1.Two Sum

#include <stdio.h>

#include <stdlib.h>

#define TABLE\_SIZE 10000

typedef struct {

    int key;

    int value;

} Pair;

int hash(int key) {

    return abs(key) % TABLE\_SIZE;

}

void insert(Pair\* table[], int key, int value) {

    int index = hash(key);

    while (table[index] != NULL) {

        if (table[index]->key == key) return;

        index = (index + 1) % TABLE\_SIZE;

    }

    table[index] = (Pair\*)malloc(sizeof(Pair));

    table[index]->key = key;

    table[index]->value = value;

}

int search(Pair\* table[], int key) {

    int index = hash(key);

    while (table[index] != NULL) {

        if (table[index]->key == key) {

            return table[index]->value;

        }

        index = (index + 1) % TABLE\_SIZE;

    }

    return -1;

}

int\* twoSum(int\* nums, int numsSize, int target, int\* returnSize) {

    Pair\* hashTable[TABLE\_SIZE] = {NULL};

    int\* result = (int\*)malloc(2 \* sizeof(int));

    for (int i = 0; i < numsSize; i++) {

        int complement = target - nums[i];

        int foundIndex = search(hashTable, complement);

        if (foundIndex != -1) {

            result[0] = foundIndex;

            result[1] = i;

            \*returnSize = 2;

            for (int j = 0; j < TABLE\_SIZE; j++) {

                if (hashTable[j]) free(hashTable[j]);

            }

            return result;

        }

        insert(hashTable, nums[i], i);

    }

    \*returnSize = 0;

    free(result);

    return NULL;

}