CMPT 423/820

Assignment 1 Question 2   ¶

- Seyedeh Mina Mousavifar
- 11279515
- sem311

Python Basics

- Data: integers, floating point, strings, Boolean, None
  - All data in Python is stored as objects, even numeric values
  - integers in Python are not limited in size
  - floating point values in Python are standard IEEE floating point values, and are limited in range.
  - These basic data types are immutable
  - type conversion functions `int() float() str()`
- Variables and expressions:
  - Every variable in Python stores a reference to an object
  - assignment statements
  - Arithmetic: `+ * - / // %`
  - Relational `== < > <= >= !=`
  - Boolean: `and or not`
- Sequential data structures: lists, tuples
  - Lists are mutable, tuples are immutable
  - Integer indexing for lists and tuples
  - Slicing lists creates new list object
- Dictionaries (associational)
  - Mutable
  - indexable by integers, strings, or tuples containing immutable data
- Python Control Flow
  - if-statements
  - for loops
  - while loops
  - function definitions
    - Since all variables store references, all Python parameters are pass-as-reference
    - By default, a function returns the value `None`.
- Console output
  - `print()`
  - String method `format()`

Tasks

Task 1.

Write Python code to calculate Fibonacci numbers. Use this code to produce the following output ("console output" using the function `print()`):

```
The 111 th Fibonacci number is 70492524767089125814114
As a float, this number is 7.049252476708912e+22
The integer difference between the two values is: 1634146
```

For the sake of this question, assume that the Fibonacci sequence starts with 0, 1, 1, 2, ... Also assume that the 0th Fibonacci number is 0, and the 3rd fibonacci number is 2.

Notice that the difference is non-zero. The integer value is exact. The floating point value is inexact due to truncation).

```python
In [11]:  from decimal import Decimal

          def fibonacci(number):
              """
              :purpose: This function calculates nth Fibonacci number, which is Fn
          = Fn-1 + Fn-2
              :param number: nth Fibonacci number to be calculated
              :return: value of nth number in Fibonacci sequence
              """
              nth_number = 1
              previous_number = 0
              if number == 0:
                  return 0
              elif number == 1:
                  return 1
              else:
                  for j in range(2, number + 1):

                      temp = previous_number + nth_number
                      previous_number = nth_number
                      nth_number = temp

                  return nth_number

          n = 111
          fibonacci_number = fibonacci(n)

          # print function uses str() implicitly
          print('The', n, 'th Fibonacci number is', fibonacci_number)

          float_number = float(fibonacci_number)
          print('As a float, this number is', float_number)

          difference =  fibonacci_number - int(float_number)
          print('The integer difference between the two values is:', difference)
```

```
The 111 th Fibonacci number is 70492524767089125814114
As a float, this number is 7.049252476708912e+22
The integer difference between the two values is: 1634146
```

Task 2. Lists and slicing

Write Python to

1. Store the first 50 Fibonacci numbers in a list.
2. Use this list, and slicing, to construct a list containing only 30th-34th Fibonacci numbers.
3. Display the sublist
4. Using list indexing, modify the sublist making the first entry `12345`
5. Display the sublist again
6. Display the corresponding sequence from the original list.

Here's an example of the output:

```
The sublist of Fibonacci numbers (before): [832040, 1346269, 2178309, 352457
8, 5702887]
The sublist of Fibonacci numbers (after): [12345, 1346269, 2178309, 3524578,
5702887]
The original sequence of Fibonacci numbers: [832040, 1346269, 2178309, 35245
78, 5702887]
```

Slicing is really really useful. This requirement is a very simple use of the concept.

Note: The Numpy module has a different, but similar operation, called a view. Python slicing creates a new object, always. Numpy views are not new objects.

```
In [12]: fibonacci_sequence = [fibonacci(i) for i in range(50)]

         fibonacci_sublist = fibonacci_sequence[30:35] #to preserve 30-34 in the
          list

         print('The sublist of Fibonacci numbers (before):', fibonacci_sublist)

         fibonacci_sublist[0] = 12345
         print('The sublist of Fibonacci numbers (after):', fibonacci_sublist)

         print('The original sequence of Fibonacci numbers:', fibonacci_sequence[
         30:35])
```

```
The sublist of Fibonacci numbers (before): [832040, 1346269, 2178309, 3
524578, 5702887]
The sublist of Fibonacci numbers (after): [12345, 1346269, 2178309, 352
4578, 5702887]
The original sequence of Fibonacci numbers: [832040, 1346269, 2178309,
3524578, 5702887]
```

What to hand in

Your version of this notebook named A1Q2.pdf, containing completed work above, and your name and student number at the top.

Evaluation:

- 2 marks: Your code cell for Task 1 uses Python (only -- no imported modules) to show the given output.
- 2 marks: Your code cell for Task 2 uses Python (only -- no imported modules) to show the given output