0 Student Info

Your name and student id:

Solution:

1 Selection Algorithm (10 points)

(a) As we have learned in the lecture, partition takes $O(n)$ time complexity. In terms of random selection algorithm, when will we encounter the worst-case scenario? What is the worst-case runtime for random selection algorithm? Is this related to input sequence? Why or Why not? (4 points)

(b) Do we have a sequence of input which guarantees that it can have the best-case runtime for random selection algorithm? What is the best-case runtime for random selection algorithm? (2 points)

(c) Deterministic selection algorithm guarantees its runtime to be $O(n)$ for every input array of length n. Given the assumption that there exists a positive constant k that satisfies:

i) $T(1) \leq k$

ii) $T(n) \leq kn + T\left(\frac{9n}{13}\right) + T\left(\frac{3n}{26}\right)$

Then prove that $T(n) = O(n)$. (Please clearly show your steps in detail) (5 points)

Solution:

2 Hash Fruits (30 points)

Suppose Joseph is using a hash table to store information about the color of different kinds of fruit. The keys are strings, and the values are also strings. Furthermore, he uses a very simple hash function where the hash value of a string is the remainder of the integer representing its first letter (all the letters are in lowercase) divided by 13. For example:

- h("apple") = 0
- h("banana") = 1
- h("nectarine") = 0
- h("zebrafruit") = 12

And we have:

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
0	1	2	3	4	5	6	7	8	9	10	11	12	0	1	2	3	4	5	6	7	8	9	10	11	12

Now, assume we are working with a hash table of size 9 and the compression function $c(x) = x \ \% \ 9$. This means that "zebrafruit" would hash to 12, but ultimately fall into bucket $12 \ \% \ 9 = 3$ of our table. For this problem, you will determine where each of the given fruits lands after inserting a sequence of values using three different collision resolution schemes:

- linear probing
- quadratic probing
- double hashing with $h'(k) = q - (h(k)^2 \ \% \ q)$, where $q = 7$

For each of these three collision resolution schemes, determine the resulting hash table after inserting the following (*key*, *value*) pairs in the given order:

1. ("banana", "yellow")
2. ("blueberry", "blue")
3. ("blackberry", "black")
4. ("cranberry", "red")
5. ("apricot", "orange")
6. ("watermelon", "green")

Every incorrect value counts for 1 point.

2.1 Linear Probing (10 points)

Please use the **linear probing** collision resolution method to simulate the given insertion steps, and then show the final position of each (*key*, *value*) pair inside the related buckets below.

Solution:

Index	0	1	2
Key			
Value			
Index	3	4	5
Key			
Value			
Index	6	7	8
Key			
Value			

2.2 Quadratic Probing (10 points)

Please use the **quadratic probing** collision resolution method to simulate the given insertion steps, and then show the final position of each (*key*, *value*) pair inside the related buckets below.

Solution:

Index	0	1	2
Key			
Value			
Index	3	4	5
Key			
Value			
Index	6	7	8
Key			
Value			

2.3 Double Probing (10 points)

Please use the **double probing** collision resolution method to simulate the given insertion steps, with the double hash function $h'(k) = q - (h(k)^2 \ \% \ q)$ and $q = 7$, and then show the final position of each (*key*, *value*) pair inside the related buckets below.

Solution:

Index	0	1	2
Key			
Value			
Index	3	4	5
Key			
Value			
Index	6	7	8
Key			
Value			

3 Hash! Hash! Hash! (30 points)

3.1 Possible Insertion Order (10 points)

Suppose you have a hash table of size 10 uses open addressing with a hash function $H(k) = k \bmod 10$ and linear probing. After entering six values into the empty hash table, the state of the table is shown below.

Index	0	1	2	3	4	5	6	7	8	9
Key			42	43	14	62	76	83		

Write out five different possible insertion orders and clearly state the insertion procedures for each.

Solution:

3.2 Time Complexity of Hashing (10 points)

i) In a few sentences, explain why a hash table with open addressing and rehashing (doubling the size every time when the load factor exceeds a threshold) achieves insertion in $O(1)$ time. (5 points)

Solution:

ii) When rehashing is not triggered, is the insertion time $O(1)$ (does it have an upper bound)? Why? (5 points)

Solution:

3.3 Wrong Delete (10 points)

William implements a hash table that uses open addressing with linear probing to resolve collisions. However, his implementation has a mistake: when he erases an element, he replaces it with an empty bucket rather than marking it as deleted! In this example, the keys are strings, with the hash function:

```
size_t hash(string s) {
  return s.empty() ? 0 : s[0] - 'A';
}
```

and the hash table initially contains 100 buckets. After which of the following sequences of operations will the hash table be in an invalid state due to erased items being marked empty rather than

as deleted? In this case, a hash table is invalid if subsequence "find" or "size" operations do not return the correct answer. You may select **one or more** options.

A. insert "A1"; insert "B1"; insert "C1"; erase "A1"; erase "C1";

B. insert "A1"; insert "A2"; insert "A3"; erase "A3"; erase "A2";

C. insert "A1"; insert "B1"; insert "A2"; erase "B1"; insert "B2";

D. insert "B1"; insert "C1"; insert "A1"; insert "A2"; erase "C1";

E. none of the above

Please clearly explain your answer.

Solution:

4 Bloom Filter (30 points)

Suppose there is an array A of $n = 17$ bits. Each bit is either 0 or 1. And we have 3 hash functions h_1, h_2, h_3, each mapping inside $\{0, 1, \ldots, n-1\}$:

$$h_1(x) = x \ \% \ n$$
$$h_2(x) = (5 * x + 2) \ \% \ n$$
$$h_3(x) = (7 * x + 3) \ \% \ n$$

Initially the array is all-zero.

i) Joseph first inserts one element 26 into the bloom filter. Please write down the current values of entries of array A and clearly explain your answer. (5 points)

Solution:

ii) Then Joseph inserts 3 into the bloom filter. Please write down the current values of entries of array A. (5 points)

Solution:

iii) After inserting 3, Joseph finds that it is actually a mistake, and would like to remove 3 from the bloom filter. Can he remove the element 3 from the fliter? If so, how it can be achieved? (5 points)

Solution:

iv) Now Joseph wants to find out whether some of the elements are in this filter. Does element 7 in this filter? What about 17? Please clearly show your steps. (15 points)

Solution: