
Problem Set 3

This problem set is due **Wednesday, March 21** at **11:59PM**.

Solutions should be turned in through the course website. **You must enter your solutions by modifying the solution template (in Python) which is also available on the course website.** The grading for this problem set will be largely automated, so it is important that you follow the specific directions for answering each question.

For multiple-choice and true/false questions, no explanations are necessary: your grade will be based only on the correctness of your answer. For all other non-programming questions, full credit will be given only to correct solutions which are described clearly and concisely.

Programming questions will be graded on a collection of test cases. Your grade will be based on the number of test cases for which your algorithm outputs a correct answer within time and space bounds which we will impose for the case. Please do not attempt to trick the grading software or otherwise circumvent the assigned task.

1. Hash Collisions (20 points)

Consider hashing integers which are selected independently at random from the universe $U = [1, 2, \dots, 84]$.¹ Recall that a hash family h_i from U to $\{0, 1, \dots, m - 1\}$ is universal if, for any distinct x and y :

$$\Pr_i[h_i(x) = h_i(y)] \leq \frac{1}{m}.$$

- a. Suppose we would like $m = 4$. Consider the hash family $h_i(x) = x^2 + x + i \pmod{4}$, for $i \in \{0, 1, 2, 3\}$.
 1. What is the probability of having NO collisions when TWO random elements are hashed using the function $h_2(x) = x^2 + x + 2 \pmod{4}$?
 2. The family $h_i(x)$ a universal hash family. **True or False?**
- b. Suppose we would like $m = 3$. Consider the hash family $h_i(x) = x^2 + i \pmod{3}$, for $i \in \{0, 1, 2\}$.
 1. What is the probability of having NO collisions when TWO random elements are hashed using the function $h_1(x) = x^2 + 1 \pmod{3}$?
 2. The family $h_i(x)$ a universal hash family. **True or False?**

¹The integer keys are chosen independently. It is possible that we attempt to hash two things with the same key, in which case we will consider this a hash collision.

- c. Suppose we would like $m = 12$. Consider the hash family $h_i(x) = ix + 2 \pmod{12}$, for $i \in \{0, \dots, 11\}$.
1. What is the probability of having NO collisions when TWO random elements are hashed using the function $h_7(x) = 7x + 2 \pmod{12}$?
 2. The family $h_i(x)$ a universal hash family. **True** or **False**?
- d. Suppose we would like $m = 7$. Consider the hash family $h_i(x) = ix + 2 \pmod{7}$, for $i \in \{0, \dots, 6\}$.
1. What is the probability of having NO collisions when TWO random elements are hashed using the function $h_5(x) = 5x + 2 \pmod{7}$?
 2. The family $h_i(x)$ a universal hash family. **True** or **False**?

Solution Format:

Your answer for the first part of each question should consist of a float probability in the range $[0, 1]$, accurate to within 0.001 of the correct answer. Your answer to the second part of each question should be a boolean.

2. Open Addressing (30 points)

Suppose you are hashing integers into the hash table of size 10 below, using the hash function $h(k) = k \bmod 10$ to find the location of key k and using linear probing to resolve collisions.

After inserting 6 values into the empty hash table, the table is in the state below:

0	
1	
2	22
3	13
4	54
5	32
6	46
7	43
8	
9	

- (a) Which one of the following insertion orders would result in this state?
- 1) 46, 22, 54, 32, 13, 43
 - 2) 54, 22, 13, 32, 43, 46
 - 3) 46, 54, 22, 13, 32, 43
 - 4) 22, 46, 43, 13, 54, 32
- (b) Suppose that 46 was deleted from the table. How many cells would be inspected if you then searched the table for 65?
- (c) Is there some sequence of insertions and deletions, starting from an empty table, after which each cell i contains the value $i+1$? Give an example of such a sequence, or prove that no such sequence exists.
- (d) Is there some sequence of insertions and deletions, starting from an empty table, after which each cell i contains the value $9 - i$? Give an example of such a sequence, or prove that no such sequence exists.

Solution Format:

For part a), your answer should be an integer choice, and for part b) it should be a integer answer. For parts c) and d), if you believe that no such sequence exists, your answer should be a string containing a proof of this fact. If you have a counterexample, you should enter it as a list of tuples; the first element of each tuples should either be an 'i' for insertion or a 'd' for deletion, and the second should be the key being inserted or deleted.

3. Price changes (20 points)

The local supermarket sells n products whose prices are stored in a sorted array $[p_1, p_2, \dots, p_n]$, where $p_i \leq p_{i+1}$ for all $i \leq n - 1$.

After some seasonal price cuts, k of these prices are updated. You are informed of an array of price changes $[(i_1, d_1), \dots, (i_k, d_k)]$. Here, a tuple (i, d) means that the i th price should be changed by d (which may be negative), so p_i is changed to $p_i + d$.

Give a fast algorithm which takes the original price array and the array of price changes, and computes the resorted array of new prices after the changes. Analyze its running time in terms of n and k .

You can assume that comparison of prices can be done in constant time (and you may not assume anything else about the prices).

Solution Format:

Your answer to this problem should be a string containing a concise description of your algorithm and a brief analysis of its runtime.

4. One-Bit Error Correction (60 points)

Suppose that you want to recover messages sent over a noisy channel. You are given a list of k valid messages m_1, m_2, \dots, m_k , each of which is an n -bit binary string. The messages r received from the channel are all corruptions of one of these k strings - each one differs from exactly one of the m_i in exactly one position. Your goal is to find the index i of the valid message that r is derived from.

Write a function `recover_original_messages` that takes two parameters, the lists `valid_messages` and `corrupted_messages`, and returns a list containing the indices of the valid messages corresponding to each corrupted message. Each of the valid and corrupted messages will be a string containing only the characters '0' and '1'. All of these strings will be the same length.

Note that the number of valid messages, the number of corrupted messages, and the length of each message will be quite large. Your algorithm should scale well with all of these parameters.

Here are some tests which your function should pass:

```
recover_original_messages(['000', '111'], ['110', '010']) == [1, 0]
recover_original_messages(['000000', '110001', '001110', '111111'],
                          ['001010', '110000', '110111']) == [2, 1, 3]
```

Solution Format:

You should answer this problem by filling in the body of the `recover_original_messages` function in the solution template.