

COS PRO Level 2 - 10 Câu Hỏi

PHẦN 1: ĐIỀN VÀO CHỖ TRỐNG (Câu 1-5)

Câu 1: Tính tổng từ n đến m

Mô tả bài toán

Đang muốn tính tổng các số tự nhiên từ n đến m.

Để làm điều đó, đã xây dựng cấu trúc chương trình đơn giản theo 3 bước như sau.

1. Tính tổng từ 1 đến m.
2. Tính tổng từ 1 đến n-1.
3. Lấy giá trị tính được ở bước 1 trừ đi giá trị tính được ở bước 2.

Khi hai số tự nhiên n và m được cho dưới dạng tham số, đã viết hàm solution để trả về tổng từ n đến m.

Lúc này, tham khảo cấu trúc trên, phần bị trùng lặp được viết thành hàm có tên func_a.

Hãy điền vào chỗ trống một cách thích hợp để code có thể hoạt động đúng.

Mô tả tham số

Hai số tự nhiên n và m được truyền vào dưới dạng tham số của hàm solution.

n, m là các số tự nhiên từ 1 đến 10.000, và luôn thỏa mãn $n \leq m$.

Mô tả giá trị trả về

Hàm solution trả về tổng các số tự nhiên từ n đến m.

Ví dụ

n	m	result
5	10	45
6	6	6

Giải thích ví dụ

Ví dụ #1: Tổng các số tự nhiên từ 5 đến 10 là 45.

Ví dụ #2: Tổng các số tự nhiên từ 6 đến 6 là 6.

Code

```
def func_a(k):  
    sum = 0  
    for i in range(@@@):  
        sum += @@@  
    return sum  
  
def solution(n, m):  
    sum_to_m = func_a(m)  
    sum_to_n = func_a(n-1)  
    answer = sum_to_m - sum_to_n  
    return answer
```

Câu 2: Tổng điểm trừ điểm cao nhất và thấp nhất

Mô tả bài toán

Khi cho một mảng chứa điểm từng môn học của một học sinh, muốn tính tổng điểm của các môn **ngoại trừ** môn có điểm cao nhất và môn có điểm thấp nhất.

Để làm điều đó, đã xây dựng cấu trúc chương trình theo 4 bước như sau.

1. Tính tổng điểm của tất cả các môn.
2. Tìm điểm cao nhất.
3. Tìm điểm thấp nhất.
4. Trả về giá trị (tổng điểm tất cả các môn) - (điểm cao nhất) - (điểm thấp nhất).

Khi mảng điểm từng môn học của học sinh `scores` được truyền vào làm tham số, muốn viết hàm `solution` để trả về tổng điểm các môn, ngoại trừ điểm cao nhất và điểm thấp nhất trong điểm từng môn của học sinh.

Tham khảo cấu trúc trên, hãy điền hàm `func_a`, `func_b`, `func_c` được cho sẵn vào chỗ trống một cách phù hợp để mã nguồn có thể hoạt động đúng.

Mô tả tham số

Mảng `scores` chứa điểm từng môn học của học sinh được truyền làm tham số của hàm `solution`.

Độ dài của `scores` từ 3 đến 10.

Điểm của học sinh là số nguyên từ 0 đến 100, và không có môn nào trùng điểm.

Mô tả giá trị trả về

Hàm `solution` trả về tổng điểm các môn, ngoại trừ điểm cao nhất và điểm thấp nhất trong điểm từng môn của học sinh.

Ví dụ

scores	result
[50, 35, 78, 91, 85]	213

Giải thích ví dụ

Ví dụ #1

Điểm cao nhất: 91 điểm

Điểm thấp nhất: 35 điểm

Tổng điểm của các môn, ngoại trừ điểm cao nhất và điểm thấp nhất là $50 + 78 + 85 = 213$ điểm.

