COMP 125, Fall 2022
Homework Assignment 1

Due: 23:59, November 22 (Tuesday), 2022

In this assignment, you will use your programming skills that you acquired on conditional and repetition statements.

This assignment would be better implemented if collections/sequences (for example, lists) were used. But you will learn them later. **Thus, in this homework, you are NOT allowed to use any kind of collections/sequences. Similarly, you CANNOT allowed to use the index [] operator. Otherwise, you will get no points**.

Likewise, you will learn functions later. Thus, **you are NOT allowed to use any functions, except the built-in `int`, `input` and `print` functions. Otherwise, you will get no points**.

This assignment has two parts. As specified below, you will save the solution of each part in a separate py file. You MUST conform to the following naming convention to give a name to each of these py files. Otherwise, you will lose 5 points each time you are using another file name.

- For Part A, the file name should be **`hw1_part_a.py`**
- For Part B, the file name should be **`hw1_part_b.py`**

Please put all these py files in a compressed file (zip, rar, etc) and submit it as a single file via Blackboard.

Please note that the correctness of your program will, of course, affect your grade. However, in addition to this, the following are important to get the full credit from each part.

- You MUST use meaningful names for the variables.
- You MUST write explanatory and clearly understandable comments.
- At the beginning of each file, you MUST write your name, your id, and your section as comments. After these, you MUST provide a very brief description about what the program does as additional comments.

For this assignment, you are allowed to use the codes given in our lecture slides. However, you ARE NOT ALLOWED to use any code from other sources (including the codes given in textbooks, found on the Internet, belonging to your classmates, etc.). **Do not forget that plagiarism and cheating will be heavily punished. Please do the homework yourself.**

This homework will be graded by your TAs, Merve Rabia Barın (mbarin22@ku.edu.tr), Muhammad Awais (mawais21@ku.edu.tr), and Osman Yasal (oyasal22@ku.edu.tr). Thus, you may directly contact them for your homework-related questions.

Before continuing further, please be sure that you carefully read all explanations given above and understood them very well.

PART A: (40 points)

Write a program that takes a positive integer from the user and displays it in the expanded form. The expanded form of a number corresponds to representing this number as the sum of each digit multiplied by its place value (ones, tens, hundreds, thousands, etc.). Please see the examples below for further understanding. You may assume that the input is always valid and a positive integer.

Implement the program in a file called **hw1_part_a.py**.

You are NOT allowed to use collections/sequences (e.g., lists), the index [] operator, and any built-in or user-defined functions except the int, input and print functions. Otherwise, you will get no points.

After implementing the program, test it with different inputs. Below are some example outputs to show how the program should work for few inputs. You may also use these inputs to test your program but they are not complete. It means, we will test your program also with other inputs. Thus, we strongly recommend you to use other test cases.

```
In [1]: runfile('/Users/hw1_part_a.py', wdir='/Users')
Enter a number: 5462
5462 = (5 * 1000) + (4 * 100) + (6 * 10) + (2 * 1)

In [2]: runfile('/Users/hw1_part_a.py', wdir='/Users')
Enter a number: 673
673 = (6 * 100) + (7 * 10) + (3 * 1)

In [3]: runfile('/Users/hw1_part_a.py', wdir='/Users')
Enter a number: 195065
195065 = (1 * 100000) + (9 * 10000) + (5 * 1000) + (6 * 10) + (5 * 1)

In [4]: runfile('/Users/hw1_part_a.py', wdir='/Users')
Enter a number: 3
3 = (3 * 1)

In [6]: runfile('/Users/hw1_part_a.py', wdir='/Users')
Enter a number: 3005
3005 = (3 * 1000) + (5 * 1)

In [7]: runfile('/Users/hw1_part_a.py', wdir='/Users')
Enter a number: 3010
3010 = (3 * 1000) + (1 * 10)

In [8]: runfile('/Users/hw1_part_a.py', wdir='/Users')
Enter a number: 100
100 = (1 * 100)
```

In this part, we will grade the format of the expanded form. You need to strictly obey the format given in the examples above.

To clarify it further, also examine the following example. Please note that this example is to point out some incorrect formats, but it does not cover all possibilities of incorrect formats. Any format rather than the correct one will cause you to lose points.

```
Enter a number: 2005
2005 = (2 * 1000) + (5 * 1)                          correct format
2005 = (2 * 1000) + (0 * 100) + (0 * 10) + (5 * 1)   incorrect format
2005 = 2 * 1000 + 5 * 1                              incorrect format
2005 = (2 * 1000) + (5 * 1) +                        incorrect format
2005 = (1000 * 2) + (1 * 5)                          incorrect format
```

PART B: (60 points)

In this part, you will implement a simple number guessing game. This game first takes a number from the first user (suppose that this is your opponent in the game), and then takes guesses from the second user (suppose that this is you) one by one until the second user makes the correct guess.

The rules are as follows:

- The number and all guesses should be three-digit positive integers. In other words, the number and guesses should be in between 100 and 999. If a user enters an invalid input, the program should continue asking to enter a valid number or a guess. Please see the example run in the first box below.

- When a guess is entered, the program first identifies the digits of the guess found in the correct place, and the digits of the guess found in an incorrect place. Then, it displays the numbers of correct and incorrect places.

  In identifying correct and incorrect places, each digit in the guess can be matched at most one digit in the number or vice versa.

  Please examine the following examples together with the explanations to understand how your program should determine the number of correct and incorrect places (digits with the correct place are typed in green and those with an incorrect place are typed in red). More examples can be found in the second, third, and fourth boxes given below.

