HW09

Sam Ly

November 9, 2025

Required Exercise 1 [4]

Included below.

Required Exercise 2 [3]

Proposition 1. *A function $f : A \to B$ is a bijection if and only if there exists a function $G : B \to A$ such that $g(f(a)) = a$ for all $a \in A$ and $f(g(b)) = b$ for all $b \in B$.*

1. Prove the "forward direction" of the proposition by assuming that $f : A \to B$ is surjective and injective, and concluding that there exists an inverse function.

   Proof. Suppose that $f : A \to B$ is a bijection. Thus, for every unique $b \in B$ there exists a unique $a \in A$ such that $f(a) = b$.

   Therefore, there must exist the inverse function $f^{-1} : B \to A$ such that for every value $b \in B$, $f^{-1}(b)$ is a unique value $a \in A$. □

2. Prove the "backward direction" of the proposition by assuming that $f^{-1} : B \to A$ is an inverse function of f, and concluding that f is surjective and injective.

   Proof. First, we see that because $f{-}1$ exists, f itself must be a valid fumction. f^{-1} maps every value $b \in B$ to a unique value $a \in A$. So, f must be injective because the domain of f^{-1} is B. Also, because f^{-1} is a valid function, no two values $a \in A$ map to the same value $b \in B$. Thus, f is injective. □

3. (a) Give an example of sets A and B together with a pair of functions $f : A \to B$ and $g : B \to A$ where
      i. $g(f(a)) = a$ for all $a \in A$,
      ii. there exists $b \in B$ such that $f(g(b) \neq b$, and
      iii. f is not a bijection.

   Let sets $A = \mathbb{N}$ and $B = \mathbb{R}$. Let $f : \mathbb{N} \to \mathbb{R}$, where $f(n) = n$. Let $g : \mathbb{R} \to \mathbb{N}$, where $g(r) = \lfloor r \rfloor$.
   We see that for all $a \in A$, $g(f(a)) = a$. However, for any non-integer value $b \in B$, $f(g(b)) \neq b$. Thus, f is not a bijection.

   (b) Let sets $A = \mathbb{R}$ and $B = \{r \in \mathbb{R} : 0 \leq r \leq 1\}$. For $f : A \to B$, $f(x) = \sin(x)$. and $g : B \to A$, $g(x) = \sin^{-1}(x)$.
   We see that for $a = 4\pi \in A$, $g(f(a)) = 0 \neq 4\pi$. However, $f(g(b)) = b$ for all $b \in B$.

Required Exercise 3 [3]

1. Give an example of a bijection $h : \mathbb{N}_{>0} \to \mathbb{N}_{\geq 0}$ Note that the domain and the codomain are different, and explain why the map $h(x) = x$ is not a bijection.

   $h(x) = x - 1$ is a valid bijection. $h(x) = x$ is not a bijection because it is not surjective. The value 0 is not reached.

2. Consider the function $f : \mathbb{N}_{>0} \to \mathbb{N}_{\geq 0}$ where

$$f(n) = \begin{cases} \frac{n}{2}, & \text{if } n \text{ is even} \\ -\frac{(n+1)}{2}, & \text{if } n \text{ is odd.} \end{cases}$$

   (a) Prove that f is a bijection by proving that it is both injective and surjective.

   Proof. To see that f is both injective and surjective, we must first see that the individual "peices" of f are each injective and surjective, and that these "pieces" have mutually exclusive codomains.
   First, we see that for even n, $f(n) = n/2$. This is itself a valid bijection between the set A of even natural numbers including 0 and the set B of all natural numbers including 0.
   Then, we see that for odd n, $f(n) = -\frac{n+1}{2}$. This is a valid bijection between the set C of all odd natural numbers and the set D of all negative integers.
   Notice that the range of both "pieces" is equal to their codomains.
   Now, we see that $B \cap D = \emptyset$, $A \cap C = \emptyset$, and $A \cup B = \mathbb{N}_{\geq 0}$. So, f must be injective.
   Also, because $B \cup D = \mathbb{Z}$, f must be surjective.
   Therefore, f is a bijection. □

   (b) Prove that f is a bijection by describing a (piecewise-defined) function for the inverse map $g : \mathbb{Z} \to \mathbb{N}_{\geq 0}$, and checking that $g \circ f : \mathbb{N}_{\geq 0} \to \mathbb{N}_{\geq 0}$ and $f \circ g : \mathbb{Z} \to \mathbb{Z}$ are both the identity function on their respective domains.