Code

```
def func_a(s):
    ret = 0
    for i in s:
        if i > ret:
            ret = i
    return ret

def func_b(s):
    ret = 0
    for i in s:
        ret += i
    return ret

def func_c(s):
    ret = 101
    for i in s:
        if i < ret:
            ret = i
    return ret

def solution(scores):
    sum = func_a(scores)
    max_score = func_b(scores)
    min_score = func_c(scores)
    return sum - max_score - min_score
```

Câu 3: Hiệu giữa hành khách nhiều nhất và thứ hai

Mô tả bài toán

Tại sân bay XX, người ta đã khảo sát số lượng hành khách đến sân bay mỗi ngày trong N ngày.

Lúc này, muốn tính hiệu giữa ngày có số lượng hành khách nhiều nhất và ngày có số lượng

hành khách nhiều thứ hai.

Giả sử rằng không có ngày nào có cùng số lượng hành khách.

Để làm điều đó, đã xây dựng cấu trúc chương trình đơn giản theo 4 bước sau.

1. Tìm số lượng hành khách nhiều nhất trong mảng được cho làm đầu vào.
2. Tạo một mảng mới chỉ gồm các giá trị còn lại sau khi loại bỏ giá trị tìm được ở bước 1.
3. Tìm số lượng hành khách lớn nhất trong mảng mới tạo ra ở bước 2.
4. Tính hiệu giữa giá trị tìm được ở bước 1 và giá trị tìm được ở bước 3.

Khi mảng visitor chứa số lượng hành khách trong N ngày được truyền làm tham số, muốn viết hàm solution để trả về hiệu giữa số lượng hành khách nhiều nhất và số lượng hành khách nhiều thứ hai.

Tham khảo cấu trúc trên, hãy điền các hàm func_a, func_b, func_c được cho sẵn vào chỗ trống sao cho mã nguồn hoạt động đúng.

Mô tả tham số

Mảng visitor chứa số lượng hành khách trong N ngày được truyền làm tham số cho hàm solution.

visitor là mảng chứa số lượng hành khách đến sân bay trong N ngày.

Độ dài của visitor từ 1 đến 100, và mỗi phần tử là số tự nhiên từ 1 đến 200.000.

Trong visitor không có số nào bị trùng lặp.

Mô tả giá trị trả về

Hàm solution trả về hiệu giữa số lượng hành khách nhiều nhất và số lượng hành khách nhiều thứ hai.

Ví dụ

visitor	result
[4, 7, 2, 9, 3]	2

Giải thích ví dụ

Ví dụ #1

Số lượng hành khách đến sân bay trong 5 ngày, theo thứ tự là [4 người, 7 người, 2 người, 9 người, 3 người].

Ngày có số lượng hành khách nhiều nhất là ngày thứ 4 với tổng 9 người.

Ngày có số lượng hành khách nhiều thứ hai là ngày thứ 2 với tổng 7 người.

Do đó, hiệu giữa số lượng hành khách nhiều nhất và số lượng hành khách nhiều thứ hai là 2.

Code

```

def func_a(arr, n):
    ret = []
    for x in arr:
        if x != n:
            ret.append(x)
    return ret

def func_b(a, b):
    if a >= b:
        return a - b
    else:
        return b - a

def func_c(arr):
    ret = -1
    for x in arr:
        if ret < x:
            ret = x
    return ret

def solution(visitor):
    max_first = func_@@@(@@@)
    visitor_removed = func_@@@(@@@)
    max_second = func_@@@(@@@)
    answer = func_@@@(@@@)
    return answer

```

Câu 4: Tính học điểm theo điểm số

Mô tả bài toán

Tại trường XX, người ta cho điểm học sinh theo thang điểm như sau.

85 ~ 100 : học điểm A

70 ~ 84 : học điểm B

55 ~ 69 : học điểm C

40 ~ 54 : học điểm D

0 ~ 39 : học điểm F

Khi mảng scores chứa điểm số của học sinh được truyền làm tham số, muốn viết hàm solution để trả về số lượng học sinh nhận học điểm A, B, C, D, F theo đúng thứ tự trong một mảng.

Hãy điền vào chỗ trống để hoàn thiện toàn bộ mã nguồn.

Mô tả tham số

Mảng scores chứa điểm số của học sinh được truyền làm tham số cho hàm solution.

Độ dài của scores từ 1 đến 200.

Điểm số của học sinh là số nguyên từ 0 đến 100.

