



You can view this report online at : <https://www.hackerrank.com/x/tests/1173405/candidates/38253222/report>

Full Name:

Fabio gomez guzman

Email:

fabiogomez28@gmail.com

Test Name:

20210917 PRUEBA |Lógica de programación

Taken On:

28 Mar 2022 10:13:31 -05

Time Taken:

170 min 28 sec/ 180 min

Work Experience:

> 5 years

Invited by:

Carlos

Skills Score:

Tags Score:

Arithmetic

50/50

Basic implementation

50/50

Greedy Algorithms

50/50

Logic

50/50

Math

100/100

Strings

50/50

ad hoc

150/150

100%

250/250

scored in 20210917 PRUEBA

|Lógica de programación in

170 min 28 sec on 28 Mar 2022

10:13:31 -05

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Robot en Marte > Coding	1 hour 3 min 23 sec	50/ 50	✓
Q2	El programador perfeccionista > Coding	26 min 3 sec	50/ 50	✓
Q3	Estimación Perdida > Coding	27 min 30 sec	50/ 50	✓
Q4	La deuda > Coding	23 min 49 sec	50/ 50	✓
Q5	Numeros diferentes > Coding	29 min 26 sec	50/ 50	✓

QUESTION 1

✓

Correct Answer

Score 50

Robot en Marte > Coding

Strings

ad hoc

QUESTION DESCRIPTION

La NASA nos ha contratado y como empresa llevaremos nuestro primer robot a Marte.

Nuestro robot se mueve 1 metro en cada dirección con el comando Izquierda (L) Derecha (R) Arriba (U) Abajo (D).

La NASA prepara una lista de indicaciones para el movimiento del robot desde la base de carga en el ejemplo marcada como punto 0.

Sin embargo están preocupados porque en caso de una emergencia el robot pueda regresar a tiempo a la base de carga y quieren que evaluemos los planes de movimiento en un simulador, y les digamos la cantidad de instrucciones máximas que deberíamos enviar al robot cuando se encuentre en su punto más lejano para que pueda retornar a la base.

Calcule cuál es el número máximo de instrucciones que debería enviarse al robot para que en algún punto del recorrido regrese a la base.

### Function Description

Complete la función abajo para completar la tarea requerida, la función tendrá una lista de planes a ejecutar, evalúe cada uno y retorne una lista con el numero máximo de instrucciones

### Constraints

```
len(instruccion) <= 10000
```

#### ▼ Input Format For Custom Testing

Primero ingresara un entero N definiendo la cantidad de planes que la NASA quiere evaluar, luego existirán N líneas con las cadenas de instrucciones

#### ▼ Sample Case 0

##### Sample Input For Custom Testing

```
1
RUULLLDDDR
```

##### Sample Output

```
4
```

##### Explanation

**Ruta:** *RUULLLDDDR* el robot se moverá como se ve en la imagen

6	5	4	3	
7			2	
8		0	1	
9	10			

Siguiendo esta ruta, el punto 6 sería el punto más lejano de la base, y necesitaría 4 instrucciones para poder retornar a la base, (RDRD o RRDD o DDDR o DRDR).

