HW01: Happy Number
CSE2050 Fall 2017
Instructor: Wei Wei
TA: Zigeng Wang & Param Bidja

1. Introduction

In this exercise, we will complete one Python function to check whether a given number is a *Happy Number* or not.

2. Objectives

The purpose of this assignment is to give you experience in:

- Using mathematical calculations in Python.
- Refreshing knowledge on function and loops in Python.
- Refreshing knowledge on string operations and set operations.

Note: *Before you start, if you are not familiar with string operations, if/else statement, for loop, while loop or function in Python, you are recommended to review the sample codes we covered in lecture first.*

3. Background

3.1. Happy Numbers

A happy number is a number defined by the following process: Starting with any positive integer, replace the number by the sum of the squares of its digits, and repeat the process until the number either equals 1 (where it will stay), or it loops endlessly in a cycle which does not include 1. Those numbers for which this process ends in 1 are happy numbers, while those that do not end in 1 are unhappy numbers. Please see the following examples.

- 13 is a happy number since

$$1^2 + 3^2 = 1 + 9 = 10,$$

$$1^2 + 0^2 = 1 + 0 = 1$$

- 85 is an unhappy number since

$$8^2 + 5^2 = 64 + 25 = 89,$$

$$8^2 + 9^2 = 64 + 81 = 145,$$

$$1^2 + 4^2 + 5^2 = 1 + 16 + 25 = 42,$$

$$4^2 + 2^2 = 16 + 4 = 20,$$

$$2^2 + 0^2 = 4,$$

$$4^2 = 16,$$

$$1^2 + 6^2 = 1 + 36 = 37,$$

$$3^2 + 7^2 = 9 + 49 = 58,$$

$$5^2 + 8^2 = 25 + 64 = 89, \text{which is a repeated result}$$

In the examples above, we can find that a good way to check whether a number is a happy number is to check whether there is a result equal to 1 or a **_repeated result_** greater than 1 through the iterative process.

4. Assignment

When you are reading this lab assignment, you must have already downloaded the skeleton zip file. In the zip file, you can find a skeleton code, named *happy.py*. All you need to do is to complete the skeleton code based on the instructions and submit it to the Mimir system.

4.1. Happy number checking function

Now, open the skeleton code. You can see one function named *happy(number)*, which is the only function that you need to complete. This function should return True (Boolean value) when the input is a happy number or return False when the input is not a happy number.

In order to check a large number, the type of the input parameter variable should be string instead of int. In other words, this function is to check whether a positive integer represented by a string is a happy number or not.

For example, a typical string input may look like "2345678901234567899999". Note the number represented by this string could be larger than any long integers. You need to write your own code to hand this string. **_Variable type conversion_** and **_set operation_** can be of help.

In the skeleton code, the statements below the function are designed to check the functionality of your code by outputting all the happy numbers up to 1000.

5. Submit your work to Mimir

Submit your code to Mimir after you complete your code. The Mimir will automatically check the output of your code and grade your submission. You can submit your code to Mimir up to **30 times** to refresh your existing score before the submission deadline.

6. Due date

This lab assignment is worth **4 points** in the final grade. It will be due by **11:59pm on Sep 18th 2017**. A penalty of **10% per day** will be deducted from your grade, starting at 12:00am.