Mô tả giá trị trả về

Hàm solution trả về một mảng chứa lần lượt [số học sinh nhận học điểm A, số học sinh nhận học điểm B, số học sinh nhận học điểm C, số học sinh nhận học điểm D, số học sinh nhận học điểm F].

Ví dụ

scores	result
[86, 72, 98, 60, 45]	[2, 1, 1, 1, 0]

Giải thích ví dụ

Ví dụ #1

Điểm số của các học sinh được cho sẽ nhận học điểm như sau.

A điểm : [86, 98] 2 học sinh

B điểm : [72] 1 học sinh

C điểm : [60] 1 học sinh

D điểm : [45] 1 học sinh

F điểm : 0 học sinh

Vì vậy, cần trả về [2, 1, 1, 1, 0].

Code

```
def solution(scores):
    grade_counter = [0 for i in range(5)]
    for x in scores:
        if '@@':
            grade_counter[0] += 1
        elif '@@':
            grade_counter[1] += 1
        elif '@@':
            grade_counter[2] += 1
        elif '@@':
            grade_counter[3] += 1
        else:
            grade_counter[4] += 1
    return grade_counter
```

Câu 5: Đếm số lần nhảy của ếch qua bệ đá

Mô tả bài toán

Một con ếch đang cố gắng băng qua những hòn đá bước (bệ đá) được đặt thẳng hàng với khoảng cách đều nhau.



Trên mỗi hòn đá bước có ghi một số tự nhiên, và con ếch phải nhảy về phía trước đúng bằng số được ghi trên hòn đá mà nó đang đứng. Con ếch hiện đang ngồi trên hòn đá bước thứ nhất.

Khi mảng stones, chứa các số được ghi trên từng hòn đá bước theo thứ tự từ hòn thứ nhất, được truyền làm tham số, muốn viết hàm solution để trả về số lần nhảy cần thiết để con ếch băng qua hết tất cả các hòn đá bước.

Hãy điền vào chỗ trống để hoàn thiện toàn bộ mã nguồn.

Mô tả tham số

Mảng stones, chứa các số được ghi trên từng hòn đá bước theo thứ tự từ hòn thứ nhất, được truyền làm tham số cho hàm solution.

Độ dài mảng stones từ 1 đến 100.

Mỗi phần tử trong stones (số được ghi trên hòn đá) là số tự nhiên từ 1 đến 5.

Mô tả giá trị trả về

Hàm solution trả về số lần nhảy cần thiết để con ếch băng qua hết tất cả các hòn đá bước.

Ví dụ

stones	result
[2, 5, 1, 3, 2, 1]	3

Giải thích ví dụ

Ví dụ #1

Đây là ví dụ trong trường hợp có 6 hòn đá bước và lần lượt ghi [2, 5, 1, 3, 2, 1] theo thứ tự.



Ban đầu con ếch đang ngồi trên hòn đá bước thứ nhất, là hòn đá ngoài cùng bên trái.

Trên hòn đá thứ nhất ghi số 2 nên nó nhảy 2 ô để đến hòn đá thứ 3.

Trên hòn đá thứ 3 ghi số 1 nên nó nhảy 1 ô để đến hòn đá thứ 4.

Trên hòn đá thứ 4 ghi số 3 nên nó nhảy 3 ô và băng qua toàn bộ các hòn đá bước.

Trong ví dụ trên, con ếch đã nhảy tổng cộng 3 lần để băng qua tất cả các hòn đá bước.

Code

```
def solution(stones):
    cnt = 0
    current = 0
```

```

n = len(stones)
while @@@:
    current += @@@
    cnt += 1
return cnt

```

PHẦN 2: GỖ LỖI (Câu 6-8)

Câu 6: Đếm số người cao hơn k

Mô tả bài toán

Muốn tìm xem trong danh sách chứa chiều cao của học sinh thì có bao nhiêu người cao hơn k.

학생 키(cm)
165
170
175
180
184

Số người có chiều cao lớn hơn 175 là 2 người.

Khi mảng height chứa chiều cao của học sinh và giá trị k được truyền làm tham số, đã viết hàm solution để đếm và trả về số học sinh có chiều cao lớn hơn k.

