# CS2020
# Introduction to Programming
## Practice Exercise Set 2

## Defining and Using Functions

For each exercise, you either define and/or use a function. Do not use any data structures such as lists in these exercises. It should not take more than 10 minutes to complete any exercise in this set.

### Practice Topic

- Defining functions
- Using functions

### Prerequisite

You need to possess knowledge on

- Basic operations such as assignments and expressions
- Input and output (print)
- Data types int, float, and str.
- Simple selection control using if-else
- Basic looping (Practice Exercise Set 1)

### Example

Define a function total that accepts two numbers and returns their sum. Print out the sum of 15 and 25 using the function total.

```
def total(x, y):
    result = x + y
    return return


print(total(15, 25))
```

1. Input two numbers num1 and num2. Compute and print out their total using the total function defined in the example.

2. Define a function greeting that accepts a person's name (string) and returns a greeting message Hello, <name> where <name> is the value passed to the function. Use this function to print the greeting message to "John", "Jack", and "Jill".
3. Define a function get_total that returns the sum of integers between the two parameters low and high. Assume low is less than or equal to high. Use this function to compute the sum of integers between 10 and 100.
4. Define a function input_positive that prompts the user to enter a positive integer. If the number is positive, return this number. If not, prompt the user again (the input was 0 or negative). Repeat the prompt until a positive integer is entered. Use this function to input two positive numbers and compute the sum of all integers between them using the get_total function. Note that the get_total function requires the first parameter to be the smaller of the two parameters.
5. Define a boolean function is_vowel that accepts a string of single character and returns True if this character is a vowel (either upper or lowercase). The function returns False otherwise.
6. Define a function count_vowels that accepts a string and returns the number of vowels in the passed string. Use is_vowel in implementing the count_vowels function.
7. Define a boolean function is_even that accepts an integer. The function returns True if the integer is even and False, otherwise.
8. Define a boolean function is_divisible that accepts two integers i and j. The function returns True if i is divisible by j (i.e., remainder of i divided by j is 0) and False, otherwise.
9. Define a function find_max that accepts three numbers and returns the maximum of the three passed values. Do not use the built-in function max here.
10. Define a function find_avg that accepts four numbers and returns their average.
11. Define a function find_longer that accepts two string values and returns the longer of the two. You may use the built-in function len here. If they are of the same length, return either one.
12. Define a function find_longer that accepts two string values and returns the longer of the two. You may not use the built-in function len here. If they are of the same length, return either one.
13. Define a function find_longer_alpha that accepts two string values and returns the longer of the two. You may use the built-in function len here. If they are of the same length, return the one that comes before in lexicographical (alphabetical) order.
14. Define a function find_longest that accepts four string values and returns the longest string. If they are all the same length, return any one of the four strings. Use the find_longer function.
15. Define a function more_vowels that accepts two string values and returns the string that has more vowels (both upper and lowercase). If they have the same number of vowels, return either one. Use the count_vowels function.
16. Define a function most_vowels that accepts three string values and returns the string that has the most vowels. Use the more_vowels function.