

hw1

January 17, 2023

1 ECE 60146 HW 1

1.0.1 Akshita Kamsali

1.0.2 akamsali@purdue.edu

1.0.3 1. Creating a class Sequence

Creating `__init__` and overloading `__gt__` in Sequence class

```
[1]: class Sequence(object):
    def __init__(self, array: list) -> None:
        self.array = array

    # overload the ">" operator using the __gt__ method
    def __gt__(self, second: 'Sequence') -> bool:
        # check if the two arrays are of equal length
        if len(self.array) != len(second.array):
            raise ValueError('Two arrays are not equal in length!')
        # initialise the count to zero
        num_gt = 0
        for i in range(len(self.array)):
            if self.array[i] > second.array[i]:
                num_gt += 1
        return num_gt
```

1.0.4 2. Creating a class Fibonacci

Create class and overload `__call__`, `__len__`, `__iter__` Outputs are shown below

```
[2]: class Fibonacci(Sequence):
    def __init__(self, first_value: int, second_value: int) -> None:
        super().__init__([ ])
        self.first_value = first_value
        self.second_value = second_value

    def __call__(self, length=5) -> list:
        # initialise the array with the first two values
        self.array = [self.first_value, self.second_value]
```

```

        for i in range(length - 2):
            # adding last two numbers in the array
            self.array.append(self.array[-1] + self.array[-2])
        return self.array

    def __len__(self):
        return len(self.array)
    def __iter__(self):
        return F_iterator(self)

# reference from Avi's slides
class F_iterator:
    def __init__(self, F):
        self.F = F.array
        self.index = 0

    def __next__(self):
        if self.index >= len(self.F):
            raise StopIteration
        else:
            self.index += 1
            return self.F[self.index - 1]

    def __iter__(self):
        return self

FS = Fibonacci(1, 2)
print("Function call: ", FS(length=5))
print("Length: ", len(FS))
print("Iterable: ", [n for n in FS])

```

Function call: [1, 2, 3, 5, 8]
 Length: 5
 Iterable: [1, 2, 3, 5, 8]

1.0.5 3. Creating a class Prime

Create class and overload `__call__`, `__len__`, `__iter__` Outputs are shown below

```

[3]: class Prime(Sequence):
    def __init__(self):
        Sequence.__init__(self, [])

    def isPrime(self, n: int):
        for i in range(2, n//2 + 1):
            if n % i == 0:
                return 0

```

```

        return 1

    def __call__(self, length=5):
        self.array = []
        i = 2
        while len(self.array) < length:
            if self.isPrime(i):
                self.array.append(i)
            i += 1
        return self.array

    def __len__(self):
        return len(self.array)

    def __iter__(self):
        return F_iterator(self)

PS = Prime()
print("Function call: ", PS(length=8))
print("Length: ", len(PS))
print("Iterable: ", [n for n in PS])

```

Function call: [2, 3, 5, 7, 11, 13, 17, 19]
 Length: 8
 Iterable: [2, 3, 5, 7, 11, 13, 17, 19]

1.0.6 4. Comparing classes Prime and Fibonacci with > operator

Outputs are shown below for overloaded > operator

```

[4]: FS = Fibonacci(1, 2)
print("FS call to len 8: ", FS(length=8))
PS = Prime()
print("PS call to len 8: ", PS(length=8))
print("FS > PS call: ", FS > PS)
print("PS call to len 5: " )
PS(length=5)
print("FS > PS call: ", FS > PS)

```

FS call to len 8: [1, 2, 3, 5, 8, 13, 21, 34]
 PS call to len 8: [2, 3, 5, 7, 11, 13, 17, 19]
 FS > PS call: 2
 PS call to len 5:

```

-----
ValueError                                Traceback (most recent call last)
/Users/akshita/Documents/Acads/ECE60146/HW1/hw1.ipynb Cell 9 in <cell line: 8>

```

```

    <a href='vscode-notebook-cell:/Users/akshita/Documents/Acads/ECE60146/HW1.
↪hw1.ipynb#X11sZmlsZQ%3D%3D?line=5'>6</a> print("PS call to len 5: " )
    <a href='vscode-notebook-cell:/Users/akshita/Documents/Acads/ECE60146/HW1.
↪hw1.ipynb#X11sZmlsZQ%3D%3D?line=6'>7</a> PS(length=5)
----> <a href='vscode-notebook-cell:/Users/akshita/Documents/Acads/ECE60146/HW1.
↪hw1.ipynb#X11sZmlsZQ%3D%3D?line=7'>8</a> print("FS > PS call: ", FS > PS)

/Users/akshita/Documents/Acads/ECE60146/HW1/hw1.ipynb Cell 9 in Sequence.
↪__gt__(self, second)
    <a href='vscode-notebook-cell:/Users/akshita/Documents/Acads/ECE60146/HW1.
↪hw1.ipynb#X11sZmlsZQ%3D%3D?line=5'>6</a> def __gt__(self, second: 'Sequence')
↪-> bool:
    <a href='vscode-notebook-cell:/Users/akshita/Documents/Acads/ECE60146/HW1.
↪hw1.ipynb#X11sZmlsZQ%3D%3D?line=6'>7</a>         # check if the two arrays are of
↪equal length
    <a href='vscode-notebook-cell:/Users/akshita/Documents/Acads/ECE60146/HW1.
↪hw1.ipynb#X11sZmlsZQ%3D%3D?line=7'>8</a>         if len(self.array) != len(second
↪array):
----> <a href='vscode-notebook-cell:/Users/akshita/Documents/Acads/ECE60146/HW1.
↪hw1.ipynb#X11sZmlsZQ%3D%3D?line=8'>9</a>         raise ValueError('Two arrays
↪are not equal in length!')
    <a href='vscode-notebook-cell:/Users/akshita/Documents/Acads/ECE60146/HW1/
↪hw1.ipynb#X11sZmlsZQ%3D%3D?line=9'>10</a>         #initialise the count to zero
    <a href='vscode-notebook-cell:/Users/akshita/Documents/Acads/ECE60146/HW1/
↪hw1.ipynb#X11sZmlsZQ%3D%3D?line=10'>11</a>         num_gt = 0

ValueError: Two arrays are not equal in length!

```