CSC 120 **ICA-4**

Work with your neighbor. (This will be graded for participation only.)

1. Write a function `print_keys(d)` that prints the keys in the dictionary `d.` For example, if the dictionary passed in is

   {"I": 1, "V": 5, "X": 10, "L": 50}

   then the function prints the following:

```
'I'
'V'
'X'
'L'
```

2. Assume that the dictionary `d` and the list `words` are defined as follows:

```
>>> d = {}
>>> d['one'] = 1
>>> d['eight'] = 8
>>> d['two'] = 2
>>> d['seven'] = 7
>>> d['five'] = 5
>>>
>>> words = ["one","two","three","four","five", "six","seven","eight"]
```

   a) Write a loop that prints the values of `d` that are even.

   b) Write a loop that iterates through `words` and prints `True` for elements that are keys in `d` and `False` otherwise.

3. Write a function `key_of_max_value(adict)` that finds the maximum of all the values in the dictionary `adict` and returns the corresponding key. For example, if the dictionary passed in is

   ```
   {"hello" :  34, "sunny" : 51, "the" : 82, "street" : 13}
   ```

   then the function returns the key `"the"`. All the dictionary values are >= 0.

   Note: You'll have to iterate through the dictionary and keep track of the maximum value seen so far, but also keep track of the corresponding key for that value.

   ```
   def key_of_max_value(adict):
   ```

4. Write a function `identify_unique_words(slist)` that takes a list of strings `slist`. The function returns a dictionary where the keys are the strings in `slist` and the corresponding values are 0, if the string occurred only once in `slist`, and 1 otherwise. For example, if the function is called with the list

   ```
   ['here', 'is', 'the', 'root', 'of', 'the', 'root','and', 'the']
   ```

   then the dictionary returned is

   ```
   {'here': 0, 'is': 0, 'the': 1, 'root': 1, 'of': 0, 'and': 0}
   ```

   Notice that the strings that are unique in `slist` have a value of 0, and the words that are duplicates have a value of 1.