문제1

왼손 장갑의 제품 번호가 들어있는 리스트와 오른손 장갑의 제품 번호가 들어있는 리스트가 있습니다.
제품 번호는 1부터 10 사이의 자연수입니다. 제품 번호가 같은 왼손 장갑과 오른손 장갑을 합쳐 장갑 한 쌍을 만들 수 있습니다. 이때, 최대한 많은 쌍의 장갑을 만들면 최대 몇 쌍을 만들 수 있는지 구하려 합니다.
이를 위해 다음과 같이 프로그램 구조를 작성했습니다.

1. 왼손 장갑이 제품 번호별로 몇 개씩 있는지 개수를 셉니다. 2. 오른손 장갑이 제품 번호별로 몇 개씩 있는지 개수를 셉니다. 3. 각 제품 번호별로 최대한 많은 장갑 쌍을 만들면서 개수를 셉니다.

왼손 장갑의 제품 번호가 들어있는 리스트 left_gloves와 오른손 장갑의 제품 번호가 들어있는 리스트 right_gloves가 매개변수로 주어질 때, 최대 몇 개의 장갑 쌍을 만들 수 있는지 return 하도록 solution 함수를 작성하려 합니다.
이때, 위 구조를 참고하여 중복되는 부분은 func_a라는 함수로 작성했습니다. 코드가 올바르게 동작할 수 있도록 빈칸을 알맞게 채워주세요.

매개변수 설명

왼손 장갑의 제품 번호가 들어있는 리스트 left_gloves와 오른손 장갑의 제품 번호가 들어있는 리스트 right_gloves가 solution 함수의 매개변수로 주어집니다.

* left_gloves의 길이는 1 이상 100 이하입니다.
* left_gloves의 원소는 1 이상 10 이하의 자연수입니다.
* right_gloves의 길이는 1 이상 100 이하입니다.
* right_gloves의 원소는 1 이상 10 이하의 자연수입니다.

return 값 설명

왼손과 오른손의 제품 번호가 같은 장갑 끼리 한 쌍을 만들 때, 최대 몇 개의 쌍을 만들 수 있는지 개수를 return 해주세요.

예시

left_gloves	right_gloves	return
[2, 1, 2, 2, 4]	[1, 2, 2, 4, 4, 7]	4

예시 설명

예시 #1

왼손 장갑: 1번 장갑 1개, 2번 장갑 3개, 4번 장갑 1개가 있습니다.
오른손 장갑 : 1번 장갑 1개, 2번 장갑 2개, 4번 장갑 2개, 7번 장갑 1개가 있습니다.

따라서 1번 장갑 한 쌍, 2번 장갑 두 쌍, 4번 장갑 한 쌍을 만들면 최대 4개의 장갑 쌍을 만들 수 있습니다.

```
max_product_number = 10

def func_a(gloves):
    counter = [0 for _ in range(max_product_number + 1)]
    for x in gloves:
        @@@
    return counter

def solution(left_gloves, right_gloves):
    left_counter = func_a(left_gloves)
    right_counter = func_a(right_gloves)

    total = 0

    for i in range(1, max_product_number + 1):
        total += min(left_counter[i], right_counter[i])
    return total

# 아래는 테스트케이스 출력을 해보기 위한 코드입니다.
left_gloves = [2, 1, 2, 2, 4]
right_gloves = [1, 2, 2, 4, 4, 7]
ret = solution(left_gloves, right_gloves)

# [실행] 버튼을 누르면 출력 값을 볼 수 있습니다.
print("solution 함수의 반환 값은", ret, "입니다.")
```

문제2

자연수가 들어있는 리스트에 3의 배수와 5의 배수 중 어떤 수가 더 많은지 알아보려 합니다.
이를 위해 다음과 같이 프로그램 구조를 작성했습니다.

1. 3의 배수의 개수를 셉니다.
2. 5의 배수의 개수를 셉니다.
3. 3의 배수와 5의 배수의 개수를 비교 후 다음을 수행합니다.
   3-1. 만약 3의 배수가 더 많다면 "three"를 return 합니다.
   3-2. 만약 5의 배수가 더 많다면 "five"를 return 합니다.
   3-3. 만약 3의 배수와 5의 배수의 개수가 같다면 "same"을 return 합니다.

