HackerRank For Work

Info6205_Summer18_Quiz10 30 minutes

Question - 1

SCORE: **5 points**

Question 1

Consider a hash table with 100 slots. Collisions are resolved using chaining. Assuming simple uniform hashing, what is the probability that the first 6 slots are unfilled after the first 3 insertions?

- [x] (94 × 94 × 94) / (100^3)
- [ ] (96 × 95 × 94) / (100^3)
- [ ] (94 × 93 × 92) / (3! * 100^3)
- [ ] (94 × 93 × 92) / (100^3)

Question - 2

SCORE: **5 points**

Question 2

Suppose we are using Hash(k) = 3 * k % 13, and an array of size 13 as a Hash Table (indexes start at zero), what's the result after we put the numbers into the hash table if we use linear probing? (where "*" represents that there is no value in the hash table)

The numbers are inserted according to the following order: 22 -> 40 -> 36 -> 55 -> 24 -> 27 -> 28

- [ ] * 22 * 40 36 27 * 24 * 55 28 * *
- [ ] 22 * 40 36 27 28 24 * 55 * * * *
- [ ] 22 * 27 36 28 * 24 * 55 * * * *
- [x] * 22 * 40 36 27 28 24 * 55 * * *
- [ ] * 22 * 40 27 36 * 24 * 55 * * *
- [ ] * 22 * 27 36 28 * 24 * * * * *
- [ ] * 22 * 40 36 * * 24 * 55 * * *

Question - 3

SCORE: **5 points**

Question 3

Suppose we have a class *X* which contains 2 attributes: Name and ID. We manually override the *hashCode* function with our own implementation that returns a value based on both attributes, eg. *name.hashCode() + ID*31*.

First, we create a new instance *X x = new X("INFO6205", 27)*. We then add this instance (*x*) into an empty HashSet (*s*), i.e. *s.put(x)*. Next, we modify the ID of this instance, e.g. *x.setID(42)*.

Finally, we invoke the *contains* function, i.e. *s.contains(x)*. What result should we expect?

- ( ) Null
- ( ) True
- (•) False
- ( ) Runtime Exception

Question - 4

SCORE: **5 points**

Question 4

An advantage of separate chaining as an implementation of a hash table over the linear probing (open addressing) scheme is:

- ( ) Space used is less
- (•) Deletion is easier
- ( ) Worst case complexity of search operations is less
- ( ) None of the above

Question - 5

SCORE: **30 points**

Linear Probing Hash Table

Please implement the *put* and *get* methods for a linear probing hash table.