# MOCK TEST - 2

## Problem 1:

In [ ]:
```
start = int(input())
end = int(input())
three = set()
five = set()
i = start
while i <= end:
    if i%3 == 0:
        three.add(i)
    if i%5 == 0:
        five.add(i)
    i += 1
o1 = three | five
o2 = three & five
o3 = three - five
o4 = five - three
```

## Problem 2:

In [ ]:
```
eng_dict = {}
def lower(x):
    return x.lower()
result = list(map(lower,words))
for i in 'abcdefghijklmnopqrstuvwxyz':
    for j in result:
        if i == j[0]:
            eng_dict[i] = []
for ele in set(result):
    eng_dict[ele[0]].append(ele)
```

## Problem 3:

In [ ]:
```
def is_perfect(dic):
    res = list(dic.values())
    for ele in res:
        if res.count(ele) > 1:
            return False
    return True
```

## Problem 4:

In [ ]:
```
f = open('file.txt','r')
lines = f.readlines()
sentence_count = len(lines)
word_count = 0
max_count = 0
word_freq = {}
for i in range(len(lines)):
    lines[i] = lines[i].rstrip().split()
    word_count += len(lines[i])
    if max_count < len(lines[i]):
        max_count = len(lines[i])
        max_sentence = ' '.join(lines[i])
    for word in lines[i]:
        if word not in word_freq.keys():
            word_freq[word] = 1
        else:
            word_freq[word] += 1
unique = set(word_freq)
f.close()
```

## Problem 5:

In [ ]:
```
class Book:
    def __init__(self, name, author, pages, genre):
        self.name, self.author, self.pages, self.genre = name, author, pages, genre
    def is_fiction(self):
        return self.genre == 'Fiction'
    def is_nonFiction(self):
        return self.genre == 'Nonfiction'
    def time_to_read(self):
        if self.pages < 100:
            return  '5 days'
        if 100 <= self.pages <= 500:
            return  '20 days'
        if 500 < self.pages :
            return  'infinite'
    def same_author(self, book):
        return self.author == book.author
```