   Proof. Let $g : \mathbb{Z} \to \mathbb{N}_{\geq 0}$, where

$$g(n) = \begin{cases} 2n & n \geq 0 \\ -2n - 1 & n < 0 \end{cases}$$

   Now, we observe that $g \circ f$ is the identity function because for all even $n \in \mathbb{N}$, $f(n) = n/2$. Because $n \geq 0$, $f(n) \geq 0$. Thus, $g(n) = 2n$. Finally, $g(f(n)) = n$.
   Then, for all odd $n \in \mathbb{N}$, $f(n) = -\frac{n+1}{2}$. Now, $f(n) < 0$, so $g(n) = -2n - 1$. Thus,

$$\begin{aligned} g(f(n)) &= -2(-\frac{n+1}{2}) - 1 \\ &= (n+1) - 1 = n. \end{aligned}$$

   Thus, $g \circ f$ is the identity on $\mathbb{N}_{\geq 0}$.
   Also, $f \circ g$ is the identity on $\mathbb{Z}$ because for $n \geq 0$, $g(n) = 2n$ is even, and $f(n) = n/2$. Thus, $f(g(n)) = n$. Then for $n < 0$, $g(n) = -2n - 1$ and $f(n) = -\frac{n+1}{2}$. Thus,

$$\begin{aligned} f(g(n)) &= -\frac{(-2n-1)+1}{2} \\ &= -\frac{-2n}{2} = n. \end{aligned}$$

   Therefore, f is a bijection. □

3. Describe a bijection $s : \mathbb{N}_{>0} \to \mathbb{Z}$ in ters of h and f.

   Since $h : \mathbb{N}_{>0} \to \mathbb{N}_{\geq 0}$ and $f : \mathbb{N}_{\geq 0} \to \mathbb{Z}$, $s = f \circ h$.

Choice Exercise 6 [5]

In this exercise, you will prove that the relation "there exists a bijection between" is an equivalence relation on sets.

1. Prove that the relation is reflexive: for all sets A, there exists a bijection $f : A \to A$.

   Proof. For any set A, there must exist the identity function f. This function must be a bijection since it relates all elements of set A, to a unique elemeent of A. □

2. Prove that the relation is symmetric: for all sets A and B such that there exists a bijection $f : A \to B$, there also exists a bijection $g : B \to A$.

   Proof. For a function f to be well defined on a domain A, it must be defined for every value $a \in A$. This means that, if f is injective, there bust exist a surjection from B to A. Because f is a bijection, f must also be injective, thus there exists some surjection from B to A.

   Also, since f is a surjection and well-defined, there must exist an injection from B to A. Therefore, there must exist a bijection from $g : B \to A$. □

3. Prove that the relation is transitive: for all sets A, B, and C such that there exist bijections $f_1 : A \to B$ and $f_2 B \to C$, there also exists a bijection $f_3 : A \to C$.

   Proof. Assume that there exists bijections $f_1 : A \to B$ and $f_2 B \to C$.

   This means that every value $a \in A$ can be mapped to a unique value $b \in B$ by a the function f_1. Also, every value $b \in B$ is mapped to by a value $a \in A$. Similarly, every value $b \in B$ is mapped to a unique value $c \in C$ by the function f_2, and all values of $c \in C$ are mapped to by a value $b \in B$.

   So, this means that the composition $f_2 \circ f_1$ first maps all unique values of $a \in A$ to a unique value $b \in B$, then maps all values $b \in B$ to a unique value $c \in C$. This means that all elements $a \in A$ can be mapped to a unique element $c \in C$. Also, this mapping is surjective because all values of C are reached by f_2, and all values of B are reached by f_1.

   Therefore, this relation is transitive. □

Choice Exercise 7 [6]

2. (a)
```python
from collections import deque
from csv import Error
from typing import Iterator, TypeVar
from itertools import islice, tee

# 2. a.
def f(n: int):
    if n < 0:
        raise Error("n must be a natural number")

    if n % 2 == 0:
        return n // 2
    return - (n + 1) // 2

print("2. a. ")
print([f(n) for n in range(11)])
```

```
2. a.
[0, -1, 1, -2, 2, -3, 3, -4, 4, -5, 5]
```

(b)
```python
def g(n: int):
    if n >= 0:
        return 2 * n
    return -2*n - 1

print("2. b. ")
# 2. b. i.
print([g(n) for n in range(-5, 6)])

# 2. b. ii.
print([f(g(n)) for n in range(-5, 6)])

# 2. b. iii.
print([g(f(n)) for n in range(0, 11)])
```

```
2. b.
[9, 7, 5, 3, 1, 0, 2, 4, 6, 8, 10]
[-5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

(c) Here, my implementation may seem a bit unorthodox. If you are familiar with Generators in Python, then skip ahead to the implementation.

Rather than defining an actual function h that receives a natural number and outputs a specific tuple $t \in \mathbb{N} \times \mathbb{N}$, I return a Generator object that allows you to iterate through an infinitely many tuples $t \in \mathbb{N} \times \mathbb{N}$. This way, $h(1)$ is the first elements returned in the iteration, and $h(2)$ is the second, etc.

Note: I use the type hint Iterator rather than Generator because All Generators are Iterators. This works just fine if done with Iterator objects.

