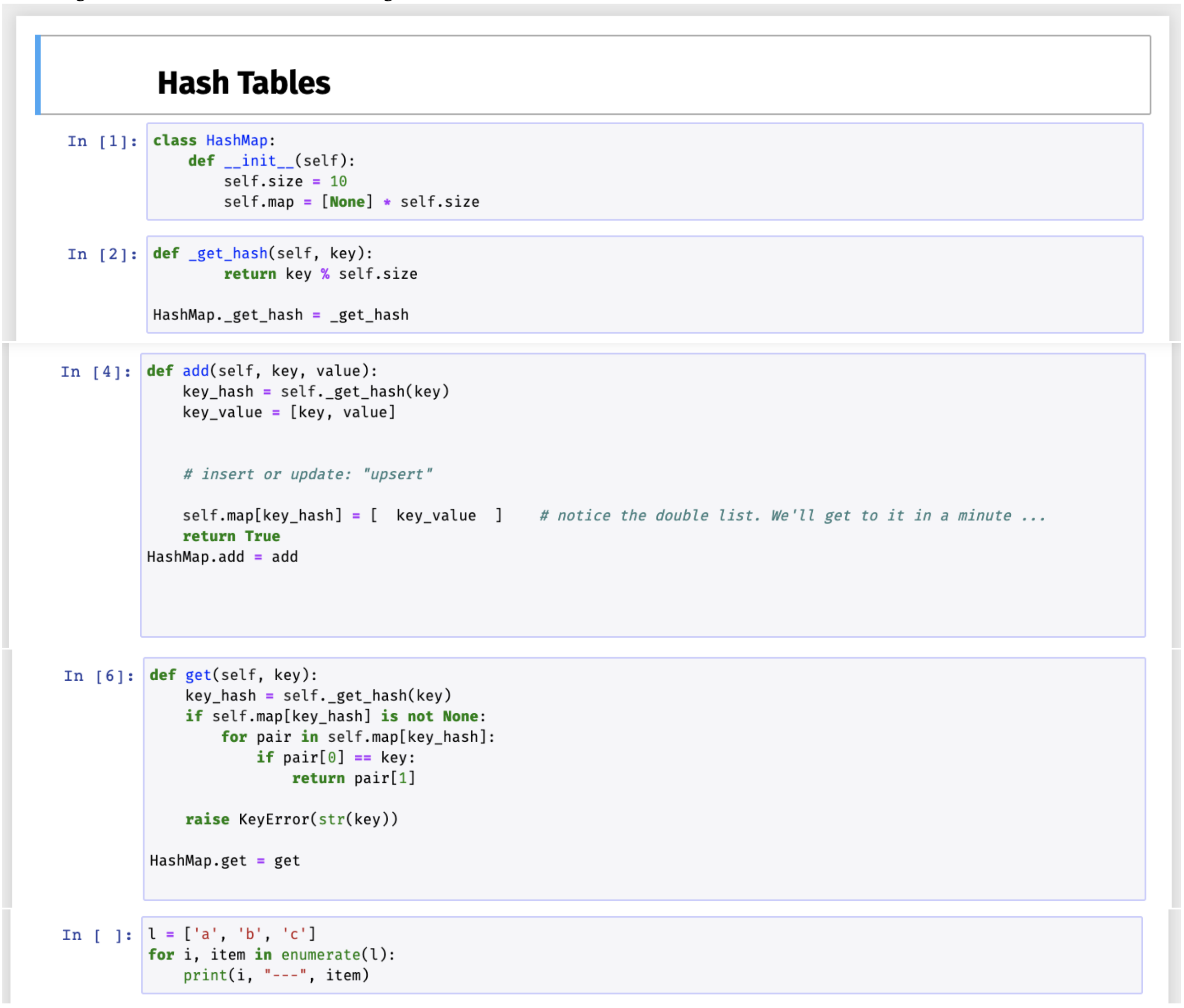
CS218 - Data Structures
FAST NUCES Peshawar Campus
Dr. Nauman (recluze.net)

October 22, 2019

1 Maps and HashMaps

Raster images of the notebook 16-hashing.



Hash Tables

In [1]:
```
class HashMap:
    def __init__(self):
        self.size = 10
        self.map = [None] * self.size
```

In [2]:
```
def _get_hash(self, key):
        return key % self.size

HashMap._get_hash = _get_hash
```

In [4]:
```
def add(self, key, value):
    key_hash = self._get_hash(key)
    key_value = [key, value]


    # insert or update: "upsert"

    self.map[key_hash] = [  key_value  ]    # notice the double list. We'll get to it in a minute ...
    return True
HashMap.add = add
```

In [6]:
```
def get(self, key):
    key_hash = self._get_hash(key)
    if self.map[key_hash] is not None:
        for pair in self.map[key_hash]:
            if pair[0] == key:
                return pair[1]

    raise KeyError(str(key))

HashMap.get = get
```

In [ ]:
```
l = ['a', 'b', 'c']
for i, item in enumerate(l):
    print(i, "---", item)
```

In [ ]:
```python
def __str__(self):
    ret = ""

    for i, item in enumerate(self.map):
        if item is not None:
            ret += str(i) + ": " + str(item) + "\n"
    return ret

HashMap.__str__ = __str__
```

In [ ]:
```python
h = HashMap()
```

In [ ]:
```python
h.add(17, "seventeen")
h.add(26, "twenty six")
h.add(35, "thirty five")
h.add(26, "twenty six updated")
```

In [ ]:
```python
print(h)
```

In [ ]:
```python
print(h.get(26))
```

Collisions and Avoidance

In [ ]:
```python
h = HashMap()

h.add(17, "seventeen")
h.add(26, "twenty six")
h.add(35, "thirty five")
h.add(25, "twenty five")
h.add(26, "twenty six updated")
h.add(887, "large number")

print(h)
```

In [ ]:
```python
def add(self, key, value):
    key_hash = self._get_hash(key)
    key_value = [key, value]

    if self.map[key_hash] is None:
        self.map[key_hash] = [key_value]
        return True

    else:
        # check if it's an update
        for pair in self.map[key_hash]:
            if pair[0] == key:
                print("updating: ", key)
                pair[1] = value
                return True

        # nope, it's a collision: insert it
        self.map[key_hash].append(key_value)
        return True
HashMap.add = add
```

In [ ]:
```python
h = HashMap()

h.add(17, "seventeen")
h.add(26, "twenty six")
h.add(35, "thirty five")
h.add(25, "twenty five")
h.add(26, "twenty six updated")
h.add(887, "large number")

print(h)
```

In [ ]:
```
def delete(self, key):
    key_hash = self._get_hash(key)

    if self.map[key_hash] is None :
        raise KeyError(str(key))

    for i in range(0, len(self.map[key_hash])):
        if self.map[key_hash][i][0] == key:
            self.map[key_hash].pop(i)
            return True

HashMap.delete = delete
```

In [ ]:
```
h.delete(17)
```

In [ ]:
```
print(h)
```