Worksheet: Iterations

CS 101 Algorithmic Problem Solving

Fall 2023

Name(s): ______________________

HU ID *(e.g., xy01042)*: ______________________

1. Limit the Earnings

Maaz works at a restaurant that operates in a different way. The restaurant keeps taking orders until the total income of the day reaches (or crosses) 15000 Rupees. Maaz has to figure out the total number of the orders daily before they are stopped.

Given the cost C of each order, figure out the total number of orders taken by the restaurant on a particular day. Keep taking the input until the total cost reaches 15000.

Constraints

- $1 \leq C \leq 5000$

Interaction

The input comprises multiple lines containing an integer denoting the cost of each order.

The output (for all those integers) must contain a single number denoting the total number of orders.

Sample

Input	Output
3000 2500 5000 4600 2000	4

In the above case, order number 1 is for Rs. 3000 then order number 2 is for Rs. 2500 then order number 3 is for Rs. 5000 and order number 4 is for Rs. 4600.
$3000 + 2500 + 5000 + 4600 = 15100$. When order number 4 is taken the total income exceeds Rs. 15000 therefore no more orders will be taken. The output is thus 4.

Exercise

In the space provided, indicate the outputs for the given inputs.

Input	Output
2500 2500 4500 1500 4000 1000	5

Problem Identification
Briefly explain the underlying problem you identified in the above question that led you to your solution.
Answer: Input: C
Output: add the input orders till the total reaches 15000.

Input: C
Output: $\sum_{i=1}^{n} C_i$ where C_i is order number i till $\sum_{i=1}^{n} C_i < 15000$ and n is total orders

Pseudocode

```
total_orders = 0
total_earnings = 0
while total_earnings < 15000:
  total_earnings +=  int(input())
  total_orders += 1
return total_orders
```

Dry Run
Using the inputs provided in the Exercise section above, dry run your psuedocode in the space below.
Input: $(C1, C2, C3, C4, C5, C6) = (2500, 2500, 4500, 1500, 4000, 1000)$
Output:
$total\_orders = 0$
$total\_earnings = 0$
$0 < 15000$ so loop begins
$total\_earnings = 0 + 2500$
$total\_orders = 0 + 1 = 1$
$2500 < 15000$ so loop continues
$total\_earnings = 2500 + 2500 = 5000$
$total\_orders = 1 + 1 = 2$
$5000 < 15000$ so loop continues
$total\_earnings = 5000 + 4500 = 9500$
$total\_orders = 2 + 1 = 3$
$9500 < 15000$ so loop continues
$total\_earnings = 9500 + 1500 = 11000$
$total\_orders = 3 + 1 = 4$
$11000 < 15000$ so loop continues
$total\_earnings = 11000 + 4000 = 15000$
$total\_orders = 4 + 1 = 5$

15000 == 15000 so loop breaks
output is 5 which is the expected output, so correct!

This means the applied logic is correct

2. Frustrated Teacher

A teacher frustrated with the lack of discipline of his students, decides to cancel class if fewer than some number of students are present when class starts. Arrival times t go from on time ($t >= 0$) to arrived late ($t < 0$).

Given the total number of students n, arrival time of each student t and a threshhold number of attendees a, check whether the teacher cancels the class or not.

Constraints

- $0 \leq n, a \leq 200$
- $-10 \leq t \leq 10$

Interaction

The input comprises a line containing 2 space-separated integers denoting the values of n and a respectively. Followed by a line containing n space separated integers each denoting the arrival time t of each student.

The output must contain Yes or No, stating whether the class is cancelled or not.

Sample

Input	Output
5 3	Yes
-1 0 2 -2 -4	
4 2	No
1 0 2 -1	

In the first case, only 2 students arrive on time while the threshold is 3. Therefore, the class is cancelled.

In the second case, 3 students arrive on time while the teacher required atleast 2 to arrive on time. Therefore, the class is not cancelled.

Exercise

In the space provided, indicate the outputs for the given inputs.

Input	Output
6 4	No
-1 0 2 -2 4 1	
5 2	Yes
-1 0 -2 -1 -3	

Problem Identification

Briefly explain the underlying problem you identified in the above question that led you to your solution.

Input: n, a, t (n times)
Output: "Yes" if number of students arriving on time are greater than a else "No"

Pseudocode

```
students_ontime = 0
for i in range(n):
    time = int(input())
    if time >= 0:
        students_ontime += 1
if students_ontime >= a:
    print("No")
else:
    print("Yes")
```

Dry Run

Using any two of the inputs provided in the Exercise section above, dry run your psuedocode in the space below.

