Problem Solving Methodology in IT (COMP1001)

Assignment Eight

(Due at noon on 26 Nov 2018)

**(Instructions: see the instructions in Blackboard)**

Rocky K. C. Chang

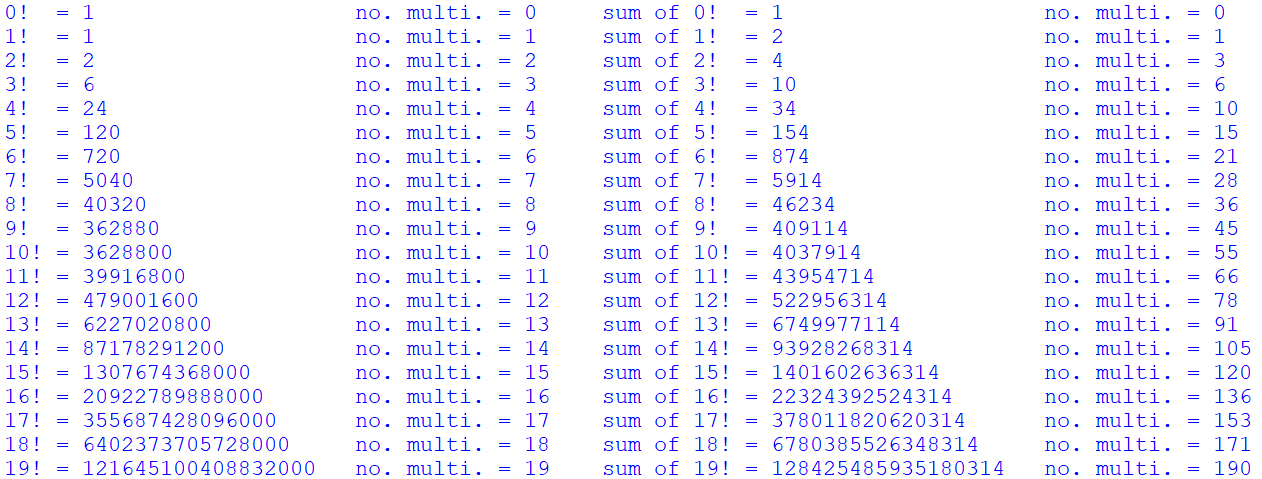
14 November 2018

1. [Weight = 8] We consider two functions of computing *n*! and , where *n* is a non-negative integer. Their code is given below.

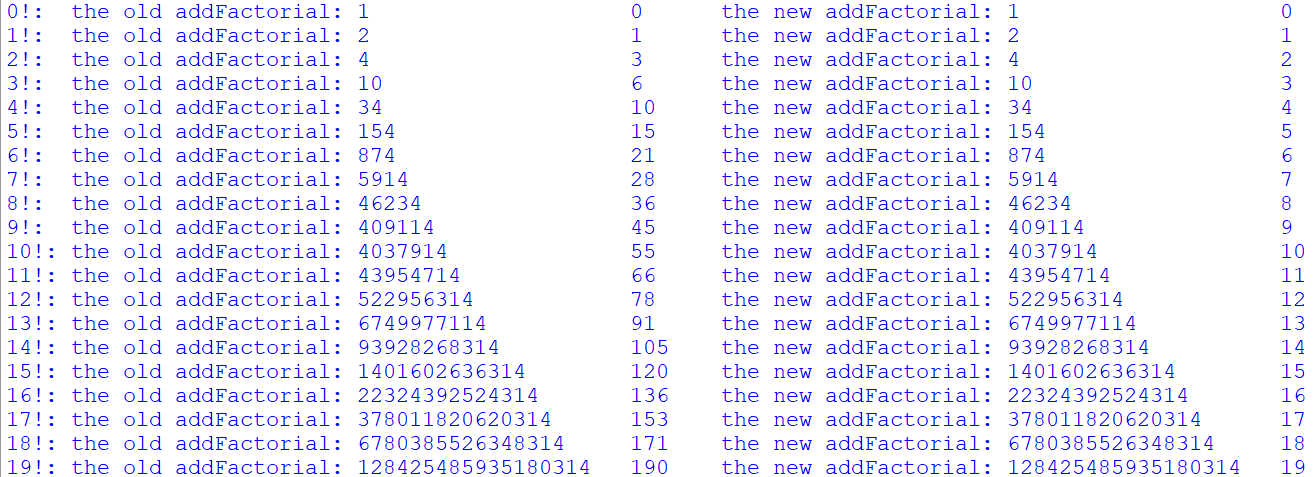
|  |  |
| --- | --- |
| *def factorial(n):*  *# Input: n, a non-negative integer*  *# Output: return n!*  *if n == 0:*  *return 1*  *else:*  *result = 1*  *for i in range(1,n+1):*  *result = result \* i*  *return result* | *def addFactorial(n):*  *# Input: n, a non-negative integer*  *# Output: 0!+1!+…,+n!*  *sum = 0*  *for i in range(n+1):*  *sum += factorial(i)*  *return sum* |

In this question we would like to measure the time complexity of these two functions. We use the number of multiplications as the measure for the time complexity.

