Computer Science I — CSCI-141
Hashing — Homework 12

10/28/2020

Problem

You should do this assignment *without* the aid of software development tools like PyCharm or a Python interpreter. Write down your answers to these questions by hand or by using any word processing program.

There will be questions on the final exam very similar to these.

In these exercises you should treat a hash table as a *set*, which means that it only has keys; there are no values. Here is a translation of operation types from dictionary to set. Note that these are generic operation names; they are not related to methods provided by Python's `dict` and `set` classes.

Hashing Dictionary `d`	Hashing Set `s`
`contains(d,key)`	`contains(s,key)`
`put(d,key,value)`	`add(s,key)`
`get(d,key)`	*no equivalent*

Questions are formatted on separate pages. That way you have the option of writing your answers on a hard copy of this document and then scanning the pages to submit them.

1 Hash Table of Strings

Here are the assumptions and rules for hashing a set of strings:

- Assume that strings contain only lower case English letters.
- Letters are coded according to the English alphabet, starting with 'a' encoded as 1 and ending with 'z' encoded as 26.
- The hash function is computed by taking the codes for letters in positions 0, 2, 4, etc. positions and adding them together.
- Example: `hash("howdydoody")` $= 8 + 23 + 25 + 15 + 4 = 75$

You have a hash table of size 10 that uses open addressing to handle collisions.

It is initially empty. Then the following strings get added in the order shown.

“afghanistan”, “latvia”, “myanmar”, “canada”, “papuanewguinea”, “ecuador”, “mongolia”, “newzealand”

1.1 Drawing of Table

Draw the contents hash table after the above strings were added. Make sure you show the index numbers of the entries in the table.

Hint: “afghanistan” will end up going into the location at index 2 in the table.

For full credit, show your work! That means showing character codes and hash values, like in the `"howdydoody"` example above.

1.2 Collision Count

How many collisions occurred while placing the strings in the table? For each string placed in the table, add the count of collisions that resulted in putting the string where it ended up. If a string was placed where its hash calculation initially indicated, i.e., if there was no *probing*, that is a collision count of 0.

2 Hash Table of Objects

A new hash table contains objects. The objects' data class is defined as follows.

```
@dataclass(frozen=True)
class CountryInfo:
    name: str
    pop: int
    region: int
```

The hash function for `CountryInfo` objects is computed by adding the hash function of `name` as defined in **Problem 1** to the product of the values of `pop` and `region`.

Show a size 10 table of `CountryInfo` objects to which the following have been added in the order shown. As before, open addressing is used to handle collisions. Similarly to Problem 1, show your calculations of hash values from `CountryInfo` objects.

```
CountryInfo( ‘‘afghanistan’’, 37, 3 )
CountryInfo( ‘‘latvia’’, 2, 5 )
CountryInfo( ‘‘myanmar’’, 53, 3 )
CountryInfo( ‘‘canada’’, 37, 6 )
CountryInfo( ‘‘ecuador’’, 17, 7 )
CountryInfo( ‘‘mongolia’’, 3, 3 )
```

3 Bad Hash Function

With a hash table of size 8 and a hash function that always returns 3, what would the table look like after the following values are added, in the given order?

"afghanistan", "latvia", "myanmar", "canada", "papuanewguinea", "ecuador", "mongolia", "newzealand"

As before, open addressing is used to handle collisions.

Logistics

3.1 Grading

- 40%: Question 1
- 36%: Question 2
- 24%: Question 3

3.2 Submission

Create the file **hw12.pdf** so that it contains your written answers converted to PDF.

Any other format will incur a loss of 20% on your grade. (The format originally used that got converted to PDF is your decision.)

Submit the file **hw12.pdf** to the myCourses dropbox.