#### ▼ Sample Case 1

##### Sample Input For Custom Testing

```
2
U
UUU
```

##### Sample Output

```
1
3
```

# CANDIDATE ANSWER

Language used: Python 3

```

1  #
2  # Complete the 'calcularMaximoRetorno' function below.
3  #
4  # The function is expected to return an INTEGER_ARRAY.
5  # The function accepts STRING_ARRAY instruccions as parameter.
6  #
7
8  def calcularMaximoRetorno(instruccions):
9
10     max_sum = [0] * len(instruccions)
11
12     for idx, string in enumerate(instruccions):
13
14         x = 0
15         y = 0
16         tmp_max_sum = 0
17         for idx2, step in enumerate(string):
18             #accumulate value of every step in x and y
19             x, y = getStepValue(x, y, step)
20             #calcula de sum of absolute values of steps
21             tmp_max_sum = abs(x) + abs(y)
22             #save the step with maximun value
23             if tmp_max_sum > max_sum[idx]:
24                 max_sum[idx] = tmp_max_sum
25
26     return max_sum
27
28 # Calculate the Move in a plane x, y do it in every step
29 def getStepValue(x, y, step):
30
31     if step == "U":
32         return x, y + 1
33     if step == "D":
34         return x, y - 1
35     if step == "L":
36         return x - 1, y
37     else:
38         return x + 1, y
39
40
41
42
43
44
45
46     # Write your code here

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Sample	Easy	Sample case	✓ Success	0	0.0696 sec	9.49 KB
Sample 2	Easy	Sample case	✓ Success	0	0.0544 sec	9.54 KB
Border Cases	Easy	Hidden case	✓ Success	25	0.0925 sec	9.49 KB
Random 2	Easy	Hidden case	✓ Success	25	0.5385 sec	10.4 KB

## QUESTION 2



Correct Answer

Score 50

El programador perfeccionista > Coding Logic ad hoc

## QUESTION DESCRIPTION

Ricardo es miembro de nuestro equipo, y aunque suele ser muy productivo, sufre de un impulso que le lleva a perfeccionar y reescribir una parte del trabajo que hace cada día.

Cada día es capaz de escribir F cantidad de funciones nuevas, pero su impulso lo lleva el mismo día a borrar R cantidad de las funciones, y así cada día, increíblemente cuando termina su trabajo no vuelve a refactorizar ninguna función, sino que envía el pull request a review.

Todos aman el trabajo de Ricardo, pero el arquitecto del equipo está preocupado por el deadline (fecha de entrega) de las historias que le asigna a Ricardo, y te pide ayuda con un programa que determine si Ricardo podrá cumplir con su asignación.

Para ello el arquitecto te entregará un número D correspondiente al número de días del deadline de entrega, T la cantidad de funciones totales que espera el cliente que realice Ricardo, F la cantidad de funciones nuevas que escribirá Ricardo por día, R la cantidad de funciones que borrará Ricardo cada día al finalizar su trabajo.

Calcule si Ricardo puede o no cumplir con las tareas asignadas.

```
true      Lo logrará
false     No lo logrará
```

La plantilla convertirá el true en 1 y false en 0 automáticamente

**Function Description**

Complete la función, que recibirá un arreglo de casos a evaluar por el arquitecto.  
cada caso se compondrá de la siguiente manera

- caso[i][0] (D) Número de días para el deadline
- caso[i][1] (T) Número de funciones esperadas por el cliente al finalizar el deadline
- caso[i][2] (F) Número de funciones nuevas que puede escribir Ricardo en un día
- caso[i][3] (R) Número de funciones que borrará Ricardo al finalizar el día

**Constraints**

```
0 < D <= 10000
1 <= T <= 5000
1 <= F <= 5000
F <= R <= 5000
```

**▼ Input Format For Custom Testing**

El caso comienza con un número N que representa el número de posibilidades que el arquitecto del equipo quiere evaluar.

Luego una línea con el número 4 que representa las 4 variables a leer

Posteriormente vienen N líneas cada una con 4 variables de la siguiente forma

```
D T F R
```

### ▼ Sample Case 0

#### Sample Input For Custom Testing

```
1
4
5 110 30 10
```

#### Sample Output

```
1
```

#### Explanation

El cliente espera en 5 días que Ricardo escriba 110 funciones, Ricardo podrá escribir 30 funciones, pero finalizando el día borrará 10 de las funciones escritas.

Cada día Ricardo escribirá 25 funciones, pero borrará 5, al finalizar el día 4 Ricardo tendrá 80 funciones en código, como el día 5 podría escribir 30 funciones nuevas terminaría su trabajo y enviaría el pull request sin borrar ninguna función, Ricardo cumpliría con el deadline.

Dia	Total Funciones
Dia 1	20
Dia 2	40
Dia 3	80
Dia 4	60
Dia 5	110

### ▼ Sample Case 1

#### Sample Input For Custom Testing

