CompSci 101 Lab 5 — Lists, `append()`, `range()` and `for...in` loops

Topics covered:

- Creating a list
- Processing the elements of a list
- The `append()` method
- `range()` function
- `for … in range()` Loops

Programming Exercises

IMPORTANT: for each of your programs you need to add a docstring at the top of the program. The docstring should contain your name, your username, date and a short description of the program.

Notes

1. The skeleton code for each of the Lab 5 programs is provided. Each program requires you to complete a function. As you complete each function you can test that the function is correct by pasting the whole function (including the header) into CodeRunner3 and pressing the CHECK button.
2. After the description of each program there is a textbox with some example code and beside it a textbox containing the expected output for that code.

Question 1 Program 1

Complete the `print_even_numbers(first_num, last_num)` function which is passed two integer parameters. The function prints all the even numbers between the two parameter numbers (both inclusive) in one line of output. Each number is separated from the next number by a single space (there is a space after the last number).

Notes

- Your code **MUST** use a `for … in range()` loop.
- The first parameter may be an odd number.
- If the second parameter is less than the first parameter then nothing is printed.

```
print_even_numbers(2, 14)
print_even_numbers(5, 16)
```

```
2 4 6 8 10 12 14
6 8 10 12 14 16
```

Question 2 Program 2

Complete the `print_numbers(number1, number1)` function which is passed two integer parameters. The function prints all the numbers between 1 and 99 (both inclusive) which are exactly divisible by **either** of the two parameter numbers. All numbers are printed on the same line and each number is separated from the next number by a single space (there is a space after the last number).

Notes

- Your code **MUST** use a `for … in range()` loop.
- If the second parameter is smaller than the first parameter then nothing is printed.

```
print_numbers(10, 25)
print_numbers(25, 20)
```

```
10 20 25 30 40 50 60 70 75 80 90
20 25 40 50 60 75 80
```

Question 3 Program 3

Complete the `get_list_of_non_negative_evens(numbers_list)` function which is passed a list of integers as a parameter. The function returns a **new list** of all the non-negative even numbers from the parameter list.

Notes

- If the parameter does not contain any non-negative even numbers then the function returns an empty list.
- You **MUST** use the `append()` method to add numbers to the end of the list.

```
print(get_list_of_non_negative_evens([51, -55, 56, 23, 23, 54]))
print(get_list_of_non_negative_evens([46, 26, 24, -44, -20]))
```

```
[56 54]
[46, 26, 24]
```

Question 4 Program 4

Complete the `get_count_same_start_end(numbers_list)` function which is passed a list of integers as a parameter. The function returns the count of all the numbers in the parameter list which have the same starting and ending digit.

Notes

- Single digit numbers are considered as starting and ending with the same digit.
- You can assume there are no negative integers in the parameter list.

```
print(get_count_same_start_end([313, 636, 2042, 40, 447]))
print(get_count_same_start_end([101, 4559, 241, 124, 9249]))
```

```
3
2
```

Question 5 Program 5

Define the `print_longest_word(word_list)` function which is passed a list of strings as a parameter. The function prints the longest word in the parameter list. If there is more than one word with the longest length the function prints the longest word which is closest to the end of the list.

```
print_longest_word(['fish', 'barrel', 'like', 'shooting', 'in', 'a'])
print_longest_word(['cat', 'the', 'the', 'bag', 'let', 'out', 'of'])
print_longest_word(['the', 'the', 'bag', 'let', 'out', 'of', 'cat'])
```

```
shooting
out
cat
```

Question 6 Program 6

Complete the `contains_only_3_digit_numbers(numbers)` function which is passed a list of positive and negative integers as a parameter. The function returns `True` if all the numbers in the parameter list have exactly three digits, otherwise the function returns `False`.

Hint: use the `abs()` function to change each element of the parameter list to a positive number.

```
print(contains_only_3_digit_numbers([117, -241, -171, 112,
                                     317, 290, 77, 394]))
print(contains_only_3_digit_numbers([-491, -375, -65, -348]))
print(contains_only_3_digit_numbers([-716, -948, -636, 595,
                                     179, -708, 867, -173]))
```

```
False
False
True
```