German International University
Informatics and Computer Science
Prof. Dr. Slim Abdennadher

Data Structures and Algorithms, Winter semester 2021
Practice Assignment 11

Exercise 11-1

Assume you have a hash table of size 5 and a hash function of:

```
H(N) = N % TABLESIZE
```

Suppose we use a hashing strategy, which is a **combination of linear probing and the chaining method**, i.e., we first try to insert a number using linear probing, and if all cells are filled, we insert it with the original match `(N % TABLESIZE)` using the chaining method. We will call each insertion using chained hash tables a **collision**.

For example, if all the cells are filled and you already have 11 in cell 1 and you need to insert 21, you will insert it with 11 using chaining method and we will have one collision.

You are given the following inputs (N):

```
22, 91, 7, 881, 99, 15, 12, 31, 10
```

How many collisions will you have for cell 0 `(N % TABLESIZE = 0)`?

- ☐ 1
- ☐ **2**
- ☐ 3
- ☐ None of the above

Show your workout.

Exercise 11-2

Given an unsorted array that can contain duplicates. How can you remove duplicates guaranteeing expected linear running time? First explain the algorithm in English and justify the time complexity. Then implement it. You can make any assumption needed to guarantee the desired time complexity.

Exercise 11-3 To be discussed in the Tutorial

Draw the hash table of size `b = 10` that results from using the hash function:

$$h(x) = x\%b$$

to hash the integer keys:

```
4371, 1323, 6137, 4199, 4344, 9679, 1989
```

Show each step needed to build a hash table in each of the following cases:

a) A hash table using linear probing

b) A hash table using quadratic probing

c) A hash table using double hashing with the following sequence of hash functions:

$$h_i(x) = (x\%b + i \times (7 - (x\%7)))\%b, \quad \text{for } i = 0, 1, 2, \ldots$$

Exercise 11-4

Given the following definition for a hash table class:

```
class MyHashTable{
    private static int N = 11;
    private int[] table = new int[N];

    public int h(int key) {
        return (3 * key + 17)%N;
    }

    public MyHashTable(){
        //a new Hashtable is initially empty
        //fill all cells with value -1
        Arrays.fill(table, -1);
    }

    public void insert(int key) {
        //write your solution
    }

    public boolean find(int key) {
        //write your solution
    }
}

public class MyHashTableTest {
    public static void main(String[] args) {
        MyHashTable h = new MyHashTable();
        //write your test cases
    }
}
```

Write code to implement the find() and insert() methods. To simplify the problem, do not check for a full array. Also, you may assume that the hashing function will eventually find an empty cell if one exists. For simplicity, assume that an empty cell is having the value -1. To avoid collision, linear probing is used.

Exercise 11-5

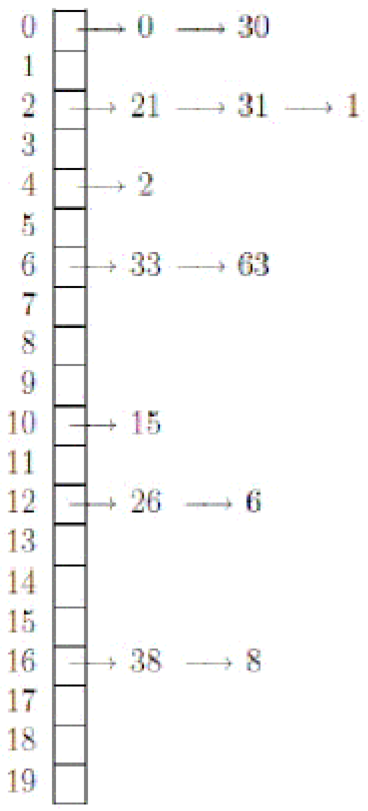
Consider two sets of integers stored in two unsorted arrays A and B. Neither A nor B can contain duplicates, but they can share some values. We want to compute the symmetric difference of these two sets: $A - B$

$$A - B = \{x | (x \in A \ \texttt{and} \ x \notin B) \ \texttt{or} \ (x \in B \ \texttt{and} \ x \notin A)\}$$

How can $A - B$ be computed guaranteeing expected linear running time? First explain the algorithm in English and justify the time complexity. Then implement it using the built-in Java type HashSet.
https://docs.oracle.com/javase/7/docs/api/java/util/HashSet.html
Assume that the hash function is a perfect one. Thus, searching and insertion can be done in O(1).

Exercise 11-6 Final Exam 2013

Suppose you have a hash table and have inserted some elements. When you inspect it you see that the result looks as the picture below. Being a good student, you realize this is a problem



a) State the problem that occurs here.

b) Give an example of a hash function that could give rise to this behavior for the numbers above.

c) What would be a perfect hash function for the table above?

Exercise 11-7

a) What is one of the negative side effects of linear probing:

- ☐ Too many values hash to the same key and hence create long chains
- ☐ **Too many values, all hash to the same area of the hash table and cause clustering**
- ☐ Too many linked lists in the table, increasing search time complexity
- ☐ None of the above

b) Searching with hash tables is highly dependent on how full the table is.

- ☐ **Yes**
- ☐ No

c) What is the effect of adding objects to a hash table if they are sorted versus in random order?

- ☐ There is an effect
- ☐ **No effect**

d) What can you say about a hash table that has n cells and $10 \times n$ objects?

Exercise 11-8

Draw the hash table of size 11 that results from using the hash function

$$h(i) = (2i + 5)\%11$$

to hash the integer keys:

```
12 44 13 88 23 94 11 39 20 16 5
```

for each of the following collision resolution strategies:

a) chaining method

b) linear probing

c) quadratic probing (up to the point where the method fails)

d) double hashing using the secondary hash function:

$$h_2(key) = 7 - (key\%7)$$

then if the secondary function fails, linear probing is used.

Exercise 11-9

Write a `Java` program that **implements a hashed symbol table**. Your resulting hash table must be at most 70% full. Create a hash function that minimizes the size of the hash table and the number of collisions, using chaining method. Your hash table should be implemented as an array of references, each referring to a linked list. Each node of the linked list will contain a string (i.e. a word from the input file) and a reference to the next node. Your program should do the following:

a) Read in a set of identifiers from a file. A file containing over 10000 English words will be posted on the course web site. Use this file to test your program. The second smaller data set is also provided. You can use it for debugging.

b) Enter each of the identifiers into the hash table.

c) Re-read the set of identifiers from the file, and look up each one in the hash table.

d) Determine and print the average number of comparisons required to look up a word in the input file. Determine and print the load factor of your resulting hash table. Determine and print the size of the longest chain.

Answer the following questions:

- Provide a brief description of your hash function.
- Analyze the quality of your hash function based on the statistics reported. In which way could you improve it?
- If the input data set would contain numerals (for instance, student IDs) instead of alphabetical characters, how would it influence the performance of your hash function. Would it improve, deteriorate or remain the same? Justify.