```
1
4
2 40 20 5
5 110 30 10
```

#### Sample Output

```
0
1
```

#### Explanation

En este caso Ricardo no lograría cumplir su cometido, ya que el día 2 tendrá a lo sumo 35 funciones escritas de las 40 requeridas por el cliente.

### CANDIDATE ANSWER

Language used: **Python 3**

```
1 #
2 # Complete the 'podraCumplir' function below.
3 #
4 # The function is expected to return a BOOLEAN_ARRAY.
5 # The function accepts 2D_INTEGER_ARRAY caso as parameter.
6 #
7
8 def podraCumplir(caso):
9     arr_result = [0]*len(caso)
10     for idx, evaluate in enumerate(caso):
11         deadline_days = evaluate[0]
```

```

12     deadline_functions = evaluate[1]
13     add_functions = evaluate[2]
14     remove_functions = evaluate[3]
15
16     #calculate number of functions given by developer, knowing that the
17     last day they can't delete any function
18     total_add_functions = (deadline_days-1) * (add_functions -
19     remove_functions) + add_functions
20
21     #validate if he can comply with the deadline
22     if (total_add_functions >= deadline_functions):
23         arr_result[idx] = 1
24     else:
25         arr_result[idx] = 0
26
27     return arr_result

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Sample	Easy	Sample case	✔ Success	0	0.0511 sec	9.4 KB
Sample 2	Easy	Sample case	✔ Success	0	0.0627 sec	9.44 KB
All true	Easy	Hidden case	✔ Success	15	0.0471 sec	9.72 KB
Random	Easy	Hidden case	✔ Success	15	0.0777 sec	9.7 KB
Bordes	Easy	Hidden case	✔ Success	20	0.067 sec	9.46 KB

No Comments

### QUESTION 3



Correct Answer

Score 50

## Estimación Perdida > Coding Math Arithmetic

### QUESTION DESCRIPTION

Diego es un líder de proyectos, que muy juicioso llevaba el control de las estimaciones de las tareas asignadas a los diferentes equipos y personas en un Excel, pero por desgracia el archivo quedó corrupto y no pudo ser rescatado.

Sin embargo Diego tenía un resumen que entregaba a su jefe cada semana, donde le contaba por equipo, la cantidad de tareas que dicho equipo tenía, el promedio aritmético de las estimaciones de tareas definido en horas, y la duración de la tarea con menor valor estimado.

Diego necesita saber para una presentación hoy con el cliente, ¿Cuál es la máxima duración que puede tener una de las tareas? Y te ha pedido ayuda para resolver el problema.

### Function Description

Complete la función en el editor abajo, la función recibe la lista de equipos con 3 valores (N,R,T), y debe retornar una lista que representa el numero de días necesario para pagar cada deuda.

- `equipo[j][0] = (N)` El número de tareas asignadas en el equipo `j`
- `equipo[j][1] = (T)` El número de horas promedio de las tareas en el equipo `j`
- `equipo[j][2] = (R)` La duración en horas de la tarea más pequeña en el equipo `j`

### Constraints

```

2 <= N <= 100
R <= T <= 2000
1 <= R <= 2000

```

### ▼ Input Format For Custom Testing

La primera línea contiene el numero de equipos y la constante 3

Luego viene una línea por cada equipo con las variables N, T y R en ese orden

### ▼ Sample Case 0

#### Sample Input For Custom Testing

```
2 3
2 4 2
3 16 8
```

#### Sample Output

```
6
32
```

#### Explanation

Si el equipo (1) tiene 2 tareas asignadas, el promedio para hacer dichas tareas es 4 horas y la tarea más pequeña tiene una duración estimada de 2 horas, la tarea más grande debería ser de 6 horas.

