

hw0

Coding the Matrix, Summer 2013

Please fill out the stencil file named “hw0.py”. While we encourage you to complete the Ungraded Problems, they do not require any entry into your stencil file.

Python Comprehension Problems

Write each of the following 3 procedures using a comprehension:

Problem 1: `my_filter(L, num)`

input: list of numbers and a number.

output: list of numbers not containing a multiple of `num`.

example: given `list = [1,2,4,5,7]` and `num = 2`, return `[1,5,7]`.

Problem 2: `my_lists(L)`

input: list `L` of non-negative integers.

output: a list of lists: for every element x in `L` create a list containing $1, 2, \dots, x$.

example: given `[1,2,4]` return `[[1], [1,2], [1,2,3,4]]`.

Problem 3: `my_function_composition(f,g)`

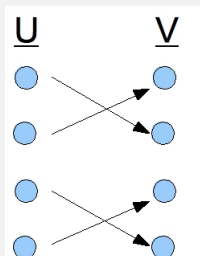
input: two functions f and g , represented as dictionaries, such that $g \circ f$ exists.

output: dictionary that represents a function $g \circ f$.

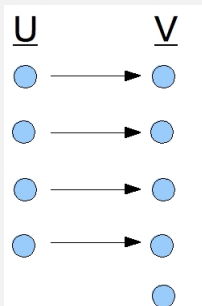
example: given $f = \{0:'a', 1:'b'\}$ and $g = \{'a':\text{'apple'}, 'b':\text{'banana'}\}$, return $\{0:\text{'apple'}, 1:\text{'banana'}\}$.

Functional Inverses

Ungraded Problem: Is the following function invertible? If yes, explain why. If not, can you change domain and/or codomain of the function to make it invertible?



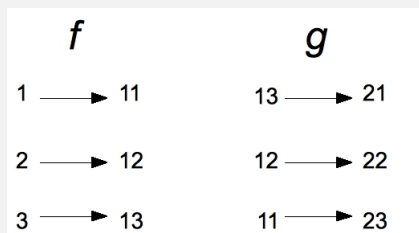
Ungraded Problem: Is the following function invertible? If yes, explain why. If not, can you change domain and/or codomain of the function to make it invertible?



Functional Composition

Ungraded Problem: Let $f : \mathbb{R} \rightarrow \mathbb{R}$ where $f(x) = \text{abs}(x)$. Is there a choice of domain and co-domain for the function $g(x)$ with rule $g(x) = \sqrt{x}$ such that $g \circ f$ is defined? If so, specify it. If not, explain why not. Could you change domain and/or codomain of f or g so that $g \circ f$ will be defined?

Ungraded Problem: Consider functions f and g in the following figure:



Is $f \circ g$ defined? If so, draw it, otherwise explain why not.

Complex Addition Practice

Problem 4: Each of the following asks for the sum of two complex numbers. For each, find the solution.

- $(3 + 1i) + (2 + 2i)$
- $(-1 + 2i) + (1 - 1i)$
- $(2 + 0i) + (-3 + .001i)$
- $4(0 + 2i) + (.001 + 1i)$

GF(2) Arithmetic

Problem 5: For each of the following, calculate the answer over $GF(2)$.

- a. $1 + 1 + 1 + 0$
- b. $1 \cdot 1 + 0 \cdot 1 + 0 \cdot 0 + 1 \cdot 1$
- c. $(1 + 1 + 1) \cdot (1 + 1 + 1 + 1)$

Python loop problems

For procedures in the following problems, use the following format:

```
def <ProcedureName>(L):  
    current = ...  
    for x in L:  
        current = ...  
    return current
```

The value your procedure initially assigns to **current** turns out to be the return value in the case when the input list **L** is empty. This provides us insight into how the answer should be defined in that case. Note: You are **not** allowed to use Python built-in procedures **sum(.)** and **min(.)**.

Summing numbers in a list

Problem 6: `mySum(L)`
Input: list of numbers
Output: sum of numbers in the list

Multiplying numbers in a list

Problem 7: `myProduct(L)`
Input: list of numbers
Output: product of numbers in the list

Minimum of a list

Problem 8: `myMin(L)`
Input: list of numbers
Output: minimum number in the list

Concatenation of a List

Problem 9: myConcat(L)

Input: list of strings

Output: concatenation of all the strings in L

Union of Sets in a List

Problem 10: myUnion(L)

Input: list of sets

Output: the union of all sets in L.

In each of the above problems, the value of **current** is combined with an element of **myList** using some operation \diamond . In order that the procedure return the correct result, **current** should be initialized with the *identity element* for the operation \diamond , i.e. the value i such that $i \diamond x = x$ for any value x .

It is a consequence of the structure of the procedure that, when the input list is empty, the output value is the initial value of **current** (since in this case the body of the loop is never executed). It is convenient to define this to be the correct output!

Identity Values

Ungraded Problem: Keeping in mind the comments above, what should be the answer for each of the following?

1. the sum of the numbers in an empty set;
2. the product of the numbers in an empty set;
3. the minimum of the numbers in an empty set;
4. the concatenation of an empty list of strings;
5. the union of an empty list of sets

What goes wrong when we try to apply this reasoning to define the intersection of an empty list of sets?