□ 함수 설명 및 시간 복잡도 \* 데이터가 존재 하는 경우에 시간복잡도

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **자료구조**  **함 수** | **ArrayList** | | **DoubleLinkedList** | |
| ListInit | 리스트의 데이터와 가리키고 있는  포인터 초기화 | | 리스트의 데이터와 head, tail 초기화 | |
| Insert\_Head | 배열의 첫 번째에 데이터 삽입 후  데이터의 개수 증가시킨다. | | 리스트 맨 앞에 데이터를 삽입한다.  이 데이터가 최초 삽입 노드이므로  head와 tail이 이 노드를 가리킨다. | |
| front | 0번째 삽입시 0번째부터 옮겨야하므로 ⨁(n) | front | 헤드가 새 노드 가리키게만 하면 되므로 O(1) |
| back | - | back | - |
| k | - | k | - |
| Find | 배열에서 찾는 데이터를 만날때까지  인덱스를 가리키는 curPosition을  이동시킨다. 찾으면 해당 데이터를  결과파일에 쓰고 못 찾을시 -1출력  curPosition이 배열에 들어있는 데이터의 수가 넘어간다면 찾는 데이터가 없다.  curPosition이동에 따른 수행시간 O(n) | | 임의의 노드를 만들어 리스트의 헤드를  가리키게 한 후 데이터를 찾을 때 까지  다음 노드를 탐색.  임의의 노드가 꼬리의 다음인 NULL이  된다면 찾기 실패  데이터 찾는데 수행시간 O(n) | |
| front | 0번째 index 조회 O(1) | front | head노드 조회 O(1) |
| back | 마지막 index 조회 O(1) | back | tail노드 조회 O(1) |
| k | k번째 index 조회 O(1) | k | k번째 노드 조회 O(1) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Insert\_Before | data1 앞에다가 data2 삽입.  리스트와 같은 내용의 배열을 생성해서  data1의 위치 인덱스부터 끝까지를  원래 리스트 data1의 위치부터 복사한다.  그 후 data1의 위치에 data2를 넣는다.  data1를 찾는데 n시간 data2삽입 후 옮기는데 n시간 | | data2의 노드를 생성해 data1의 노드의  prev노드와 data1의 노드사이에  넣어 연결시킨다.  data1의 노드가 헤드라면 data2를  헤드가 가리키게 만들고 data2의 prev를  NULL을 가리킨다.  data1를 찾는데 n시간 삽입하는데 상수시간 | |
| front | 0번째에 삽입 후 0번째부터 데이터 옮기는데 걸리는 n시간 ⨁(n) | front | head포인터를 새로운 노드를 가리키게 하면 되므로 ⨁(1) |
| back | 마지막 전 index에 삽입 후 마지막 데이터만  옮긴다. ⨁(1) | back | tail노드의 앞에 삽입 ⨁(1) |
| k | k번째 삽입 후 데이터 옮기는데 n-k시간 O(n) | k | k번째에 노드 삽입 ⨁(1) |
| Insert\_After | data1 뒤에다가 data2 삽입  리스트와 같은 내용의 배열을 생성해서  data1의 위치 다음 인덱스부터 끝까지를  원래 리스트 data1의 위치+2부터 복사한다.  그 후 data1+1의 위치에 data2를 넣는다.  data1를 찾는데 n시간 data2삽입 후 옮기는데 n시간 | | data2의 노드를 생성해 data1의 노드의  next노드와 data1의 노드사이에  넣어 연결시킨다.  data1의 노드가 tail이라면 data2를  tail이 가리키게 만들고 data2의 next를  NULL을 가리킨다.  data1를 찾는데 n시간 삽입하는데 상수시간 | |
| front | 1번째에 삽입 후 1번째부터 데이터 옮기는데 걸리는 n시간 ⨁(n) | front | head노드 뒤에 새로운 노드를 삽입 ⨁(1) |
| back | 마지막에 삽입 ⨁(1) | back | tail노드의 뒤에 삽입 ⨁(1) |
| k | k+1번째 삽입 후 데이터 옮기는데 n-k+1시간 O(n) | k | k번째에 노드 삽입 ⨁(1) |
| Erase | data의 다음위치부터 끝까지를 data의 위치부터  옮긴다. data는 data의 다음 data로 덮여진다.  data찾는데 n시간 삭제하고 덮는데 n시간 | | 삭제할 data의 노드가 한 개라면  head와 tail포인터가 NULL을 가리킨다.  한 개는 아니지만 head라면 다음 노드를  head가 가리키게 한다.  tail이라면 이전 노드를 tail이 가리키게 한다.  중간 노드면 삭제노드의 prev노드와 next노드를  연결시킨다.  data찾는데 n시간, 삭제하는데 상수시간 | |
| front | 0번째 데이터 삭제 후 옮기는 n시간 ⨁(n) | front | 헤드 노드 삭제 ⨁(1) |
| back | 마지막 데이터 삭제 ⨁(1) | back | 꼬리 노드 삭제 ⨁(1) |
| k | k+1번째부터 데이터 옮기는데 n-k+1시간 O(n) | k | k번째 노드 삭제 ⨁(1) |
| Print | 데이터 출력 | | 데이터 출력 | |
| Next | 다음 인덱스를 가리키고 출력  data의 인덱스가 마지막이라면 -1출력  data찾는데 걸리는 n시간 | | 다음 노드 데이터 출력  노드가 tail이라면 -1출력  data찾는데 걸리는 n시간 | |
| front | 1번째 데이터 조회 ⨁(1) | front | head 다음 노드 조회 ⨁(1) |
| back | - | back | - |
| k | k+1번째 데이터 조회 ⨁(1) | k | k번째 다음 노드 조회 ⨁(1) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Previous | 이전 인덱스를 가리키고 출력  data의 인덱스가 0이라면 -1출력  data찾는데 걸리는 n시간 | | 이전 노드 데이터 출력  노드가 head라면 -1출력  data찾는데 걸리는 n시간 | |
| front | - | front | - |
| back | n-1번째 데이터 조회 ⨁(1) | back | tail 이전 노드 조회 ⨁(1) |
| k | k-1번째 데이터 조회 ⨁(1) | k | k번째 이전 노드 조회 ⨁(1) |
| Exit | 프로그램 종료 | | 프로그램 종료 | |