자연수가 들어있는 리스트 arr가 매개변수로 주어질 때, 리스트에 3의 배수의 개수가 더 많다면 "three"를, 5의 배수의 개수가 더 많다면 "five"를, 3의 배수와 5의 배수의 개수가 같다면 "same"을 return 하도록 solution 함수를 작성하려 합니다. 위 구조를 참고하여 코드가 올바르게 동작할 수 있도록 빈칸에 주어진 func_a, func_b, func_c 함수와 매개변수를 알맞게 채워주세요.

매개변수 설명

자연수가 들어있는 리스트 arr가 solution 함수의 매개변수로 주어집니다.
* arr의 길이는 1 이상 100 이하입니다.
* arr에 들어있는 숫자는 1 이상 1,000 이하의 자연수입니다.

return 값 설명

리스트에 3의 배수의 개수가 더 많다면 "three"를, 5의 배수의 개수가 더 많다면 "five"를, 3의 배수와 5의 배수의 개수가 같다면 "same"을 return 해주세요.

예시

arr	return
[2, 3, 6, 9, 12, 15, 10, 20, 22, 25]	"three"

예시 설명

* 3의 배수 : 3, 6, 9, 12, 15
* 5의 배수 : 10, 15, 20, 25

3의 배수는 5개, 5의 배수는 4개이므로 3의 배수가 더 많습니다. 따라서 "three"를 return 합니다.

```
def func_a(arr):
    count = 0
    for n in arr:
        if n % 5 == 0:
            count += 1
    return count

def func_b(three, five):
    if three > five:
        return "three"
    elif three < five:
        return "five"
    else:
        return "same"

def func_c(arr):
    count = 0
    for n in arr:
        if n % 3 == 0:
            count += 1
    return count

def solution(arr):
    count_three = func_@@@(@@@)
    count_five = func_@@@(@@@)
    answer = func_@@@(@@@)
    return answer

#아래는 테스트케이스 출력을 해보기 위한 코드입니다.
arr = [2, 3, 6, 9, 12, 15, 10, 20, 22, 25]
ret = solution(arr)

#[실행] 버튼을 누르면 출력 값을 볼 수 있습니다.
print("solution 함수의 반환 값은", ret, "입니다.")
```

문제3

서로 다른 두 자연수 N과 M이 매개변수로 주어질 때, N부터 M까지의 자연수 중에서 짝수들의 제곱의 합을 return 하도록 solution 함수를 완성해주세요.

매개변수 설명

두 자연수 N과 M이 solution 함수의 매개변수로 주어집니다.
* N과 M은 1 이상 1,000 이하의 자연수이며, N < M을 항상 만족합니다.

return 값 설명

N부터 M까지의 수 중에서 짝수인 수의 제곱의 합을 return 해주세요.

예시

N	M	return
4	7	52

예시 설명

4부터 7까지의 자연수 중에서 짝수는 4와 6입니다.

* 4^2 + 6^2 = 16 + 36 = 52

따라서 52를 return 하면 됩니다.

```
#다음과 같이 import를 사용할 수 있습니다.
#import math

def solution(N, M):
    #여기에 코드를 작성해주세요.
    answer = 0
    return answer

#아래는 테스트케이스 출력을 해보기 위한 코드입니다.
N = 4
M = 7
ret = solution(N, M)

#[실행] 버튼을 누르면 출력 값을 볼 수 있습니다.
print("solution 함수의 반환 값은", ret, "입니다.")
```

문제4

단어들이 들어있는 리스트에서 길이가 5 이상인 단어를 리스트에 들어있는 순서대로 이어 붙이려 합니다.

예를 들어 리스트가 다음과 같은 경우

["my", "favorite", "color", "is", "violet"]

"favoritecolorviolet"을 만들면 됩니다.

단어들이 들어있는 리스트 words가 solution 함수의 매개변수로 주어질 때, 길이가 5 이상인 단어를 순서대로 이어 붙인 문자열을 return 하도록 solution 함수를 완성해주세요.

