09/10

# Group_08_Exercice_05

January 19, 2021

## 1 Exercice 1:

1/2 pt

Given two integers a and b, which can be positive or negative, find the sum of all the numbers between including them too and return it. If the two numbers are equal return a or b.

Note: a and b are not ordered!

Examples

```
get_sum(1, 0) == 1   // 1 + 0 = 1 get_sum(1, 2) == 3   // 1 + 2 = 3 get_sum(0, 1)
== 1   // 0 + 1 = 1 get_sum(1, 1) == 1   // 1 Since both are same get_sum(-1, 0)
== -1 // -1 + 0 = -1 get_sum(-1, 2) == 2  // -1 + 0 + 1 + 2 = 2
```

[ ]:
```
def get_sum(num1, num2):
    return np.arange(min(num1, num2),max(num1, num2)+1,1).sum()
get_sum(-1, 2)
```

## 2 Exercice 2:

2/2 pts

**Task**

Each day a plant is growing by upSpeed meters. Each night that plant's height decreases by downSpeed meters due to the lack of sun heat. Initially, plant is 0 meters tall. We plant the seed at the beginning of a day. We want to know when the height of the plant will reach a certain level.

**Example**

**For upSpeed = 100, downSpeed = 10 and desiredHeight = 910, the output should be 10.**

```
After day 1 --> 100  After night 1 --> 90  After day 2 --> 190  After night 2 -->
180  After day 3 --> 280  After night 3 --> 270  After day 4 --> 370  After night
4 --> 360  After day 5 --> 460  After night 5 --> 450  After day 6 --> 550  After
night 6 --> 540  After day 7 --> 640  After night 7 --> 630  After day 8 --> 730
After night 8 --> 720  After day 9 --> 820  After night 9 --> 810  After day 10
--> 910
```

**For upSpeed = 10, downSpeed = 9 and desiredHeight = 4, the output should be 1.**

**Because the plant reach to the desired height at day 1(10 meters).**

```
After day 1 --> 10
```

Input/Output

[input] integer upSpeed

A positive integer representing the daily growth.

Constraints: 5   upSpeed   100.

[input] integer downSpeed

A positive integer representing the nightly decline.

Constraints: 2   downSpeed < upSpeed.

[input] integer desiredHeight

A positive integer representing the threshold.

Constraints: 4   desiredHeight   1000.

[output] an integer

The number of days that it will take for the plant to reach/pass desiredHeight (including the last day in the total count).

[ ]:
```python
def growing_plant(upSpeed, downSpeed, desiredHeight):
    days = 1
    height = upSpeed
    day = upSpeed
    night =downSpeed
    while height < desiredHeight:
        height = height + day
        days = days + 1

        height = height - night

    return days
```

[68]:
```python
print(growing_plant(10,2,30))
print(growing_plant(10,9,4))
print(growing_plant(100,10,910))
```

```
4
1
10
```

## 3 Exercice 3: (Use map) 3/3 pts

Given the current exchange rate between the USD and the EUR is 1.1363636 write a function that will accept the Curency type to be returned and a list of the amounts that need to be converted.

Don’t forget this is a currency so the result will need to be rounded to the second decimal.

```
'USD' Return format should be '$100,000.00'

'EUR' Return format for this should be '100,000.00€'
```

to_currency is a string with values ‘USD’,‘EUR’ , values_list is a list of floats

solution(to_currency,values)

#EXAMPLES:

```
solution('USD',[1394.0, 250.85, 721.3, 911.25, 1170.67])
= ['$1,584.09', '$285.06', '$819.66', '$1,035.51', '$1,330.31']

solution('EUR',[109.45, 640.31, 1310.99, 669.51, 415.54])
= ['96.32€', '563.47€', '1,153.67€', '589.17€', '365.68€']
```

[67]:
```
def solution(to_cur, value):
    if to_cur == 'USD':
        def usd(usd):
            return round(usd * 1.1363636,2)
        results = list(map(usd,value))
        newlist =[]
        for item in results:
            newlist.append(str(item))
        list1 = ['$'+item for item in newlist]
        print(list1)

    if to_cur == 'EUR':
        def eur(eur):
            return round(eur/1.1363636,2)
        results = list(map(eur,value))
        newlist =[]
        for item in results:
            newlist.append(str(item))
        list1 = [item +'€' for item in newlist]
        print(list1)



solution('USD',[1394.0, 250.85, 721.3, 911.25, 1170.67])
```

```
['$1584.09', '$285.06', '$819.66', '$1035.51', '$1330.31']
```

[65]:
```
solution('EUR',[109.45, 640.31, 1310.99, 669.51, 415.54])
```

```
['96.32€', '563.47€', '1153.67€', '589.17€', '365.68€']
```

## 4 Exercice 4

3/3 pts

Create a function that takes in the sum and age difference of two people, calculates their individual ages, and returns a pair of values (oldest age first) if those exist or null/None if: `sum < 0` `difference < 0`

`get_ages(24, 4) should return (14, 10) get_ages(63, -14) should return None`

Either of the calculated ages come out to be negative

[69]:
```
def get_ages(summe, difference):
    if (summe >0 and difference > 0):
        y = int((summe + difference)/2)
        x = int(summe - y)
        return (max(x,y), min(x,y))
    else:
        return "None"
```

[70]:
```
get_ages(24, 4)
```

[70]:
```
(14, 10)
```

[71]:
```
get_ages(63, -14)
```

[71]:
```
'None'
```

[ ]:
```
```