$$\text{AVG}(6, 2) = 4$$

Si el equipo (2) tiene 3 tareas asignadas, el promedio para hacer dichas tareas es 16 horas y la tarea más pequeña tiene una duración estimada de 8 horas, la tarea más grande nunca pasará las 32 horas. Si hacemos la tarea 1 la más pequeña de 8 horas, la tarea 2 la hacemos lo más grande posible que son 32 horas y la tarea 3 de 8 horas más, el promedio serán las 16 horas estipuladas.

$$\text{AVG}(32, 8, 8) = 16$$

### ▼ Sample Case 1

#### Sample Input For Custom Testing

```
2
3 2 1
4 1 1
```

#### Sample Output

```
4
1
```

### CANDIDATE ANSWER

Language used: **Python 3**

```
1 #
2 # Complete the 'maximoNumeroHoras' function below.
3 #
4 # The function is expected to return an INTEGER_ARRAY.
5 # The function accepts 2D_INTEGER_ARRAY equipo as parameter.
6 #
7
8 def maximoNumeroHoras(equipo):
9
10     arr result = [0]*len(equipo)
```

```

11
12     for idx, evaluate in enumerate(equipo):
13         tasks = evaluate[0]
14         mean = evaluate[1]
15         minmun_task = evaluate[2]
16         rest_of_task = tasks - 1
17         # solve x (maximun_task) using the average formula mean = ((y0 +...+
18 yn-1 + x)
19         maximun_task = (mean * tasks) - ( rest_of_task * minmun_task)
20         arr_result[idx] = maximun_task
21     return arr_result
22
23
24
25
26
27

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Sample	Easy	Sample case	✔ Success	0	0.0569 sec	9.46 KB
Sample 2	Easy	Sample case	✔ Success	0	0.0635 sec	9.53 KB
Random	Easy	Hidden case	✔ Success	20	0.056 sec	9.41 KB
Random 2	Easy	Hidden case	✔ Success	20	0.0513 sec	9.52 KB
Simple Test	Easy	Hidden case	✔ Success	10	0.0767 sec	9.46 KB

No Comments

#### QUESTION 4



Correct Answer

Score 50

### La deuda > Coding Math Basic implementation

#### QUESTION DESCRIPTION

Una nueva aplicación de préstamos sin cuotas de manejo, ni intereses, quiere probar una nueva modalidad de pagos en los préstamos, la misma consiste en que el primer día del préstamo la persona deba pagar solo 1 peso, y cada día que pasa deberá pagar el doble de lo que pagó el día anterior y así sucesivamente. El último día solo deberá pagar lo que le falte.

Los interesados en la aplicación quieren hacer un pequeño simulador, que permita conocer la cantidad de días que se requieren para pagar una deuda en totalidad con este sistema, dada la cantidad a prestar inicialmente.

Dado un número N deberás devolver un entero indicando la cantidad de días requerido para resolverlo.

#### Function Description

Complete la función en el editor abajo, la función recibe la lista de deudas, y debe retornar una lista que representa el numero de días necesario para pagar cada deuda.

#### Constraints

$0 \leq \text{deuda} \leq 9223372036854775807$

▼ Input Format For Custom Testing



La primera línea contiene un entero n, que denota el numero de prestamos que se realizará.  
Luego existen n líneas cada con el valor de un préstamo específico

#### ▼ Sample Case 0

##### Sample Input For Custom Testing

```
1
15
```

##### Sample Output

```
4
```

##### Explanation

Si a una persona le prestamos 14 pesos, requerirá de 4 días para pagar la totalidad de la deuda

Dia 1	1	total 1
Dia 2	1+2	total 3
Dia 3	1+2+4	total 7
Dia 4	1+2+4+8	total 15

#### ▼ Sample Case 1

##### Sample Input For Custom Testing

