Quarter 2 Assignment 1: Programming Contest

Due: **Tuesday Oct 26th**

In this assignment, you will write solutions to three problems from the ACM Programming Contest held at Dixie State College last spring. Each program that you write will take its input from a file called **input.txt**, and write its output to a file called **output.txt**, just as in the contest. Unlike the actual contest, you will have no time limit, and you will work alone (the contest is done in teams of three).

The problem descriptions given below are the exact descriptions from the contest. You should test your solutions well enough that you are confident they are correct before submitting each solution. Each problem includes a sample input file and the output that your program should generate ($ is used to mark where the newline character at the end of the line is). Your program's output must match the sample output perfectly. Not all interesting cases are tested by the sample input—you should make sure that your solution works correctly for all input as described for the problem.

Part1: Number Palindromes

A numeric palindrome is a number that reads the same forwards or backwards. For example, 747, 1221, and 555 are all numeric palindromes. By following a simple process, most numbers can be converted into palindromes.

The process is to take the original number, reverse the digits and add the two resulting numbers. For example, if the original number is 49, the reversed digit number is 94. If we add 49 and 94, the result is 143. This is not a palindrome yet, so we repeat, by reverseing 143 to get 341, and adding. 143 plus 341 gives us 484. Shazam! We have a palindrome.
Some numbers require only one cycle of reverse and add. Others require 10, 20, or even more cycles.

Your program will be required to find the palindromes for a set of numbers. For each number, it will need to calculate the number of cycles and the resulting numeric palindrome. The input file will contain multiple lines. The first line will contain a single number. This is the number of problems in the file. Each subsequent line will contain a number, N. N is the starting number. The program will calculate the palindrome, P, and the number of cycles required to find P. The numbers N and P will obey the constraint: 0<N<P<1,000,000,000. After the first line, each line in the input file will produce a line in the output file with two numbers, the number of cycles to find P, and the value of P, separated by a single space.

input.txt:	**output.txt:**
4$ 1$ 49$ 69$ 86$	0 1$ 2 484$ 4 4884$ 3 1111$

Part 2: Top 3

In this problem you will read in a list of students and their test scores. There will be a maximum of 50 students in the class. The students and their scores will be in random order. You will then decide the top 3 scores in the list and output the results to the output.txt file.
The input file (input.txt) consists of lines of text. the first field is the student's name. The second field is the student's test score. The first and second fields are separated by a space. The input file is terminated by the end of the file.
The output file (output.txt) consists of lines of text. The first number is the rank. The second field is the name of the student. The third is the test score. All fields are separated by a space.

input.txt:	**output.txt:**
bob 23$ jill 78$ bill 100$ george 99$ mary 98$ tim 45$	1 bill 100$ 2 george 99$ 3 mary 98$

Part 3: Check Protect

In this problem you will be working for a bank. When printing checks, the check amounts need to be formatted and padded with an * in order to avoid fraud. Here are the rules:

1. All check amounts need to have exactly 10 characters.
2. All amounts must be preceeded with a $.
3. All numbers must be formatted with 2 decimals on the right of the decimal.
4. All numbers must have a comma separating the thousands from the hundred.
5. All numbers less than one dollar must have a zero on the left of the decimal.
6. All preceding spaces need to be padded with an *.

The input file (input.txt) consists of lines of text. Each line is a number. That number ranges from 0 to 99,999.99.
The output file (output.txt) consists of lines of text. Each line is a formatted number string.
The symbol $ denotes a cr/lf.

```
Example:
12.34          ->     ****$12.34
123.45         ->     ***$123.45
1234.56        ->     *$1,234.56
12345.67       ->     $12,345.67
.1             ->     *****$0.10
.01            ->     *****$0.01
1234.5         ->     *$1,234.50
```

input.txt:	**output.txt:**
12.34$ 123.45$ 1234.56$ 12345.67$ .1$ .01$ 1234.5$	****12.34 ***123.45 *$1,234.56$ $12,345.67$ *****0.10 *****0.01 *$1,234.50$

What to submit

You should submit one Python source file for each problem by emailing Mr. Alvey : alvey@pineview.org