LT10d : About Hashing

(Part 2 of 2)

Python codes

The Hash Function

```
def hash(string):
    total = 0
    for i in range(len(string)):
        total += ord(char)*(i+1)
    return total%5
```

Create an empty Hash Table:

```
def init_table(n):
    table = []      # declare
    table += [''] * n  # initialize
    return table
```

Populate the Hash Table using the Hash Values:

```
def hashtable(seq):
    tbl = init_table(len(seq))
    for ele in seq:
        i = hash(ele)
        if tbl[i] == '':
            tbl[i] = ele
        else:
            #collision resolution
            print(ele, ' is not added.')
    return tbl
```

Using Hash Function and Table:

```
>>> lst = ['Chloe Niu Man Yun', 'Ngyuen Hoang 
Minh', 'Poh Zheng Hong', 'Suresh Kannan 
Sakthieshwar', 'Wong Yong Xiang']

>>> data_table = hashtable(lst)
```

Searching for a name in Hash Table:

```
def search(table, name):
    i = hash(name)
    if table[i] == name:
        return True
    else:
        return False

>>> search(data_table, 'Wong Yong Xiang')
```

Handling Collisions:

Some basic questions :

7. How to store 5 names in 5 boxes? (with collision)

- Chloe Niu Man Yun	Sum = 13465, Mod5 = 0
- Ngyuen Hoang Minh	Sum = 14092, Mod5 = 2
- Poh Zheng Hong	Sum = 9646, Mod5 = 1
- Suresh Kannan Sakthieshwar	Sum = 35379, Mod5 = 4
- Muhammad Asyraf Bin Omar	Sum = 26992, Mod5 = 2

Collision Resolution (1): "Separate Chain"

Hash Table (with “Separate Chain”):

```
def hashtable(seq):
    tbl = init_table(len(seq))
    for ele in seq:
        i = hash(ele)
        if tbl[i] == '':
            tbl[i] = ele
        else: #collision resolution
            if type(tbl[i]) != list:
                tbl[i] = [tbl[i], ele]
            else:
                tbl[i] = tbl[i] + [ele]
    return tbl
```

‘Separate Chain’:

8. How to search when there is collision?

Box Index	Table
0	’Chloe Niu Man Yun’
1	’Poh Zheng Hong’
2	[’Ngyuen Hoang Minh’, **’Muhammad Asyraf Bin Omar’**]
3	
4	’Suresh Kannan Sakthieshwar’

This is a “Separate Chain”, we need to perform a linear search in the separate chain !

Searching for a name in Hash Table (with “Separate Chain”):

```
def search(table, name):
    i = hash(name)
    if table[i] = ‘’
        return False
    elif table[i] == name:
        return True
    elif type(table[i]) == list:
        return name in table[i]   # Boolean
```

Collision Resolution (2): Open Hashing

Using Hash Function and Table:

```
def hashtable(seq):
    tbl = init_table(len(seq))
    for ele in seq:
        i = hash(ele)
        if tbl[i] == '':
            tbl[i] = ele
        else:
            #collision resolution
            return openhash(tbl, i, ele)
return tbl
```

Collision Resolution: Open Hashing

```
def openhash(table, index, item):
    while True:
        if table[index] == '':
            table[index] = item
            return table
        else:
            index += 1
            if index == len(table):
                index = 0
```

The search will look for next box until it reaches the end and goes to the first box .

The search will look for next box until it reaches the end and goes to the first box .

```
index += 1
if index == len(table):
    index = 0
```

For a table of size 10 and if we start looking at index = 9, then index will increase to 10, 11, 12, 13 ... and so on.

Consider index = index % len(table),

we will look at 0 instead of index = 10,

we will look at 1 instead of index = 11,

we will look at 2 instead of index = 12,

... so we are actually looking from the beginning of the table.

Collision Resolution: Open Hashing

```
def openhash(table, index, item):
    while True:
        if table[index] == '':
            table[index] = item
            return table
        else:
            index = (index + 1)%len(table)
```

This is a simpler way of search the next box and go to the first when it hits the end.

Assume ‘Open Hashing’ for collision:

8. How to search when there is collision?

Box Index	Table
0	’Chloe Niu Man Yun’
1	’Poh Zheng Hong’
2	’Ngyuen Hoang Minh’
3	**’Muhammad Asyraf Bin Omar’**
4	’Suresh Kannan Sakthieshwar’

There is one collision !

Searching for a name in Hash Table (with Open Hashing):

```
def search(table, name):
    i = hash(name)
    if table[i] == ‘’
        return False
    elif table[i] == name:
        return True
    else:
        return linear_search[table, name, i]
```

Open Hash Search:

```
def linear_search(tbl, item, i):
    for cell in range(len(tbl)):
        if tbl[i] == '':
            return False
        elif tbl[i] == item:
            return True
        i = (i + 1) % len(tbl)
    return False
```

The End