CSc 120
Introduction to Computer Programing II

Adapted from slides by Dr. Saumya Debray

01-e: Python review
python review:
dictionaries
Dictionaries

• A dictionary is like an array, but it can be indexed using strings (or numbers, or tuples, or any immutable type)
  • the values used as indexes for a particular dictionary are called its keys
  • think of a dictionary as an unordered collection of key : value pairs
  • empty dictionary: {}

• It is an error to index into a dictionary using a non-existent key
## Dictionary Summary

<table>
<thead>
<tr>
<th>Operation</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{}</code></td>
<td>Return an empty dictionary.</td>
</tr>
<tr>
<td><code>len(d)</code></td>
<td>Return the number of items in the dictionary <code>d</code>.</td>
</tr>
<tr>
<td><code>d[key]</code></td>
<td>Return the item of <code>d</code> with key <code>key</code>. Raises an error if <code>key</code> is not in the dictionary.</td>
</tr>
<tr>
<td><code>d[key] = value</code></td>
<td>Set <code>d[key]</code> to <code>value</code>.</td>
</tr>
<tr>
<td><code>del d[key]</code></td>
<td>Remove <code>d[key]</code> from <code>d</code>. Raises an error if <code>key</code> is not in the dictionary.</td>
</tr>
<tr>
<td><code>key in d</code></td>
<td>Return <code>True</code> in <code>d</code> has a key <code>key</code>, else <code>False</code>.</td>
</tr>
<tr>
<td><code>key not in d</code></td>
<td>Equivalent to not <code>key in d</code>.</td>
</tr>
<tr>
<td><code>keys()</code></td>
<td>Return the dictionary's keys.</td>
</tr>
<tr>
<td><code>values()</code></td>
<td>Return the dictionary's values.</td>
</tr>
<tr>
<td><code>items()</code></td>
<td>Return the dictionary's items as tuples.</td>
</tr>
</tbody>
</table>
The function `count_chars(s)` takes a string `s` and returns a dictionary of the counts of all characters in the string.

def count_chars(s):
    counts = {}
    s = s.lower()
    for c in s:
        if c in counts:  # if we have seen c, increment its count
            counts[c] = counts[c] + 1
        else:  # otherwise, it is the first occurrence
            counts[c] = 1
    return counts
2d-Dictionaries

```python
>>> mis_units = {'mis 101': 4, 'mis 102': 3, 'mis 202': 2}
>>> csc_units = {'csc 110': 4, 'csc 120': 4, 'csc 352': 3}
>>> ece_units = {'ece 111': 3, 'ece 222': 3, 'ece 333': 4}

>>> catalog = {'MIS': mis_units,
               'CSC': csc_units,
               'ECE': ece_units}
```
2d-Dictionaries

```python
>>> catalog

>>> for dept in catalog:
    print(dept, ':
        print(dept, ':', catalog[dept])

MIS : {'mis 101': 4, 'mis 102': 3, 'mis 202': 2}
CSC : {'csc 110': 4, 'csc 120': 4, 'csc 352': 3}
ECE : {'ece 111': 3, 'ece 222': 3, 'ece 333': 4}
```
Write a function `find_courses(catalog, units)` takes 2d-dictionary and an integer and returns a list of the courses of that many units.

```python
def find_courses(catalog, units):
    crs_list = []
    for dept in catalog:
        d = catalog[dept]
        for key in d:
            if d[key] == units:
                crs_list.append(key)
    return crs_list
```

• You don't need the intermediate variable `d`
Write a function find_courses(catalog, units) takes 2d-dictionary and an integer and returns a list of the courses of that many units.

```python
def find_courses(catalog, units):
    crs_list = []
    for dept in catalog:
        for key in catalog[dept]:
            if catalog[dept][key] == units:
                crs_list.append(key)
    return crs_list
```

• without the intermediate variable