Hashing 은 key 값을 이용하여 테이블의 주소를 계산하고 테이블의 주소로 데이터에 접근하는 것을 의미한다. Hashing 을 사용하면 1 번에 데이터를 탐색할 수 있어서 시간복잡도를 줄일 수 있다.

1.문제

1. (Programming: **50 points**)
   In p29-31 of 'DS-Lec11-Hashing', there is no function to delete the item in the hash table. Please implement '`hash_chain_delete(element item, ListNode *ht[])`', and then test the code with the following commands.

2. 문제해결

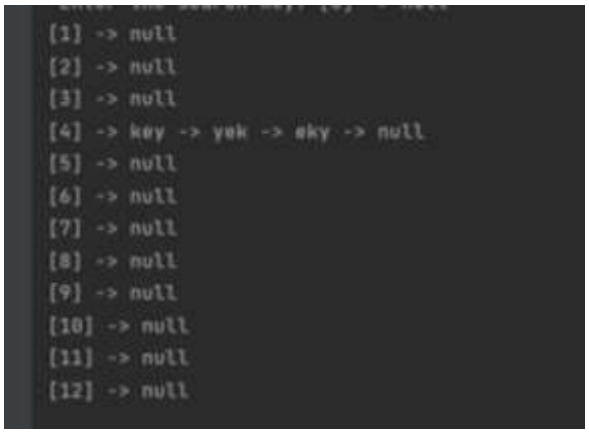
강의자료 코드에 없는 hash_delete 함수를 구현해보면, hash_function 함수로 해시주소를 구한다.

4 가지 경우를 생각해야 하는데 삭제할 node 를 기준으로 이전 node 가 있고 이후 node 가 있는 경우, 이전 node 가 있고 이후 node 가 없는 경우, 이전 node 가 없고 이후 node 가 있는 경우, 이전 node 도 없고 이후 node 도 없는 경우이다.

결과가 같은 상황을 찾아서 경우의 수를 줄여보면 이후 node 가 없는 node 는 NULL 값이 저장되어 있다. 이전 node 가 있고, Node->link 가 있는 경우와 없는 경우 모두 이전 node->link 에 node -> link 를 연결시켜준다.

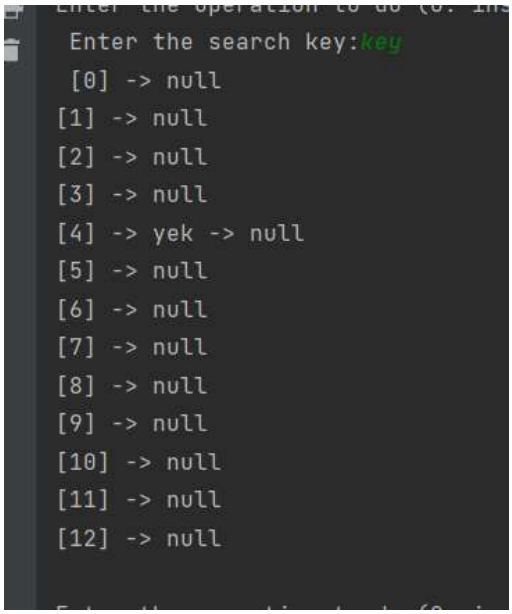
이전 노드가 없는 경우 hash array 에 해시주소 부분을 초기화 시키고 node->link 값을 넣는다.

실행결과

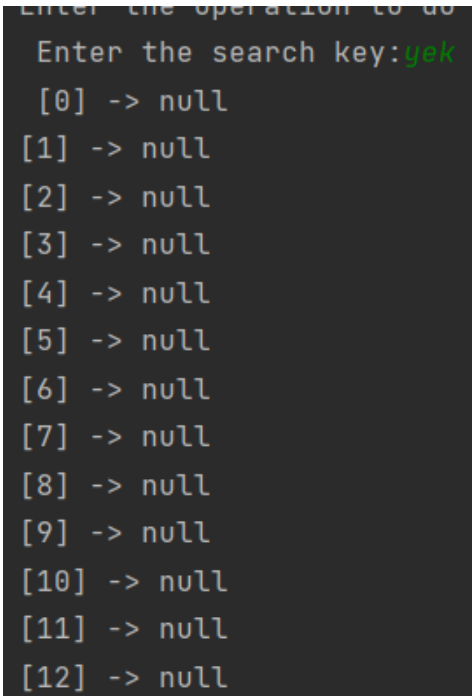


3 번째 값을 삭제하기 전

3 번째 값을 삭제한 후



2 번째 값을 삭제한 후



1 번째 값을 삭제한 후

2 번

Binary search 는 중간 값부터 값을 비교해 나가는 것이다. 찾는 값보다 low 면 left 중간값, high 면 right 중간 값부터 값을 비교한다. 이 알고리즘을 traversal tree 에도 적용할 수 있는데 root 에 중간 값을 넣고 이 값보다 크면 왼쪽 작으면 오른쪽으로 이동 후 root 값을 변경한다.

1. 문제

2. (Programming: **50 points**)
Sort an input data using the binary search tree.
- Generate an input array of size 1000 randomly.
- Insert elements in the input array iteratively into the binary search tree.
- Apply the tree traversal to print the sorted results.
(Think about what traversal should be used)

4

2. 문제해결

Binary search tree 로 input 값을 sort 할 수 있다. Tree root 값부터 내려가면서 값을 비교한다. Key 값이 node 값보다 작으면 왼쪽으로 가고 key 값이 node 값보다 크면 오른쪽으로 이동한다. Leaf node 면 그 child node 에 값을 저장한다.

3. 실행결과

```
38
22
4
9
25
29
41
8468
6335
5725
1479
492
154
107
54
56
68
59
73
76
94
81
143
141
124
113
117
```