Input: $(n, a) = (6, 4)$, $t = -1, 0, 2, -2, 4, 1$

Output:

$students\_ontime = 0$

outer loop runs from $(0, 6)$

i = 0, time $= -1$, time is not $>= 0$ hence nothing happens

i = 1, time $= 0$, $time >= 0$ hence $students\_ontime = 0 + 1 = 1$

i = 2, time $= 2$, $time >= 0$ hence $students\_ontime = 1 + 1 = 2$

i = 3, time $= -2$, time is not $>= 0$ hence nothing happens

i = 4, time $= 4$, $time >= 0$ hence $students\_ontime = 2 + 1 = 3$

i = 5, time $= 1$, $time >= 0$ hence $students\_ontime = 3 + 1 = 4$

$students\_ontime >= 4$ hence "No" is printed on screen, which is the expected output!

Input: $(n, a) = (5, 2)$, $t = -1, 0, -2, -1, -3$

Output:

$students\_ontime = 0$

outer loop runs from $(0, 5)$

i = 0, time $= -1$, time is not $>= 0$ hence nothing happens

i = 1, time $= 0$, $time >= 0$ hence $students\_ontime = 0 + 1 = 1$

i = 2, time $= -2$, time is not $>= 0$ hence nothing happens

i = 3, time $= -1$, time is not $>= 0$ hence nothing happens

i = 4, time $= -3$, time is not $>= 0$ hence nothing happens

$students\_ontime < a$ hence "Yes" is printed as output which is the expected output!

This means the applied logic is correct

3. Magic Potions

Aimen plays a game in which the character competes in a hurdle race. Hurdles are of varying heights, and the characters have a maximum height h they can jump. There is a magic potion they can take that will increase their maximum jump height by 1 unit for each dose.

Given the number of jumps n and maximum height h, figure out how many doses of the potion must the Aimen's character take to be able to jump all of the hurdles. If Aimen can already clear all of the hurdles without taking the potion, return 0.

Constraints

- $1 \leq n, h \leq 25$

Interaction

The input comprises a line containing 2 space-separated integers denoting the values of n and h respectively. Followed by a line containig n integers separated by space, each denoting the height of a hurdle.

The output must contain a single number denoting the total doses of magic potions needed.

Sample

Input	Output
4 3	0
1 2 3 2	
5 7	3
2 7 8 10 3	

In the first case, all hurdles are lower than the maximum height the character can jump. Hence, no magic potion needed.

In the second case, the tallest hurdle is 10 hence 3 doses of the magic potion are needed. If they are taken, the character will be able to jump the hurdles 8 and 10 which are greater than the maximum height.

Exercise

In the space provided, indicate the outputs for the given inputs.

Input	Output
4 3	6
9 6 1 2	
6 8	4
1 3 8 9 10 12	

Problem Identification

Briefly explain the underlying problem you identified in the above question that led you to your solution.
Answer: Input: n, h
Output: number of magic potions

Pseudocode

```
potions = 0
for i in range(n):
    jump = int(input())
    if jump > h:
        potions += jump - h
        h = jump
return potions
```

Dry Run

Using any two of the inputs provided in the Exercise section above, dry run your pseudocode in the space below.
Input: $(n, h) = (4, 3)$
Output:
potions = 0
outer loop runs from $(0, 4)$

i = 0
jump = 9, $jump > h$ $(9 > 3)$
potions = potions + jump - h
= 0 + 9 - 3 = 6
h = 9

i = 1
jump = 6, $jump < h$ $(6 < 9)$
nothing happens

i = 2
jump = 1, $jump < h$ $(1 < 9)$
nothing happens

i = 3
jump = 2, $jump < h$ $(2 < 9)$
nothing happens

potions = 6 which is printed as output, which is the expected output.

Input: $(n, h) = (6, 8)$
Output:
potions = 0
outer loop runs from $(0, 6)$
i = 0
jump = 1, $jump < h$ $(1 < 8)$
nothing happens

i = 1
jump = 3, $jump < h$ $(3 < 8)$
nothing happens

i = 2
jump = 8, $jump > h$ is false as (8 is not > 8)
nothing happens

i = 3
jump = 9, $jump > h$ $(9 > 8)$
potions = potions + jump - h
= 0 + 9 - 8 = 1
h = 9

i = 4
jump = 10, $jump > h$ $(10 > 9)$
potions = potions + jump - h

= 1 + 10 - 9 = 2
h = 10

i = 5
jump = 12, $jump > h$ (12 > 10)
potions = potions + jump - h
= 2 + 12 - 10 = 4
h = 12

potions = 4 which is printed as output, which is the expected output!

