Problem set 3: Functions

1. Write a function to rotate a list of integers in the range [1, *n*] by *k* steps to the right.

   Input:

   ```
   7
   ```

   Output:

   ```
   5, 6, 7, 1, 2, 3, 4
   ```

2. Write a function that takes an unsorted list of integers from 1 to *n*, where one number is missing, and returns the missing number. Note that the value of *n* is the maximum number found in the input sequence, and is not provided separately.

   Input:

   ```
   1 5 2 6 4
   ```

   Output:

   ```
   3
   ```

3. Write a function that returns the n^{th} Fibonacci number.

   Input:

   ```
   7
   ```

   Output:

   ```
   13
   ```

4. Write a function that uses a nested function to returns “yes”/ “no” to indicate if a given integer is found in Fibonacci series.

   Input:

   ```
   13
   ```

   Output:

   ```
   yes
   ```

5. Write a function that returns all elements that are present more than one time in a given list of integers.

   Input:

   ```
   1 2 3 4 5 3 2 3
   ```

   Output:

   ```
   2 3
   ```

6. Write a function to find the longest common prefix string among a list of strings. If there is no common prefix, return an empty string.

   Input:

   ```
   flower flow flight
   ```

   Output:

   ```
   fl
   ```

7. Write a function to return the index of the first non-repeating character in a string and -1 if there is no unique character.

   Input:

   ```
   GCTGCAGCCG
   ```

   Output:

   ```
   2
   ```

8. Write a function that compresses a string by counting the occurrences of consecutive repeated characters. If the compressed string is longer than the original, return the original string.

   Input:

   ```
   aabccccccaaa
   ```

   Output:

   ```
   a2b1c5a3
   ```

9. Write a function to find the length of the longest substring without repeating characters.

   Input:

   ```
   abcabcdbb
   ```

   Output:

   ```
   4
   ```

10. If two consecutive odd numbers are prime, then they are called as twin primes. Write a function to print twin primes less than *n*.

    Input:

    ```
    1000
    ```

    Output:

    ```
    3 and 5
    5 and 7
    …
    881 and 883
    ```