Lab 3: Python Encoding

Work through the first few problems on paper before starting Python. ***Box*** your answers.

1. Predict how the `bytes` object `b'2 Faced'` will be stored in Python. ***Write*** your answer in hexadecimal shorthand.

2. Predict how the bytes object b'\r\n' will be stored in Python. ***Write*** your answer in both **hexadecimal shorthand** and **bits**.

3. Predict how the number `104` will be stored in Python. ***Write*** your answer in **binary**, then ***write*** it in **hexadecimal shorthand**.

4. Predict how the number 0xfe19 will be stored in Python. ***Write*** your answer in **hexadecimal shorthand**, then ***write*** it in **binary**.

5. Predict how the bytes object b'\xbeef' will be stored in Python. ***Write*** your answer in **hexadecimal shorthand**, then ***write*** it in **binary**.

6. Predict how the number `1055` will be stored in Python. ***Write*** your answer in **binary**, then ***write*** it in **hexadecimal shorthand**.

7. If you have extra time, repeat for the numbers 221, 5262, 159, 348, 3294, 246, 4289, and 255.

8. Set up the showbits library:
    a. Go to the Lab 3 webpage and download the python module `showbits.py`.
    b. Place the file directly inside the top-level of your Python project.
    c. Open Settings (Ctrl-Alt-S). Type "Python Console" in the search bar. Add the full path to your project as the Working directory. For example, I added C:\d\Dropbox\PyCharm18q1, since that is the path to my Python project PyCharm18q1.
    d. Open the Python console using Tools -> Python Console.
    e. In the console, type `from showbits import bits, shorthand`. (If you use this in a file, use `import showbits` instead, and use `showbits.bits()` with the package-name when calling `bits()`.)

As you check your answers to your previous problems, ***write*** down what you learn from the differences between your predictions and the actual values.

9. ***Check*** your answer to Problem 1 by typing `shorthand(b'2 Faced')`.

10. ***Check*** your answer to Problem 2 by typing `shorthand(b'\r\n')`.

11. ***Check*** your answers to Problem 3 by typing `bits(104)` and `shorthand(104)`.

12. ***Check*** your answers to Problem 4 by typing `shorthand(0xfe19)` and `bits(0xfe19)`.

13. ***Check*** your answers to Problem 5 by typing `shorthand(b'\xbeef')` and `bits(b'\xbeef')`.

14. ***Check*** your answers to Problem 5 by typing `bits(1055)` and `shorthand(1055)`.

15. If you have time, ***continue checking*** your answers to Problem 7.

16. ***Determine*** the type of `int`. Set `i = 3`. ***Determine*** the type of `i`. ***Write*** the two types.

17. ***Assign*** the number 1000_{10} to a variable. Store the contents of this variable in a 16-bit Python **bytes** object. Looking back at your notes, check that the bytes object has the correct values in it. ***Write*** the Python code you used here:

18. In Problem 9, you created the bytes object `b'2 Faced'` by simply typing it into Python. Now, create the bytes object by ***storing*** the hexadecimal shorthand you found in Problem 1 into a variable. For example, to store the hexadecimal shorthand 12 34 FF into a variable, you could type `i=0x1234ff`. Next, ***transfer*** the contents of that variable into a Python bytes object, just as you did with the number 1000 in Problem 17. ***Display*** the bytes object to check if it is `b'2 Faced'`. ***Write*** the Python code you used just for the transfer here:

19. In Problem 14, you found the hexadecimal shorthand for 1055. ***Pad*** this (if needed) to create a two-byte number, and write it as a bytes object: b'\x__ __\x__ __'. Next, ***transfer*** the contents of the bytes object back to an integer. It should be the number 1055. ***Write*** the Python code you used here:

20. Write Python code to store the number 3338 as an int and transfer it to a two-byte `bytes` object. Display this `bytes` object in Python. ***Write*** what is displayed and your Python code here:

21. Write Python code to store the number 1885889911 as an int and transfer it to a four-byte `bytes` object. Display this `bytes` object in Python. ***Write*** what is displayed and your Python code here: (You can also try 6644322 and 1885888884.)

22. Describe how you can determine whether it is an integer or ASCII text that is stored in a `bytes` object.