This means the applied logic is correct

4. Beautiful Days

Lila goes to the market only on beautiful days. Given a range of numbered days starting from s and ending at e (where s and e are both included in the range) and a number k, determine the number of days in the range that are beautiful. Beautiful numbers are defined as numbers where n - reverse(n) is evenly divisible by k. If a day's value is a beautiful number, it is a beautiful day.

Given a range, find the number of days Lila will go to the market.

Hint: String can be reversed using [::-1].

Constraints

- $11 \leq s, e \leq 31$
- $0 \leq k \leq 10$

Interaction

The input comprises a single line containing 3 space-separated integers denoting the values of s, e and k respectively.

The output must contain a single number denoting the total number of days Lila goes to the market.

Sample

Input	Output
20 22 6	2
12 15 3	4

In the first case, the range of days is from 20 to 22 inclusive. The reverse of 20 is 02 or 2, reverse of 21 is 12 and reverse of 22 is 22. So (20 - 2) mod 6 = 0 and (21 - 12) mod 6 = 3 and (22 - 22) mod 6 = 0. Which means there are only two beautiful days within the range of 20 to 22. Lila will go to the market on these 2 days only.

In the second case, the range is from 12 to 15 inclusive. Reverse numbers of 12, 13, 14, 15 are 21, 31, 41, 51 respectively. (12 - 21) mod 3 = 0 and (13 - 31) mod 3 = 0 and (14 - 41) mod 3 = 0 and (15 - 51) mod 3 = 0. Which means all 4 days are beautiful therefore, Lila will go to the market for 4 days.

Exercise

In the space provided, indicate the outputs for the given inputs.

Input	Output
12 14 4	0
27 31 5	1
17 19 6	2

Problem Identification
Briefly explain the underlying problem you identified in the above question that led you to your solution.

Input: s, e, k
Output: total days that satisfy the condition $(n - reverse(n))\%k == 0$ in the range$(s, e + 1)$.

Pseudocode

```
beautiful_days = 0
for day in range(s, e+1):
    reverse = str(day)[::-1]
    if (day - int(reverse)) % k == 0:
        beautiful_days += 1
return beautiful_days
```

Dry Run
Using any two of the inputs provided in the Exercise section above, dry run your psuedocode in the space below. Input: $(s, e, k) = (12, 14, 4)$
Output:
$beautiful\_days = 0$
outer loop runs from $(12, 15)$
day = 12, reverse $= 21$
$(12 - 21)\%4 \neq 0$ hence nothing happens
day = 13, reverse $= 31$
$(12 - 31)\%4 \neq 0$ hence nothing happens
day = 14, reverse $= 41$
$(12 - 41)\%4 \neq 0$ hence nothing happens
output is 0 which is the expected output!

Input: $(s, e, k) = (27, 31, 5)$
Output:
$beautiful\_days = 0$
outer loop runs from $(27, 31)$
day = 27, reverse $= 72$
$(27 - 72)\%5 == 0$ hence $beautiful\_days = 0 + 1 = 1$
day = 28, reverse $= 82$
$(28 - 82)\%5 \neq 0$ hence nothing happens
day = 29, reverse $= 92$
$(29 - 92)\%5 \neq 0$ hence nothing happens
day = 30, reverse $= 3$
$(30 - 3)\%5 \neq 0$ hence nothing happens
day = 31, reverse $= 13$

$(31 - 13)\%5 \neq 0$ hence nothing happens
output is 1 which is the expected output!

This means the applied logic is correct

5. War in the Milkyway

Ali is playing a game 'War in the Milkyway'. The Milkyway is a galaxy with N planets, two of which have power sources namely $R1$ and $R2$. Planets are numbered consecutively from 0 and each planet is at a distance of 1 km to the next planet. It is not circular, so the first planet doesn't connect with the last planet.

Ali needs to determine the maximum distance from any planet to its nearest power source, in order to plan efficiently. Help him with this task.

Hint: using min(), max() and abs() may help.

Constraints

- $0 \leq R1, R2 \leq 100$
- $1 \leq N \leq 100$

Interaction

The input comprises a single line containing 3 space-separated integers denoting the values of $N, R1$ and $R2$ respectively.

The output must contain a single number denoting the maximum distance.

