```python
import copy

# Exercise 1 - Smallest Fraction Terms
def exercise1(num,den):
    def gcd(a, b):
        while b:
            a, b = b, a % b
        return a

    common_divisor = gcd(num, den)

    reduced_num = num // common_divisor
    reduced_den = den // common_divisor


    return (reduced_num, reduced_den)
print(exercise1(12, 15))  # Output: (4, 5)
print(exercise1(8, 4))    # Output: (2, 1)


# Exercise 2 - Magical Dates
def exercise2(day,month,year):

    return day * month == year % 100

# Exercise 3 - All Sublists
def exercise3(l):
    sublists = []
    for start in range(len(l)):
        for end in range(start + 1, len(l) + 1):
            sublists.append(l[start:end])
    return sublists + [[]]


# Exercise 4 - English to Pig Latin Translator
def exercise4(word):
    if not word:
        return word

    if word[-1] in ",.?!":
        punctuation = word[-1]
        word = word[:-1]
    else:
        punctuation = ""

    vowels = "AEIOUaeiou"

    if word[0] in vowels:
        return word + "way" + punctuation

    for i in range(len(word)):
        if word[i] in vowels:
            return word[i:] + word[:i].lower() + "ay" + punctuation

    return word + punctuation

# Exercise 5 - Morse Code Encoder
def exercise5(message):
    morse_code_dict = {
        'A': '.-', 'B': '-...', 'C': '-.-.', 'D': '-..', 'E': '.',
        'F': '..-.', 'G': '--.', 'H': '....', 'I': '..', 'J': '.---',
        'K': '-.-', 'L': '.-..', 'M': '--', 'N': '-.', 'O': '---',
        'P': '.--.', 'Q': '--.-', 'R': '.-.', 'S': '...', 'T': '-',
        'U': '..-', 'V': '...-', 'W': '.--', 'X': '-..-', 'Y': '-.--',
        'Z': '--..', '1': '.----', '2': '..---', '3': '...--', '4': '....-',
        '5': '.....', '6': '-....', '7': '--...', '8': '---..', '9': '----.',
        '0': '-----'
    }

    result = []
    for char in message:
        if char.isalnum():
            char_upper = char.upper()
            if char_upper in morse_code_dict:
                result.append(morse_code_dict[char_upper])
    return ' '.join(result)

# Exercise 6 - Spelling Out Numbers
def exercise6(num):
    if num < 0 or num > 999:
        return "Invalid input"

    ones = ["", "one", "two", "three", "four", "five", "six", "seven", "eight", "nine"]
    teens = ["ten", "eleven", "twelve", "thirteen", "fourteen", "fifteen", "sixteen", "seventeen", "eighteen", "nineteen"]
    tens = ["", "", "twenty", "thirty", "forty", "fifty", "sixty", "seventy", "eighty", "ninety"]
    if 0 <= num <= 9:
        return ones[num]
    elif 10 < num < 20:
        return teens[num - 10]
    elif 10 <= num < 100:
        if num % 10 == 0:
            return tens[num // 10]
        else:
            return tens[num // 10] + "-" + ones[num % 10]
    else:
        if num % 100 == 0:
            return "a hundred" if num == 100 else ones[num // 100] + " hundred"
        else:
            return ("a hundred" if num // 100 == 1 else ones[num // 100] + " hundred") + " and " + exercise6(num % 100)


# Exercise 7 - No Functions without Comments
def exercise7(filename):
    function_names = []
    previous_line_comment = False

    with open(filename, 'r') as file:
        for line in file:
            line = line.strip()

            if line.startswith('def '):
                function_name = line.split('(')[0].replace('def ', '').strip()

                if not previous_line_comment:
                    function_names.append(function_name)

                previous_line_comment = False
            else:
                previous_line_comment = line.startswith('#')

    return function_names

# Exercise 8 - Justify any Text
def exercise8(filename,length):
    with open(filename, 'r') as file:
        content = file.read()

    words = content.split()

    current_length = 0
    line = []
    aligned_text = []

    for word in words:
        if current_length + len(word) + len(line) > length:
            aligned_text.append(' '.join(line))
            line = []
            current_length = 0

        line.append(word)
        current_length += len(word)

    if line:
        aligned_text.append(' '.join(line))

    return aligned_text

# Exercise 9 - Knight's Challenge
def exercise9(start,end,moves):
    start_x, start_y = ord(start[0]) - ord('a') + 1, int(start[1])
    end_x, end_y = ord(end[0]) - ord('a') + 1, int(end[1])
    if moves == 0:
        return start_x == end_x and start_y == end_y
    possible_moves = [
        (2, 1), (1, 2),
        (-1, 2), (-2, 1),
        (-2, -1), (-1, -2),
        (1, -2), (2, -1)
    ]
    for move in possible_moves:
        new_x, new_y = start_x + move[0], start_y + move[1]
        if 1 <= new_x <= 8 and 1 <= new_y <= 8 and exercise9((chr(new_x + ord('a') - 1)) + str(new_y), end, moves - 1):
            return True

    return False



# Exercise 10 - War of Species
def exercise10(environment):
    return None
```