매개변수 설명

단어들이 들어있는 리스트 words가 solution 함수의 매개변수로 주어집니다.

* words의 길이는 1 이상 100 이하입니다.
* words에 들어있는 각 단어의 길이는 1 이상 10 이하이며, 알파벳 소문자로만 이루어져 있습니다.

return 값 설명

길이가 5 이상인 단어를 순서대로 이어 붙여 return 해주세요.

* 만약 return 할 문자열이 빈 문자열이면 "empty"를 return 해주세요.

예시

words	return
["my", "favorite", "color", "is", "violet"]	"favoritecolorviolet"
["yes", "i", "am"]	"empty"

예시 설명

예시 #1

길이가 5 이상인 단어는 "favorite", "color", "violet"입니다.
이를 리스트에 들어있는 순서대로 이어 붙이면 "favoritecolorviolet"이 됩니다.

예시 #2

길이가 5 이상인 단어가 없으므로 "empty"를 return 하면 됩니다.

```
#다음과 같이 import를 사용할 수 있습니다.
#import math

def solution(words):
    answer = ''

    #여기에 코드를 작성해주세요.

    return answer

#아래는 테스트케이스 출력을 해보기 위한 코드입니다.
words1 = ["my", "favorite", "color", "is", "violet"]
ret1 = solution(words1);

#[실행] 버튼을 누르면 출력 값을 볼 수 있습니다.
print("solution 함수의 반환 값은 ", ret1, " 입니다.")

words2 = ["yes", "i", "am"]
ret2 = solution(words2);

#[실행] 버튼을 누르면 출력 값을 볼 수 있습니다.
print("solution 함수의 반환 값은 ", ret2, " 입니다.")
```

문제5

게임 캐릭터가 몬스터와 1:1 전투를 하려 합니다. 몬스터는 처음에 일정 수치의 체력(HP)을 가지고 있습니다.
캐릭터가 전투에서 이기려면 몬스터를 공격해 몬스터의 체력을 0 이하로 만들어야 합니다.

캐릭터는 공격 마법만 사용하며, 공격 마법은 항상 같은 데미지를 입힙니다.
몬스터는 힐링 마법만을 사용하며, 힐링 마법은 항상 같은 수치로 체력을 회복합니다.

둘은 항상 번갈아 가며 마법을 사용하고, 처음에는 항상 캐릭터가 먼저 공격을 시작합니다.

캐릭터의 공격력 attack과 몬스터가 자신의 차례에 회복하는 체력 recovery, 몬스터의 초기 체력 hp가 매개변수로 주어질 때, 몬스터를 잡기 위해서 최소 몇 번 공격해야 하는지 return 하도록 solution 함수를 작성하려 합니다. 빈칸을 채워 전체 코드를 완성해주세요.

매개변수 설명

캐릭터의 공격력 attack과 몬스터가 자신의 차례에 회복하는 체력 recovery, 몬스터의 초기 체력 hp가 solution 함수의 매개변수로 주어집니다.

* attack은 1 이상 100 이하의 자연수입니다.
* recovery는 1 이상 100 이하의 자연수입니다.
* 캐릭터의 공격력은 항상 몬스터의 회복력보다 큽니다(recovery < attack).
* hp는 200 이상 1,000 이하의 자연수입니다.

return 값 설명

몬스터를 잡기 위해서 최소 몇 번 공격해야 하는지 return 해주세요.

예시

attack	recovery	hp	return
30	10	60	3

예시 설명

몬스터의 체력 변화는 아래와 같습니다.

차례	hp 변화	남은 hp
캐릭터	없음	60
몬스터	공격 -30	30
캐릭터	회복 +10	40
몬스터	공격 -30	10
캐릭터	회복 +10	20
몬스터	공격 -30	-10

따라서 최소 3번 공격해야 몬스터를 잡을 수 있습니다.

```
def solution(attack, recovery, hp):
    count = 0

    while(True):
        count += @@@
        hp -= @@@
        if hp <= 0:
            @@@
        hp += @@@
    return count

#아래는 테스트케이스 출력을 해보기 위한 코드입니다.
attack = 30
recovery = 10
hp = 60
ret = solution(attack, recovery, hp)

#[실행] 버튼을 누르면 출력 값을 볼 수 있습니다.
print("solution 함수의 반환 값은", ret, "입니다.")
```

문제6

하루 동안 엘리베이터가 멈춘 층이 순서대로 들어있는 리스트가 있습니다. 이때, 엘리베이터의 총 이동거리를 구하려 합니다. 단, 층과 층 사이의 거리는 1입니다.