  Suppose that the number is 345

  If the guess is 182 --> zero correct, zero incorrect
  No digits matched.

  If the guess is 142 --> one correct, zero incorrect
  4 locates at the tens place of the number, which is the same in the guess.

  If the guess is 143 --> one correct, one incorrect
  4 locates at the tens place of the number, which is the same in the guess.
  3 locates at the hundreds place of the number, but it locates at the ones place of the guess.

  If the guess is 543 --> one correct, two incorrect
  4 locates at the tens place of the number, which is the same in the guess.
  3 locates at the hundreds place of the number, but it locates at the ones place of the guess.
  5 locates at the ones place of the number, but it locates at the hundreds place of the guess.

  If the guess is 441 --> one correct, zero incorrect
  4 locates at the tens place of the number and is matched with 4 also located at the tens place of the guess.
  Since each digit in the number can be matched with only one digit in the guess (and vice versa), the other 4, which is located at the hundreds place of the guess, is not matched with any digit. Thus, it is counted as neither correct not incorrect.

  If the guess is 444 --> one correct, zero incorrect
  See the explanation given for the previous guess.

Suppose that the number is 988

If the guess is 412 --> zero correct, zero incorrect

If the guess is 442 --> zero correct, zero incorrect

If the guess is 128 --> one correct, zero incorrect

8 found at the ones place of the guess is matched with 8 also located at the ones place of the number.

Since each digit in the guess can be matched with only one digit in the number (and vice versa), the other 8, which is located at the tens place of the number, is not matched with any digit. Thus, only one match is found, which is at the correct place.

If the guess is 882 --> one correct, one incorrect

8 found at the tens place of the guess is matched with 8 also located at the tens place of the number. Thus, it is counted as correct.

8 found at the hundreds place of the guess is matched with 8 located at the ones place of the number. Thus, it is found at an incorrect place.

If the guess is 889 --> one correct, two incorrect

8 locates at the tens place of the number, which is the same in the guess.

8 locates at the ones place of the number, but it locates at the hundreds place of the guess.

9 locates at the hundreds place of the number, but it locates at the ones place of the guess.

If the guess is 988 --> three correct, zero incorrect

Implement the program in a file called **hw1_part_b.py**.

You are NOT allowed to use collections/sequences (e.g., lists), the index [] operator, and any built-in or user-defined functions except the int, input and print functions. Otherwise, you will get no points.

Hint 1: You may first find the three digits in the number and store them in separate variables. Likewise, you may find the three digits in each guess (in each iteration) store them in separate variables. Then, you may use these variables in your comparisons.

Hint 2: For each digit in the number, you may use a Boolean flag to indicate whether there has been a correct or incorrect match with a digit in each guess. Likewise, for each digit in each guess, you may use a Boolean flag to indicate whether there has been a correct or incorrect match with a digit in the number.

After implementing the program, test it with different inputs. Below are four example runs to show how the program should work. You may also use them to test your program, but they are not complete. It means we will test your program also with other inputs. Thus, we strongly recommend you to use other test cases.

```
# This is the first example run to show how you need to check the
# validity of the inputs. Both the number and guesses should be
# three-digit positive integers.

In [1]: runfile('/Users/hw1_part_b.py', wdir='/Users')
Enter a number: 60

Invalid number, enter a three-digit positive number: -123

Invalid number, enter a three-digit positive number: 5467

Invalid number, enter a three-digit positive number: 5555
```

```
Invalid number, enter a three-digit positive number: 567

Your guess: -124

Invalid guess, enter a three-digit positive guess: 90

Invalid guess, enter a three-digit positive guess: 2000

Invalid guess, enter a three-digit positive guess: 123
Correct place: 0 digits
Incorrect place: 0 digits

Your guess: 56743

Invalid guess, enter again: -8

Invalid guess, enter again: 10

Invalid guess, enter again: 567
You guessed the number. Congrats!
```

```
# The second example run. Test your program also with other test cases.

In [2]: runfile('/Users/hw1_part_b.py', wdir='/Users')
Enter a number: 843

Your guess: 333
Correct place: 1 digits
Incorrect place: 0 digits

Your guess: 483
Correct place: 1 digits
Incorrect place: 2 digits

Your guess: 123
Correct place: 1 digits
Incorrect place: 0 digits

Your guess: 323
Correct place: 1 digits
Incorrect place: 0 digits

Your guess: 343
Correct place: 2 digits
Incorrect place: 0 digits

Your guess: 483
Correct place: 1 digits
Incorrect place: 2 digits

Your guess: 843
You guessed the number. Congrats!
```

```
# The third example run. Test your program also with other test cases.

In [3]: runfile('/Users/hw1_part_b.py', wdir='/Users')
Enter a number: 227

Your guess: 127
Correct place: 2 digits
Incorrect place: 0 digits

Your guess: 122
Correct place: 1 digits
Incorrect place: 1 digits

Your guess: 982
Correct place: 0 digits
Incorrect place: 1 digits

Your guess: 222
Correct place: 2 digits
Incorrect place: 0 digits

Your guess: 221
Correct place: 2 digits
Incorrect place: 0 digits

Your guess: 227
You guessed the number. Congrats!
```

```
# The fourth example run. Test your program also with other test cases.

In [4]: runfile('/Users/hw1_part_b.py', wdir='/Users')
Enter a number: 888

Your guess: 128
Correct place: 1 digits
Incorrect place: 0 digits

Your guess: 828
Correct place: 2 digits
Incorrect place: 0 digits

Your guess: 120
Correct place: 0 digits
Incorrect place: 0 digits

Your guess: 888
You guessed the number. Congrats!
```