```
3
15
16
45
```

##### Sample Output

```
4
5
6
```

#### CANDIDATE ANSWER

Language used: Python 3

```
1 #
2 # Complete the 'calcularDias' function below.
3 #
4 # The function is expected to return a LONG_INTEGER_ARRAY.
5 # The function accepts LONG_INTEGER_ARRAY prestamo as parameter.
6 #
7
8 def calcularDias (prestamo) :
9
10
11     arr_result = [0] * len(prestamo)
12     for idx, one_debt in enumerate(prestamo) :
13         #set the first pay
14         sum_payed = 1
15         days = 1
16         j=1
17         #iterate while te persons don't complete the pay
18         while sum_payed < one_debt:
19             # 2 raised to the power of days, the result is the pay at this
20             day plus one
21             sum_payed += 2**days
```

```

22         days += 1
23         arr_result[idx] = days
24
25     return arr_result
26
27
28
29
30
31
32
33     # Write your code here
34

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Sample	Easy	Sample case	✓ Success	0	0.051 sec	9.46 KB
Sample 0	Easy	Sample case	✓ Success	0	0.059 sec	9.46 KB
<1000	Easy	Hidden case	✓ Success	10	0.0647 sec	9.29 KB
< 1000000	Easy	Hidden case	✓ Success	10	0.0679 sec	9.42 KB
< 100000000	Easy	Hidden case	✓ Success	10	0.0644 sec	9.46 KB
Max	Easy	Hidden case	✓ Success	20	0.0736 sec	9.3 KB

No Comments

#### QUESTION 5



Correct Answer

Score 50

### Numeros diferentes > Coding Greedy Algorithms ad hoc

#### QUESTION DESCRIPTION

A tu equipo de desarrollo el cliente les ha pedido realizar un algoritmo de indexación y optimización de almacenamiento en el proceso de inventario.

Sin embargo el equipo tiene problemas con un método que no saben como realizar de forma óptima y te han pedido ayuda para resolverlo.

Siguiendo el Principio de responsabilidad única (SRP), vas a construir una función que tiene como única responsabilidad calcular la cantidad de números diferentes dentro de una lista dada.

#### Function Description

Complete la función en el editor abajo, la función recibe la lista de tareas, y debe retornar la cantidad de valores únicos que existen.

#### Constraints

$$0 \leq N \leq 100.000$$

$$0 \leq x_i \leq 1.000.000.000$$

#### ▼ Input Format For Custom Testing

La primera línea contiene un entero  $n$ , que denota el numero de elementos en la lista  
Luego existen  $n$  líneas cada una con indicando el elemento  $x_i$  de la lista

#### ▼ Sample Case 0

#### Sample Input For Custom Testing

```

5
1

```

3  
5  
2  
5

#### Sample Output

3

#### Explanation

si tenemos tareas con duración {5, 3, 5, 2, 5}

5, 2, 3 son los elementos diferentes

#### ▼ Sample Case 1

#### Sample Input For Custom Testing

4  
1  
2  
3  
4

#### Sample Output

4

### CANDIDATE ANSWER

Language used: **Python 3**

```
1 #  
2 # Complete the 'cantidadMinimaDeDias' function below.  
3 #  
4 # The function is expected to return an INTEGER.  
5 # The function accepts INTEGER_ARRAY x as parameter.  
6 #  
7  
8 def cantidadMinimaDeDias(x):  
9  
10     return len(list(set(x)))  
11  
12  
13  
14  
15
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Basic Sample	Easy	Sample case	✔ Success	0	0.0664 sec	9.43 KB
Basic non repeated	Easy	Hidden case	✔ Success	10	0.0579 sec	9.49 KB
Basic all repeated	Easy	Hidden case	✔ Success	10	0.0521 sec	9.28 KB
Basic 50/50	Easy	Hidden case	✔ Success	10	0.0724 sec	9.44 KB
Advance Random	Medium	Hidden case	✔ Success	10	0.3447 sec	19.8 KB
Advance 50/50	Medium	Hidden case	✔ Success	10	0.2543 sec	15.7 KB
Sample 2	Easy	Sample case	✔ Success	0	0.0515 sec	9.28 KB