예를 들어 리스트에 [1, 2, 5, 4, 2]가 들어있다면, 엘리베이터가 이동한 거리는 7입니다.

하루 동안 엘리베이터가 멈춰 선 층이 순서대로 들어있는 리스트 floors가 매개변수로 주어질 때, 엘리베이터의 총 이동 거리를 return 하도록 solution 함수를 작성하려 합니다. 빈칸을 채워 전체 코드를 완성해주세요.

매개변수 설명

하루 동안 엘리베이터가 멈춘 층이 순서대로 들어있는 리스트 floors가 solution 함수의 매개변수로 주어집니다.

* floors의 길이는 2 이상 100 이하입니다.
* floors의 원소는 1 이상 100 이하의 자연수이며, 인접한 두 원소의 값이 같은 경우는 주어지지 않습니다.
* floors의 첫 번째 원소는 엘리베이터의 처음 위치를 나타냅니다.

return 값 설명

엘리베이터의 총 이동 거리를 return 해주세요.

예시

floors	return
[1, 2, 5, 4, 2]	7

예시 설명

엘리베이터는 처음에 1층에 있으며, 다음 순서대로 움직였습니다.

* 1층 - 2층 - 5층 - 4층 - 2층

층과 층사이의 거리는 1이므로, 엘리베이터가 이동한 거리는 다음과 같습니다.

* 1층 - 2층 (이동 거리 : 1)
* 2층 - 5층 (이동 거리 : 3)
* 5층 - 4층 (이동 거리 : 1)
* 4층 - 2층 (이동 거리 : 2)

따라서 총 이동 거리는 7입니다.

```
def solution(floors):
    dist = 0
    length = len(floors)

    for i in range(@@@):
        if @@@:
            dist += floors[i] - floors[i-1]
        else:
            dist += floors[i-1] - floors[i]
    return dist

#아래는 테스트케이스 출력을 해보기 위한 코드입니다.
floors = [1, 2, 5, 4, 2]
ret = solution(floors)

#[실행] 버튼을 누르면 출력 값을 볼 수 있습니다.
print("solution 함수의 반환 값은", ret, "입니다.")
```

문제7

화씨온도(℉)를 섭씨온도(℃)로, 섭씨온도(℃)를 화씨온도(℉)로 바꾸려고 합니다. 두 온도 사이의 환산 공식은 다음과 같습니다.

환산공식

* 화씨온도(℉)에서 섭씨온도(℃)로 환산 : (화씨온도 - 32) ÷ 1.8 = 섭씨온도 * 섭씨온도(℃)에서 화씨온도(℉)로 환산 : (섭씨온도 x 1.8) + 32 = 화씨온도

현재 온도 value와 현재 단위 unit이 매개변수로 주어질 때, 환산한 온도의 정수 부분을 return 하도록 solution 함수를 작성했습니다. 그러나, 코드 일부분이 잘못되어있기 때문에, 몇몇 입력에 대해서는 올바르게 동작하지 않습니다. 주어진 코드에서 **한 줄**만 변경해서 모든 입력에 대해 올바르게 동작하도록 수정하세요.

매개변수 설명

현재 온도 value와 현재 단위 unit이 solution 함수의 매개변수로 주어집니다.

* unit은 화씨온도 "F"와 섭씨온도 "C" 둘 중 하나로 주어집니다.
  * unit이 "F"인 경우 value는 화씨온도(℉)를 나타냅니다.
  * unit이 "C"인 경우 value는 섭씨온도(℃)를 나타냅니다.
* value는 -460 이상 1,000 이하의 정수입니다.

return 값 설명

환산한 온도의 정수 부분을 return 해주세요.

