Cryptopals Preparation

Hi! I'm going to go over Cryptopals Set 1 on Monday. If you are currently in or past COMP 411 and know Python3, the demo should not be hard to understand, but if not, I strongly encourage you to read through this document.

1 Installation

- Feel free to code in any language or IDE.
- If you want to copy me, get Python3, pip3, and vim.
- Be able to import the Python3 modules base64, binascii, collections, and Crypto.Cipher. base64, binascii, and collections should have been installed by default, so you should only have to run the following command:

```
sudo pip3 install pycrypto
```

2 Functions

The first three functions are the most important to research:

- b64encode/b64decode (module base64)
- hexlify/unhexlify (module binascii)
- chr/ord (built-in)
- enumerate
- strip
- map
- join
- zip

A difficult one to grasp is hexlify. It converts strings into a hexadecimal representation. Try playing around with it in Python3's interactive shell (type python3 in the terminal). You can look up a command with

```
python3 -m pydoc base64.b64encode
python3 -m pydoc chr
```

3 Data structures

When I program in Python3, I'll use sets and dictionaries. Sets are unordered collections of unique items. On the other hand, dictionaries map values to keys. Dictionary values = array indices not limited to integers, while dictionary keys = array elements.

4 Bitwise operations

Familiarity with bitwise operations will be essential when we calculate Hamming distances. Assuming x is a binary number (only 1's and 0's):

- `x & 1` retrieves its last bit

  ```
  1001 & 1 = 1
  1000 & 1 = 0
  ```

  Note: `1001 & 1 = 1001 & 0001 = 0001` if we want to be more precise

- `x >> y` shifts x to its left by y

  ```
  1001 >> 1 = 100
  1001 >> 2 = 10
  ```

  Note: `1001 >> 1 = 0100` if we want to be more precise

- `x ^ y` returns 0 if both bits are equal, else 1, if x and y are 0 or 1

5 List comprehensions

Finally, list comprehensions in Python3 are essentially one line for loops. The following two are the same:

```python
y = []
for x in range(4):
    y.append(x)

y = [x for x in range(4)]

# Note: Printing y would result in [0, 1, 2, 3]
```

6 End

I hope you found some of this useful. Feel free to check out the demo repository ahead of time. If you have any questions, feel free to email me at lobcus@live.unc.edu or message me on the Slack (lobcus)!