Tuy nhiên, do một phần của mã nguồn bị sai nên với một số đầu vào, chương trình không hoạt động đúng.

Hãy sửa lại để chương trình hoạt động đúng với mọi đầu vào, bằng cách **chỉ sửa đúng một dòng** trong đoạn mã đã cho.

Mô tả tham số

Mảng height chứa chiều cao của học sinh và giá trị k được truyền làm tham số cho hàm solution.

Độ dài của height từ 1 đến 100.

Chiều cao của học sinh là số tự nhiên từ 100 đến 300.
Giá trị k là số tự nhiên từ 100 đến 300.

Mô tả giá trị trả về

Hàm solution trả về số học sinh có chiều cao lớn hơn k.

Ví dụ

height	k	result
[165, 170, 175, 180, 184]	175	2

Giải thích ví dụ

Ví dụ #1

Chiều cao lớn hơn 175 là [180, 184], tổng cộng 2 người.

Code (có lỗi)

```
def solution(height, k):  
    answer = 0  
    n = len(height)  
    for h in height:  
        if h >= k:  
            answer += 1  
    return answer
```

Câu 7: Đổi ký tự 'a' thành 'z' và 'z' thành 'a'

Mô tả bài toán

Khi cho một chuỗi, muốn thay các ký tự 'a' trong chuỗi thành 'z', và các ký tự 'z' thành 'a'.
Ví dụ, nếu chuỗi được cho là "abz" thì cần tạo ra chuỗi "zba".

Khi chuỗi s được truyền làm tham số, đã viết hàm solution để thay 'a' thành 'z' và 'z' thành 'a' trong chuỗi rồi trả về kết quả.

Tuy nhiên, do một phần mã nguồn bị sai nên với một số đầu vào chương trình không hoạt động đúng.

Hãy sửa lại để chương trình hoạt động đúng với mọi đầu vào, bằng cách **chỉ sửa đúng một dòng** trong đoạn mã đã cho.

Mô tả tham số

Chuỗi s được truyền làm tham số cho hàm solution.

Chuỗi s chỉ gồm các chữ cái thường trong bảng chữ cái, và có độ dài từ 1 đến 100.

Mô tả giá trị trả về

Hàm solution thay 'a' thành 'z' và 'z' thành 'a' trong chuỗi rồi trả về kết quả.

Ví dụ

s	result
"abz"	"zba"

Giải thích ví dụ

Ví dụ #1

Thay ký tự 'a' thứ nhất thành 'z', và thay ký tự 'z' thứ ba thành 'a' thì được "zba".

Code (có lỗi)

```
def solution(s):
    s_lst = list(s)
    n = len(s)
    for i in range(n):
        if s_lst[i] == 'a':
            s_lst[i] = 'z'
        if s_lst[i] == 'z':
            s_lst[i] = 'a'
    return "".join(s_lst)
```

Câu 8: Đếm tên có chữ 'j' hoặc 'k'

Mô tả bài toán

Từ danh sách chứa tên của học sinh, muốn tìm số học sinh có chữ cái **j** hoặc **k** trong tên.
Ví dụ, trong "james" có chứa chữ j, và trong "jack" thì cả j và k đều có.

Khi mảng name_list chứa tên của học sinh được truyền làm tham số, đã viết hàm solution để đếm và trả về số học sinh có chữ j hoặc k trong tên.

Tuy nhiên, do một phần của mã nguồn bị sai nên với một số đầu vào, chương trình không hoạt động đúng.

Hãy sửa lại để chương trình hoạt động đúng với mọi đầu vào, bằng cách **chỉ sửa đúng một dòng** trong đoạn mã đã cho.

Mô tả tham số

Mảng name_list chứa tên của học sinh được truyền làm tham số cho hàm solution.

Độ dài của name_list từ 1 đến 100.

Tên của học sinh chỉ gồm các chữ cái thường trong bảng chữ cái, và có độ dài từ 1 đến 20.

Không có tên nào bị trùng lặp.

Mô tả giá trị trả về

Hàm solution trả về số học sinh có chữ j hoặc k trong tên.

