행되지 않는 경우가 n의 최선의 경우이다.

6. ○-표기법 : ○(n) Ω-표기법 : Ω(1)
-> 함수 sample()을 무시하고 복잡도만 신경 쓴다면 최악의 경우 random이 항상 참이라고 가정하고, for i가 실행된다면 n에 비례한 수행시간이 걸릴 것이다. 그리고 random이 항상 거짓이라 for i가 한 번도 실행되지 않게 되는 경우가 최선의 경우가 된다.

7. 알고리즘의 점근적 수행시간 : ○(n^2)
-> if문은 바로 종료할 수 있는 조건이니 for문을 돌지 않고 끝나는 이 경우가 최선의 경우가 될 수 있다. 하지만 그렇지 않고 for문이 실행된다면 for I로 우선 한번 루프가 n만큼 돌게 될 것이며 재귀 호출로 인해서 한 번 더 루프가 돌기 때문에 최악의 경우 n^2에 비례한 수행시간이 걸릴 수 있다.