* unit이 "F"인 경우에는 화씨온도(℉)에서 섭씨온도(℃)로 환산해주세요.
* unit이 "C"인 경우에는 섭씨온도(℃)에서 화씨온도(℉)로 환산해주세요.

예시

value	unit	return
527	"C"	980

예시 설명

unit이 "C" 이므로 주어진 value는 527℃를 나타냅니다. 이를 화씨온도(℉)로 환산하면 다음과 같습니다.
(섭씨온도 x 1.8) + 32 = (527 x 1.8) + 32 = 980.6

따라서 환산 결과는 980.6℉이며, 정수 부분만 return 하면 되므로 980을 return 하면 됩니다.

```
def solution(value, unit):
    converted = 0

    if unit == "C":
        value = value * 1.8 + 32

    if unit == "F":
        value = value - 32 / 1.8

    converted = int(value)

    return converted

# 아래는 테스트케이스 출력을 해보기 위한 코드입니다.
# 아래 코드는 잘못된 부분이 없으니, solution함수만 수정하세요.
value = 527
unit = "C"
ret = solution(value, unit)


#[실행] 버튼을 누르면 출력 값을 볼 수 있습니다.
print("solution 함수의 반환 값은", ret, "입니다.")
```

문제8

자연수의 각 자릿수 중에서 소수는 몇 개인지 구하려 합니다. 즉, 자연수를 각 자릿수별로 나누었을 때, 2, 3, 5, 7이 몇 개 있는지 구하려 합니다.

예를 들어, 자연수가 29022531일 때, 각 자릿수 중 소수의 위치는 다음과 같습니다.

* `2`90`2``2``5``3`1

따라서 소수의 개수는 총 5개입니다.

자연수 number가 매개변수로 주어질 때, number의 각 자릿수 중 소수는 몇 개인지 return 하도록 solution 함수를 작성했습니다. 그러나, 코드 일부분이 잘못되어있기 때문에, 몇몇 입력에 대해서는 올바르게 동작하지 않습니다. 주어진 코드에서 **한 줄**만 변경해서 모든 입력에 대해 올바르게 동작하도록 수정하세요.

매개변수 설명

자연수 number가 solution 함수의 매개변수로 주어집니다.

* number는 1 이상 1,000,000,000 이하의 자연수입니다.

return 값 설명

number의 각 자릿수 중 소수는 몇 개인지 return 해주세요.

예시

number	return
29022531	5

```
def solution(number):
    count = 0

    while number >= 0:
        n = number % 10
        if n == 2 or n == 3 or n == 5 or n == 7:
            count += 1

        number //= 10

    return count

# 아래는 테스트케이스 출력을 해보기 위한 코드입니다.
# 아래 코드는 잘못된 부분이 없으니, solution함수만 수정하세요.
number = 29022531
ret = solution(number)


#[실행] 버튼을 누르면 출력 값을 볼 수 있습니다.
print("solution 함수의 반환 값은", ret, "입니다.")
```

문제9

N명의 후보에 대해 투표한 결과가 들어있는 리스트가 있습니다.
예를 들어 5명의 후보에 대해 투표를 진행한 결과가 [2, 5, 3, 4, 1, 5, 1, 5, 5, 3]이라면
순서대로 [2번, 5번, 3번, 4번, 1번, 5번, 1번, 5번, 5번, 3번] 후보에 투표했음을 나타냅니다.
이때, 정확히 K 표를 받은 후보는 총 몇 명인지 구하려 합니다.

예를 들어 K = 2일 때, 위 투표 결과에서 정확히 2표를 받은 후보는 1번, 3번 후보로, 총 2명입니다.

투표 결과가 들어있는 리스트 votes, 후보의 수 N, 표의 개수 K가 매개변수로 주어질 때, K 표를 받은 후보는 몇 명인지 return 하도록 solution 함수를 작성했습니다. 그러나, 코드 일부분이 잘못되어있기 때문에, 몇몇 입력에 대해서는 올바르게 동작하지 않습니다. 주어진 코드에서 **한 줄**만 변경해서 모든 입력에 대해 올바르게 동작하도록 수정하세요.

매개변수 설명

투표 결과가 들어있는 리스트 votes, 후보의 수 N, 표의 개수 K가 solution 함수의 매개변수로 주어집니다.

