General Instructions

1. This activity consists of multiple short problems. Create one Python script file per problem.
2. Save each Python script according to the filename indicated in the problem statement.
3. **Each** Python script file should contain header information in comments, indicating your full name, your ID number, and the date you created your program.
4. **Each** Python script file should also contain **a comment block for the certification of authorship**, directly following the header information. See code_certification_template.py.
5. Create a folder named according to the convention: **HOA1-Surname-GivenName-IDNumber**. Place all of your Python files for this lab activity in this folder.
6. Archive your folder. On Windows, right-click on the folder and choose **Send To** > **Compressed (zipped) folder**. On a Mac, right-click on the folder and choose **Compress "folder name"**. Ensure that it is named **HOA1-Surname-GivenName-IDNumber.zip** and submit it through the appropriate Moodle submission module.

Note: For this lab, there is **no need** to include a separate Certificate of Authorship document.

IMPORTANT:

- Do not produce any excess output (e.g. *The answer is...*)
- Do not print cues for input (e.g. *Please input a number:*)
- The format of your output must match the output specifications exactly (see **Sample Output** column for examples for each problem)
- You are not allowed to use loops for any of the problems in this lab. Use only the concepts that we have learned in the lessons so far.
- Unless explicitly stated otherwise, assume that the user will always follow the input restrictions (e.g. if input `n` is described as `0 < n < 100`, then the user will always input a value within that range), so there is no need for you to check for those.
- You are expected to solve these problems using the concepts discussed in the associated module. Do not use any constructs from future modules (e.g. loops, lists, etc...)

Problem A : Smooth Operator
Filename : hoa1a.py
Description : A program that performs basic arithmetic
Input : The program accepts two positive integers, a and b, one on each line.
Output : The first line contains the sum of a and b.
The second line contains the value of b subtracted from a.
The third line contains the product of a and b.
The fourth line contains the quotient when a is divided by b. (excluding the remainder)
The fifth line contains the remainder when a is divided by b.

Sample Input #1
```
55
3
```

Sample Output #1
```
58
52
165
18
1
```

Sample Input #2
```
128
8
```

Sample Output #2
```
136
120
1024
16
0
```

Problem B : Label It!
Filename : hoa1b.py
Description : A program that takes two measurements, adds them, and expresses them in the indicated unit of measurement
Input : Input consists of a unit of measurement on one line, followed by two integers a and b,on two separate lines.
The unit will consist of lowercase and uppercase English letters. There will be no spaces in the unit. The unit will be at most 100 characters long.
Output : On a single line, output the sum of a and b, in the following format: `[sum][space][unit]`

Sample Input #1
```
meters
55
3
```

Sample Output #1
```
58 meters
```

Sample Input #2
```
Bytes
128
8
```

Sample Output #2
```
136 Bytes
```

Problem C : Easy Enough?
Filename : hoa1c.py
Description : A program that takes an 8-digit number and outputs the sum of its digits
Input : Input consists of a single positive 8-digit integer.
Output : Output a single integer on one line: the sum of the digits in the input.

Sample Input #1
```
31415926
```

Sample Output #1
```
31
```

Sample Input #2
```
12345678
```

Sample Output #2
```
36
```

Problem D : Echo Echo
Filename : hoa1d.py
Description : A program that prints five lines of text in input
Input : Input consists of five lines.
Output : Output consists of five lines. Each line should be the same as the input line.

Note: You may print the output after each input, or you may print them all after the very last input.

Sample Input #1
```
Hello
In this exercise,
you should read the line
and output the same line
very easy, huh?
```

Sample Output #1
```
Hello
In this exercise,
you should read the line
and output the same line
very easy, huh?
```

Sample Input #2
```
Let's try
another
        set of
      lines
derived from a poem
```

Sample Output #2
```
Let's try
another
        set of
      lines
derived from a poem
```

Reminders:

- Do not produce any excess output. Do not print cues for input.
- The format of your output must match the output specifications exactly.
- Follow file naming conventions.
- Follow submission procedures.
- Always double-check that you have successfully submitted the correct file.