CSE1017 프로그래밍기초(2022) 중간 시험 - 04.27(수) 총 35점

1. [2+2+2=6점] 정수범위의 수 모두 더하기

range(start,stop,step) 정수범위에 있는 수를 모두 더하는 함수 sum_range를 다음과 같이 재귀로 작성하였다.

```
def sum_range(start,stop,step):
    if step > 0:
        if start < stop:
            return start + sum_range(start+step,stop,step)
        else:
            return 0
    else:
        return None
```

step이 양수인 경우만 작동하고, 0이나 음수인 경우에는 None을 리턴 한다.

실행 사례는 다음과 같다.

```
 sum_range(5,13,0) => None
sum_range(13,5,-2) => None
 sum_range(13,5,2) => 0
 sum_range(5,13,1) => 68
 sum_range(5,13,2) => 32
```

위 재귀 함수와 똑같이 작동하도록 (1) 꼬리재귀 함수로 재작성하고, 이어서 (2) while 루프 함수, (3) for 루프 함수로 차례로 재작성하자.

2. [4점] 리스트에서 가장 작은 원소 찾기 (for 루프 활용)

리스트를 인수로 받아서, 가장 작은 원소를 찾아서 리턴 하는 함수 find_smallest를 for 루프를 활용하여 작성하자. 인수가 빈 리스트인 경우, None을 리턴 해야한다. min 내장 함수는 사용할 수 없다.

실행 사례는 다음과 같다.

```
 find_smallest([]) => None
find_smallest([3]) => 3
find_smallest([1,2,3,4,5]) => 1
find_smallest([5,4,3,2,1]) => 1
find_smallest([5,4,3,4,5]) => 3
```

3. [2+2=4점] 리스트에서 지정한 원소 하나 제거 하기

리스트 xs와 원소 x를 인수로 받아서, xs에서 x를 하나 제거하여 리턴 하는 함수 remove_one을 다음과 같이 재귀로 작성하였다.

```
def remove_one(xs,x):
    if xs != []:
        if x == xs[0]:
            return xs[1:]
        else:
            return [xs[0]] + remove_one(xs[1:],x)
    else:
        return []
```

실행 사례는 다음과 같다.

```
 remove_one([],3)) => []
remove_one([4,2,3,4,1],4)) => [2,3,4,1]
remove_one([4,2,3,4,1],1)) => [4,2,3,4]
remove_one([4,2,3,4,1],5)) => [4,2,3,4,1]
```

위 재귀 함수와 똑같이 작동하도록 (1) 꼬리재귀 함수로 재작성하고, 이어서 (2) while 루프 함수로 재작성하자.

4. [2+2+2=6점] 리스트에서 지정한 원소 모두 제거 하기

이번에는 리스트 xs와 원소 x를 인수로 받아서, xs에서 x를 모두 제거하여 리턴 하는 함수 remove_all을 (1) 재귀 함수, (2) 꼬리재귀 함수, (3) while 루프 함수로 각각 차례로 작성하자.

실행 사례는 다음과 같다.

```
 remove_all([],3)) => []
remove_all([4,2,3,4,1],4)) => [2,3,1]
remove_all([4,2,3,4,1],1)) => [4,2,3,4]
remove_all([4,2,3,4,1],5)) => [4,2,3,4,1]
```

5. [6점] 자연수 문자열에서 쉼표(,)가 천 단위로 잘 삽입되어 있는지 확인

다음 재귀 함수는 자연수 문자열 인수에 쉼표(,)가 천 단위로 제대로 삽입되어 있는지 확인하는 함수이다.
문자열은 숫자와 쉼표로만 구성되어 있다고 가정한다. (주어진 뼈대코드 파일의 실행 사례 참조)

```
def check_number_with_comma(s):
    def loop(s):
        (front,comma,rest) = s.partition(",")
        if len(front) == 3:
            if comma == ",":
                return loop(rest)
            else:
                return True
        else:
            return False
    (front,comma,rest) = s.partition(",")
    if len(front) == 3:
        if int(front) < 100:
            return False
        else:
            if comma == ",":
                return loop(rest)
            else:
                return True
    elif len(front) == 2:
        if int(front) < 10:
            return False
        else:
            if comma == ",":
                return loop(rest)
            else:
                return True
    elif len(front) == 1:
        if int(front) == 0:
            return False
        else:
            if comma == ",":
                return loop(rest)
            else:
                return True
    else:
        return False
```

아래 뼈대 코드의 True를 논리식으로 대치하여 위 함수와 동일하게 작동하도록 재작성하자.

```
def check_number_with_comma(s):
    def loop(s):
        (front,comma,rest) = s.partition(",")
        return True
    (front,comma,rest) = s.partition(",")
    return True
```

6. [5+2+2=9점] 자연수에 천 단위로 쉼표(,)를 삽입하기

자연수를 인수로 받아서 천 단위로 쉼표(,)를 삽입한 문자열로 변환하여 리턴하는 함수 `to_number_with_comma`를 작성하자.
실행 사례는 아래와 같다. 음수 인수는 고려하지 않는다.

```
          to_number_with_comma(111) => "111"
         to_number_with_comma(1000) => "1,000"
         to_number_with_comma(1011) => "1,011"
        to_number_with_comma(11001) => "11,001"
    to_number_with_comma(11111111) => "11,111,111"
to_number_with_comma(1111111111) => "1,111,111,111"
```

(1) 먼저 재귀 함수로 작성하고,
(2) 작성한 재귀 함수를 꼬리 재귀로 재작성하고,
(3) 이어서 while 루프 함수로 재작성하자.