Exercise Sheet 01

Miguel A. Ibarra-Arellano

April 2019

Programming Exercises

Ex. 1 Srnivasa Ramujan Calculates

The mathematician Srinivasa Ramanujan found an infinite series that can be used to generate a numerical approximation of π:

$$\frac{1}{\pi} = \frac{2\sqrt{2}}{9801} \sum_{k=0}^{\infty} \frac{(4k)!(1103 + 26390k)}{(k!)^4 396^{4k}}$$

Write a function called `estimate_pi` that uses this formula to compute and return an estimate of pi. it should use a `while` loop to compute terms of the summation until the last term is smaller than `1e-15` (which is the python notation for 10^{-15}). You can check the result by comparing it to `math.py`.

Ex. 2 Happy numbers

Happy numbers are defined by the following process: Start with a positive number, replace the number with the sum of the squares of its digits and repeat untill you reach the number 1 or the process enter a loop not involving the number 1. A number that reaches 1 is consider a happy number, all the other numbers ar unhappy.

a. Write a function `is_happy(n)` that checks whether a number is happy or unhappy. It should return `true` if the number is happy and `false` otherwise.

   Hint: It is known that, if a number is unhappy it will enter a loop involving the number 4. Thus you can repeat the process until the number reaches 1 (happy) or 4 (unhappy).

   Hint: You will somehow need to extract the individual digits of a number. Secction 8.7 (pg. 75) of Think Python explains how to iterate over the individual characters of a string. A number can be represented as a string using the `str` function and viceversa a string consisting of digits can be converted back to a integer using the `int` function.

b. Find all happy numbers from 1 to 100.

c. Solve the problem in two different ways using a) while-loops and b) recursion.

d. *Bonus:* Modify the problem by taking the sum of cubes instead of the sum of squares. find all the loops that can occur and all the numbers that are equal to the sum of the cubes of their digits. A number is called cube-happy if its iteration ends in a number that is identical to the sum of the cubes of its digits. Find all the cube-happy numbers from 1 to 100.

Ex. 3 The Birthday Paradox

a. Write a function called `has_duplicates` that takes a list and returns `True` if there is any element that appears more than once. It should not modify the original list.

b. If there are $n = 27$ students in your class, what are the chances that two of you have the same birthday (day/month)? You can estimate this probability by generating random samples of n birthdays and checking for matches. (For simplicity, assume that every tear has 365 days and the probability to be born on any day is the same.) *Hint:* You can generate random birthdays with the `randint` function in the `random` module.

   (a) Estimate the probability on the basis generating 10000 trials of $n = 27$ birthdays and determine the fraction of trials, where at least two persons share a birthday.

   (b) How do your estimates compare to the approximated probability $p_m(n) \approx 1 - c^{-\frac{n^2}{2m}}$ for $m = 365$,$n = 27$ and the exact probability.

$$1 - p_m(n) = 1 \cdot \frac{m-1}{m} \cdot \frac{m-2}{m} \cdot \ldots \cdot \frac{m-n+1}{m}$$

   (c) *Bonus:* Modify your approach to estimate the probability that at least three people share a birthday with another one.

You can read about this problem at:
`https://en.wikipedia.org/wiki/Birthday_problem`

Anagrams

a. Write a program that reads a word list from the file `words.txt` and prints all the sets of words that are anagrams. Limit your output to words having at least 6 anagrams (including itself).
Here is an example of what the output might look like:

```
['deltas', 'desalt', 'lasted', 'salted', 'slated', 'staled']
['retainers', 'ternaries']
['generating', 'greatening']
['resmelts', 'smelters', 'termless']
```

b. Modify the previous program so that it prints the largest set of anagrams first. followed by the second largest set, and so on.

c. Which set of 8 letters contains the most anagrams and what are they? *Hint:* there are seven.