Ví dụ

name_list	result
["james", "luke", "oliver", "jack"]	3

Giải thích ví dụ

Ví dụ #1

Những người có chữ j hoặc k trong tên là "james", "luke", "jack" tổng cộng 3 người.

Code (có lỗi)

```
def solution(name_list):
    answer = 0
    for name in name_list:
        for n in name:
            if n == 'j' or n == 'k':
                answer += 1
            continue
    return answer
```

PHẦN 3: VIẾT FULL CODE (Câu 9-10)

Câu 9: Tính tiền thừa (거스름돈)

Mô tả bài toán

Khi biết giá của các món hàng mà khách mua và số tiền khách đã trả, muốn tính xem cần phải trả lại bao nhiêu tiền thừa cho khách.

Ví dụ, nếu giá các món hàng đã mua là [2100, 3200, 2100, 800] thì tổng số tiền mua hàng là 8.200 won.

Khi đó, nếu khách trả 10.000 won thì cần trả lại 1.800 won tiền thừa.

Khi mảng price chứa giá các món hàng khách đã mua và số tiền đã trả money được truyền làm tham số, hãy hoàn thành hàm solution để trả về số tiền phải trả lại cho khách.

Mô tả tham số

Mảng price chứa giá các món hàng khách đã mua và số tiền đã trả money được truyền làm tham số cho hàm solution.

Độ dài của price từ 1 đến 50.

Mỗi phần tử của price là giá của một món hàng, là số tự nhiên từ 10 đến 100.000.

money là số tự nhiên từ 10 đến 5.000.000.

Mô tả giá trị trả về

Hãy viết hàm solution để trả về số tiền phải trả lại cho khách.

Nếu số tiền khách trả nhỏ hơn tổng số tiền hàng thì trả về -1.

Ví dụ

price	money	result
[2100, 3200, 2100, 800]	10000	1800

Giải thích ví dụ

Ví dụ #1

Khách lần lượt mua các món có giá 2.100 won, 3.200 won, 2.100 won, 800 won.

Tổng giá trị hàng khách mua là 8.200 won (2.100 + 3.200 + 2.100 + 800), và vì khách đã trả 10.000 won nên cần trả lại 1.800 won tiền thừa.

Code

```
def solution(price, money):
    answer = 0
    return answer
```

Câu 10: Tìm số nhỏ thứ k trong mảng 2 chiều

Mô tả bài toán

Từ một mảng 2 chiều kích thước $n \times 4$ chứa các số tự nhiên, muốn tìm số nhỏ thứ k.

Khi đó, n là độ dài theo chiều dọc (số hàng) của mảng, 4 là độ dài theo chiều ngang (số cột) của mảng.

Ví dụ, dưới đây là một mảng 2 chiều kích thước 4×4 chứa các số tự nhiên.

Trong mảng 2 chiều trên, số nhỏ nhất là 2.

Số nhỏ thứ hai là 4, số nhỏ thứ ba là 5, và số nhỏ thứ tư là 11.

Khi mảng 2 chiều arr và k được truyền làm tham số, hãy hoàn thành hàm solution để tìm và trả về số nhỏ thứ k trong arr.

Mô tả tham số

Mảng 2 chiều arr và số tự nhiên k được truyền làm tham số cho hàm solution.

arr là mảng 2 chiều kích thước $n \times 4$, trong đó n từ 1 đến 20.

n là độ dài theo chiều dọc (số hàng) của arr, và độ dài theo chiều ngang luôn là 4.

Mỗi phần tử trong arr là số tự nhiên từ 1 đến 1.000, và không có số nào bị trùng lặp.

k là số tự nhiên từ 1 đến $n \times 4$.

Mô tả giá trị trả về

Hãy viết hàm solution để tìm và trả về số nhỏ thứ k trong mảng 2 chiều arr.

Ví dụ

arr	k	return
[[5,12,4,31],[24,13,11,2],[43,44,19,26],[33,65,20,21]]	4	11

Giải thích ví dụ

Ví dụ #1

Mảng 2 chiều được cho giống với hình minh họa trong đề.
Khi đó, số nhỏ thứ tư là 11.

Code

```
def solution(arr, k):  
    answer = 0  
    return answer
```