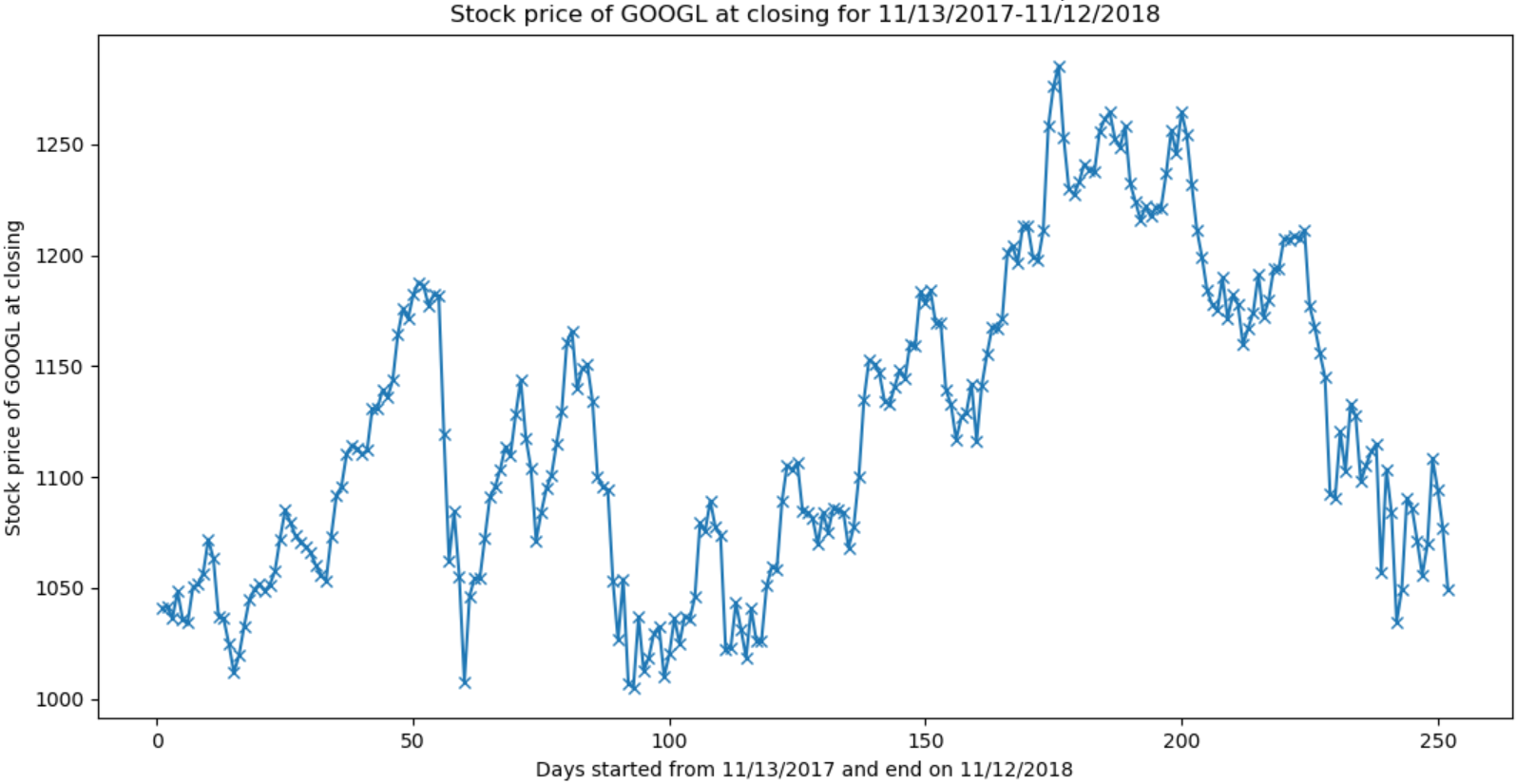
1. [Weight = 3] Modify the two functions above, so that each will return the result and the number of multiplications required for computing the result. Moreover, include an iterative *print*() statement to print out the results as below.



1. [Weight = 1] What is the number of multiplications required for invoking *addFactorial*(*n*)? Include your answer in the .*py* file.
2. [Weight = 3] Propose another implementation of *addFactorial*(*n*) that will require much less number of multiplications. Moreover, include an iterative *print*() statement to print out the results as below for the original *addFactorial*(*n*) and the improved *addFactorial*(*n*) you have implemented.
3. [Weight=1] What is the number of multiplications required for invoking the improved *addFactorial*(*n*)? Include your answer in the .*py* file.



1. [Weight = 8] In this question you will use the matplotlib, which is a Python 2-D plotting library, to plot the stock price (the closing column) and the up periods of Alphabet Inc. (GOOGL). Download the csv file for the stock price from *finance.yahoo.com*. The period is 11/13/2017-11/12/2018 daily. Name the csv file by “GOOGL.csv”.
2. [Weight = 3] Write a Python program that uses matplotlib to plot the data in GOOGL.csv. The expected output is given below. The x-axis starts with day 1.



You could reference to the code below (source: <https://pythonprogramming.net/loading-file-data-matplotlib-tutorial/>).

*import matplotlib.pyplot as plt*

*import csv*

*x = []*

*y = []*

*with open('example.txt','r') as csvfile:*

*plots = csv.reader(csvfile, delimiter=',')*

*for row in plots:*

*x.append(int(row[0]))*

*y.append(int(row[1]))*

*plt.plot(x,y, label='Loaded from file!')*

*plt.xlabel('x')*

*plt.ylabel('y')*

*plt.title('Interesting Graph\nCheck it out')*

*plt.legend()*

*plt.show()*

1. [Weight = 5] Implement a function that accepts a csv file of a stock’s price and plots the stock prices and the up periods on the same graph like the one below. You may refer to <https://matplotlib.org/gallery/api/two_scales.html> for making plots with different scales.

***Function*** *stockUp(priceFile)*

*(The function will plot the prices and the up periods against*

*the days in the same graph with different scales.)*

***Input****: priceFile is the name of a csv file that contains the*

*prices of a stock.*

***Output****: None*

Include the statement below in your .*py* file. GOOGL.csv is the same as the one in part (a).

*stockUp("GOOGL.csv")*

The expected output is given below. The x-axis starts with day 1.