* votes의 길이는 10 이상, 100 이하입니다.
* votes의 원소는 1 이상, 전체 후보의 수 N 이하의 자연수입니다.
* N은 3 이상 10 이하의 자연수입니다.
* K는 0 이상 100 이하의 정수입니다.

return 값 설명

K 표를 받은 후보는 몇 명인지 return 해주세요.

예시

votes	N	K	return
[2, 5, 3, 4, 1, 5, 1, 5, 5, 3]	5	2	2

예시 설명

각 후보가 받은 표는 다음과 같습니다.

* 1번 후보 : 2표
* 2번 후보 : 1표
* 3번 후보 : 2표
* 4번 후보 : 1표
* 5번 후보 : 4표

따라서 2표를 받은 후보는 1번, 3번 후보로 총 2명입니다.

```
def solution(votes, N, K):
    counter = [0 for _ in range(N + 1)]

    for x in votes:
        counter[x] += 1

    answer = -1

    for c in counter:
        if c == K:
            answer += 1

    return answer

# 아래는 테스트케이스 출력을 해보기 위한 코드입니다.
# 아래 코드는 잘못된 부분이 없으니, solution함수만 수정하세요.
votes = [2, 5, 3, 4, 1, 5, 1, 5, 5, 3]
N = 5
K = 2
ret = solution(votes, N, K)


#[실행] 버튼을 누르면 출력 값을 볼 수 있습니다.
print("solution 함수의 반환 값은", ret, "입니다.")
```

문제10

A 백화점에서는 고객의 구매금액에 따라 다음과 같이 상품권을 지급합니다.

구매금액	상품권
100만 원 이상 구매	5만 원 상품권
60만 원 이상 구매	3만 원 상품권
40만 원 이상 구매	2만 원 상품권
20만 원 이상 구매	1만 원 상품권

이때, 상품권은 지급 가능한 가장 큰 금액으로 한 장만 지급합니다. 예를 들어 고객이 65만 원을 구매했다면, 3만 원 상품권 한 장만 지급합니다.

고객들의 구매금액이 들어있는 리스트 purchase가 주어질 때, 고객들에게 지급해야 하는 상품권 총액을 return 하도록 solution 함수를 작성했습니다. 그러나, 코드 일부분이 잘못되어있기 때문에, 코드가 올바르게 동작하지 않습니다. 주어진 코드에서 한 줄만 변경해서 모든 입력에 대해 올바르게 동작하도록 수정해주세요.

매개변수 설명

고객들의 구매금액이 들어있는 리스트 purchase가 solution 함수의 매개변수로 주어집니다.
* purchase의 길이는 1 이상 100 이하입니다.
* purchase의 원소는 10 이상 1,500,000 이하의 자연수이며, 10원 단위로 주어집니다.

return 값 설명

고객들에게 지급해야 하는 상품권 총액을 return 해주세요.

예시

purchase	return
[150000, 210000, 399990, 990000, 1000000]	100000

예시 설명

* 210,000원, 399,990원을 구매한 고객에게 1만 원 상품권을 지급해야 합니다.
* 990,000원을 구매한 고객에게 3만 원 상품권을 지급해야 합니다.
* 1,000,000원을 구매한 고객에게 5만 원 상품권을 지급해야 합니다.

따라서 지급해야 하는 상품권은 1만 원 상품권 2장, 3만 원 상품권 1장, 5만 원 상품권 1장으로, 총액은 10만 원입니다.

```
def solution(purchase):
    total = 0

    for p in purchase:
        if p >= 1000000:
            total += 50000
        elif p >= 600000:
            total += 30000
        elif p >= 400000:
            total += 20000
        else:
            total += 10000

    return total

# 아래는 테스트케이스 출력을 해보기 위한 코드입니다.
# 아래 코드는 잘못된 부분이 없으니, solution 함수만 수정하세요.
purchase = [150000, 210000, 399990, 990000, 1000000]
ret = solution(purchase)


#[실행] 버튼을 누르면 출력 값을 볼 수 있습니다.
print("solution 함수의 반환 값은", ret, "입니다.")
```