Sample

Input	Output
5 0 4	2
3 6 7	0

In the first case, there are total 5 planets. Planet 0 and 4 have the distance of 0 to the nearest power source. Planet 1 and 3 have power source 0 and 4 respectively at a distance of 1 km. Planet 2 is 2 km away from both the power sources hence, the maximum distance for all 5 planets is 2 km.

In the second case, there are 3 planets but stations are said to be at planets numbered 6 and 7 which do not exist, hence the output is 0.

Exercise

In the space provided, indicate the outputs for the given inputs.

Input	Output
7 0 4	2
4 9 6	0
11 2 10	4

Problem Identification

Briefly explain the underlying problem you identified in the above question that led you to your solution.

Input: $n, R1, R2$
Output: maximum of all the distances between planets and the power sources which can be calculated by taking the minimum of the distance from each power source.

Pseudocode

```
max_d = 0
if r1 <= n and r2 <= n:
    for planet in range(n):
        distance = min(abs(planet - r1), abs(planet - r2))
        if distance > max_d:
        max_d = distance
print(max_d)
```

Dry Run

Using any two of the inputs provided in the Exercise section above, dry run your psuedocode in the space below. Input: $(n, r1, r2) = (7, 0, 4)$
Output:
$max\_d = 0$
$r1 <= n(0 <= 7)$ and $r2 <= n(4 <= 7)$ - condition true
loop runs from range 0 to 7
planet = 0, distance = min(abs(0-0),abs(0-4)) = min(0,4) = 0, $distance > max\_d$ i.e. 0 is not > 0 hence, nothing happens. Loop continues.
planet = 1, distance = min(abs(1-0),abs(1-4)) = min(1,3) = 1, $distance > max\_d$ i.e. $1 > 0$ hence $max\_d = 1$
planet = 2, distance = min(abs(2-0),abs(2-4)) = min(3,2) = 2, $distance > max\_d$ i.e. $2 > 1$ hence $max\_d = 2$
planet = 3, distance = min(abs(3-0),abs(3-4)) = min(3,1) = 1, $distance < max\_d$ i.e. $1 < 2$ hence no change
planet = 4, distance = min(abs(4-0),abs(4-4)) = min(4,0) = 0, $distance < max\_d$ i.e. $0 < 2$ hence no change
planet = 5, distance = min(abs(5-0),abs(5-4)) = min(5,1) = 1, $distance < max\_d$ i.e. $1 < 2$ hence no change
planet = 6, distance = min(abs(6-0),abs(6-4)) = min(6,2) = 2, $distance > max\_d$ is false i.e. $2 > 2$ hence no change
output is 2 which is the expected output!

Input: $(n, r1, r2) = (4, 9, 6)$
Output:
$max\_d = 0$
$r1 <= n(9 <= 4)$ and $r2 <= n(6 <= 4)$ - condition false
loop does not run, max_d remains 0, output is 0 which is the expected output.

This means the applied logic is correct

6. Restaurant Orders

Sami needs to find the total of all the the odd numbers given in a particular range (from a to b inclusive). Help him with a code that can print the total.

Constraints

- $0 \leq a, b \leq 200$

Interaction

The input comprises a single line containing 2 space-separated integers denoting the values of a and b.

The output must contain a single number denoting the total.

Sample

Input	Output
1 10	25
9 11	20

In the first case, there are 5 odd numbers 1, 3, 5, 7 and 9 which totals to 25.

In the second case, there are only 2 odd numbers 9 and 11 hence the total is 20.

Proposed Solution

```
total = 0
for i in range(a, b, +1):
  total += i
print(total)
```

Dry Run
Using any two of the inputs provided in the Sample section above, dry run the proposed psuedocode in the space below.

Input: (a,b) = (1,10)
Output:

i	total
1	1
2	3
3	6
4	10
5	15
6	21
7	28
8	36
9	45

total is 45 which is **NOT** the expected output!

Input: (a,b) = (9,11)
Output:

i	total
9	9
10	19

total is 19 which is **NOT** the expected output!

Error Identification
Briefly explain the errors you identified in the proposed code solution. Mention the line number and errors in each line.
Answer: If we want to add only odds, then in the range for loop we need to increment by

2 rather than 1. Moreover, as b is inclusive too we need to add 1 to it in the range as the ending is not inclusice in the for loop range.

Correct Solution
Rewrite the lines of code you mentioned above with their errors corrected.

```
total = 0
for i in range(a, b+1, +2):
    total += i
print(total)
```

Rough Work