```python
#  2. c.
def N() -> Iterator[int]:
    i = 0
    while True:
        yield i
        i += 1

R = TypeVar("R")
S = TypeVar("S")
type Tree[T] = T | tuple[Tree[T], Tree[T]]
def forward_replay(g: Iterator[R]) -> Iterator[R]:
    seen = []
    for elem in g:
        seen.append(elem)
        yield from seen

def reverse_replay(g: Iterator[R]) -> Iterator[R]:
    seen = []
    for elem in g:
        seen.append(elem)
        yield from reversed(seen)

def cartesian(a: Iterator[R], b: Iterator[S]) -> Iterator[tuple[R, S]]:
    f = forward_replay(a)
    r = reverse_replay(b)
    yield from zip(f, r)

def h() -> Iterator[tuple[int, int]]:
    yield from cartesian(N(), N())

print("2. c. ")
print(list(islice(h(), 11)))
```

```
2. c.
```

```
[(0, 0), (0, 1), (1, 0), (0, 2), (1, 1), (2, 0), (0, 3), (1, 2), (2, 1), (3, 0),
 (0, 4)]
```

(d) The logic used in the previous part of this exercise can be generalized to any countably infinite sets. (could there be some 'bijection' between countably infinite sets and Generator objects?)

```python
# 2. d.
def A1() -> Iterator[int]:
    # Fibonacci!
    a, b = 0, 1
    yield a
    yield b

    while True:
        c = a + b
        a, b = b, c
        yield c

print("2. d. ")
print("A1: ", list(islice(A1(), 11)))

def A2(s: str) -> Iterator[str]:
    # All finite strings for an alphabet s
    frontier = deque([""])
    while True:
        curr = frontier.popleft()
        yield curr
        for c in s:
            frontier.append(curr + c)

print("A2: ", list(islice(A2("01"), 11)))

def f_prime():
    yield from cartesian(A1(), A2("01"))


print("f_prime: ", list(islice(f_prime(), 11)))
```

```
2. d.
A1:  [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
A2:  ['', '0', '1', '00', '01', '10', '11', '000', '001', '010', '011']
f_prime:  [(0, ''), (0, '0'), (1, ''), (0, '1'), (1, '0'), (1, ''), (0, '00'),
(1, '1'), (1, '0'), (2, ''), (0, '01')]
```

(e) We define a function to take the cartesian product of an infinite set (represented as a Generator) with itself k times.

Now, this creates some odd data structures since my implementation of the cartesian product creates a tuple of the two original data types. When taking many cartesian products, you end up with deeply nested tuples. Luckily, these tuples are isomorphic to its own flattened counterpart. In other words, there exists a bijection between arbitrarily nested tuples (a tree of sorts), to an arbitrarily long 1-dimensional tuple that can be expressed in code.

```python
# 2. e.
def set_pow(it: Iterator[R], pow: int) -> Iterator[Tree[R]]:
    if pow < 1:
        raise Exception("Pow must be >=1")

    if pow == 1:
        return it

    it1, it2 = tee(it, 2)

```

```
    if pow % 2 == 0:

        return cartesian(*tee(set_pow(it1, pow//2), 2))

    return cartesian(it1, set_pow(it2, pow-1))


def b_k(k: int):
    yield from set_pow(N(), k)


def flatten_tree(t: Tree[R]) -> R | tuple[R, ...]:
    if not isinstance(t, tuple):
        return t

    def leaves(node: Tree[R]) -> Iterator[R]:
        if isinstance(node, tuple):
            # node is a pair of subtrees
            left, right = node
            yield from leaves(left)
            yield from leaves(right)
        else:
            # node is a leaf
            yield node

    return tuple(leaves(t))

def b_k_flat(k: int):
    for val in set_pow(N(), k):
        yield flatten_tree(val)

print("2. e.")
print("b_k: ", list(islice(b_k(4), 3)))
print("b_k_flat: ", list(islice(b_k_flat(4), 11)))
```

```
2. e.
b_k:  [((0, 0), (0, 0)), ((0, 0), (0, 1)), ((0, 1), (0, 0))]
b_k_flat:  [(0, 0, 0, 0), (0, 0, 0, 1), (0, 1, 0, 0), (0, 0, 1, 0),
 (0, 1, 0, 1), (1, 0, 0, 0), (0, 0, 0, 2), (0, 1, 1, 0), (1, 0, 0, 1),
 (0, 2, 0, 0), (0, 0, 1, 1)]
```