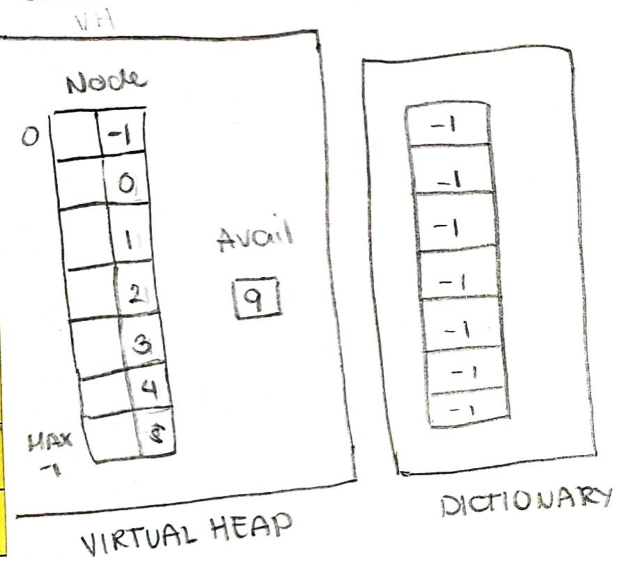
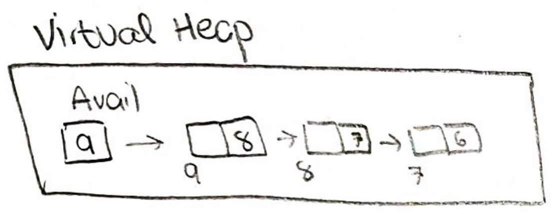
- OPEN HASHING -
CURSOR- BASED

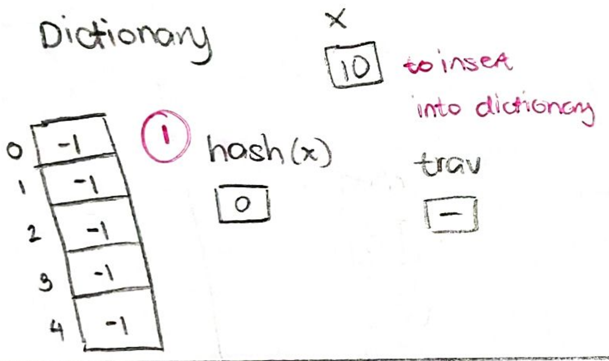
//initVH() & initDict()



```
#define MAX 10
#define HEAP 20

typedef struct node{
    char data;
    int link;
}Nodetype

typedef struct heap{
    Nodetype Node[HEAP];
    int avail;
} VirtualHeap;

typedef int Dictionary
                [MAX];
```

//insert()





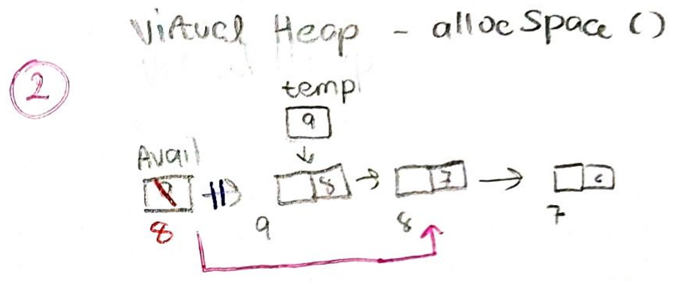
1.) Call hash fn() to get hash value of x.

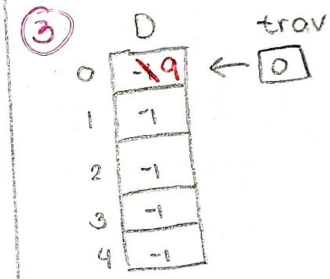
2.) Allocate space in VH/ retrieve an available node in which we insert new data to. We will then link this node to Dictionary.

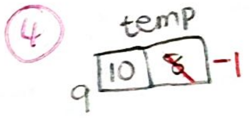
3.) Using *trav, find the appropriate position (hash index & sorted position) in Dictionary.

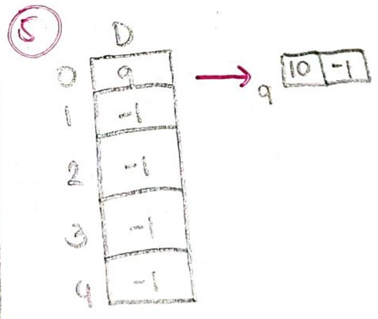
4.) Input elem x in temp node & update link to hold value in *trav.

5.) New node is now linked & inserted into Dictionary.



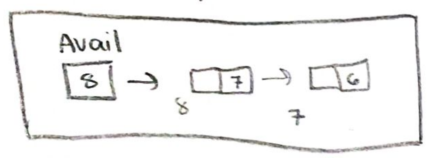




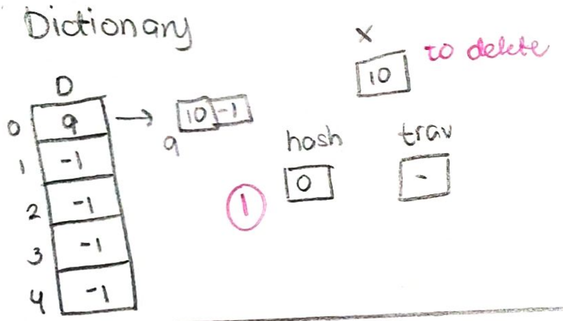


//delete ()

Virtual Heap



Dictionary

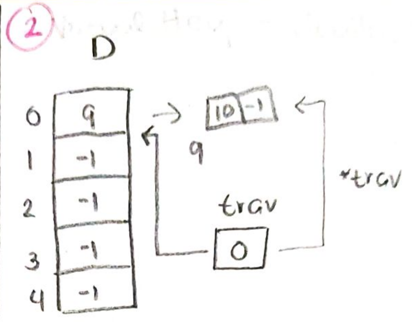


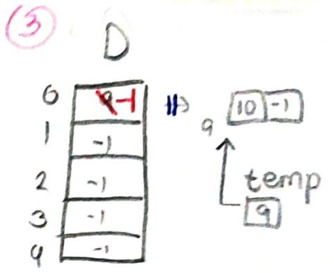
1.) call hashfn() to get hash value of x.

2&3.) Using *trav, find the element x to delete. Once found, let temp point to node.

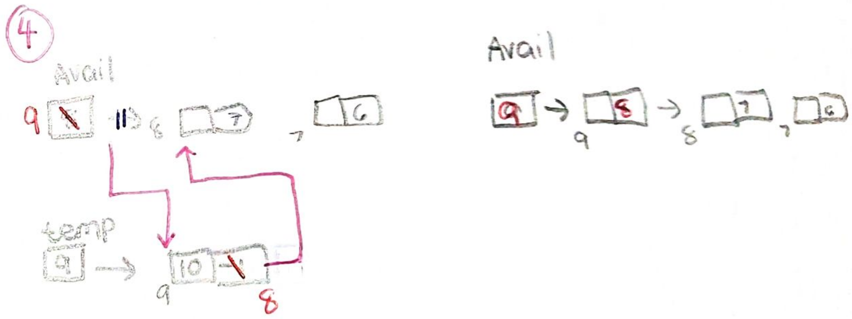
4.) Deallocate temp node/ return to list of available nodes. Update link fields.

5.) Node is now deleted from Dictionary





Virtual Heap - deallocSpace ()


④
Avail
temp
Avail