제11회 실습문제

[Java 사용자]

Java의 HashSet과 유사한 기능을 제공하는 MyHashSet<E> 클래스를 구현한다. 다음과 같은 생성자와 public메서드를 제공한다.

`MyHashSet(int initialCapacity, float loadFactor)`	해쉬 테이블의 초기 크기와 `load factor`를 지정하는 생성자이다. 저장된 원소의 개수가 `capacity*load factor`를 초과하면 배열의 크기를 2배로 늘린다.
`boolean add(E entry)`	`E` 타입의 원소 `entry`를 추가하고 `true`를 반환한다. 만약 동일한 원소가 이미 존재하면 `false`를 반환한다.
`void clear()`	저장된 모든 원소들을 삭제한다.
`boolean contains(E entry)`	`E` 타입의 원소 `entry`가 존재하는지 검사한다.
`boolean isEmpty()`	공집합인지 검사한다.
`boolean remove(E entry)`	원소 `entry`를 삭제하고 `true`를 반환한다. 그런 원소가 존재하지 않으면 `false`를 반환한다.
`int size()`	집합의 크기를 반환한다.

예시:

```
public class MyHashSet<E>
{
  private class Node {    // for chaining
      public E element;
      public Node next;
      …
  }
  private Node [] table;
  private int capacity;              // size of the table
  private float loadFactor;                  // load factor
  private int size;                          // the number of elements in the set

  public MyHashSet(int initCap, float load) {
    …
  }

  private int hash(E entry) {
    return (entry.hashCode() & 0x7fffffff) % capacity;
  }

  // public methods here

}
```

[C 사용자]

다음과 같이 문자열을 저장하는 해쉬 테이블을 구현한다.

`Table *create_hashset(int initialCapacity, float loadFactor)`	해쉬 테이블의 초기 크기와 `load factor`를 지정하는 생성자이다. 저장된 원소의 개수가 `capacity*load factor`를 초과하면 배열의 크기를 `2`배로 늘린다.
`bool add(Table * table, char *entry)`	문자열 `entry`를 추가하고 `true`를 반환한다. 만약 동일한 원소가 이미 존재하면 `false`를 반환한다.
`void clear(Table * table)`	저장된 모든 원소들을 삭제한다.
`bool contains(Table * table, char *entry)`	문자열 `entry`가 존재하는지 검사한다.
`bool isEmpty(Table * table)`	공집합인지 검사한다.
`bool remove(Table * table, char * entry)`	원소 `entry`를 삭제하고 `true`를 반환한다. 그런 원소가 존재하지 않으면 `false`를 반환한다.
`int size(Table * table)`	집합의 크기를 반환한다.

예시:

```
struct node  {    // for chaining
     char *element;
     struct node *next;
     …
};
typedef struct node Node;

typedef struct table {      //  hash table
    Node *table;
    int capacity;                  // size of the table
    float loadFactor;              // load factor
    int size;                      // the number of elements in the set
} Table;
Table *create_hashset(int initCap, float load) {
   …
}
int hash(char * entry) {
    // use hash function for string at 22 page of slide
}
…
```