You can view this report online at : https://www.hackerrank.com/x/tests/1173405/candidates/31215838/report

Full Name:	Daniel Fernando Ladino Yate
Email:	dflyate@gmail.com
Test Name:	**20210917 PRUEBA |Lógica de programación**
Taken On:	6 Oct 2021 10:34:19 -05
Time Taken:	179 min 58 sec/ 180 min
Work Experience:	4 years
Invited by:	Carlos
Skills Score:	
Tags Score:	Arithmetic 0/50 Basic implementation 30/50 Greedy Algorithms 30/50 Logic 50/50 Math 30/100 Strings 0/50 ad hoc 80/150

44%

110/250

scored in **20210917 PRUEBA |Lógica de programación** in 179 min 58 sec on 6 Oct 2021 10:34:19 -05

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Robot en Marte > Coding	20 min 37 sec	0/ 50	✗
Q2	El programador perfeccionista > Coding	17 min 51 sec	50/ 50	✓
Q3	Estimación Perdida > Coding	31 min 55 sec	0/ 50	✗
Q4	La deuda > Coding	50 min 22 sec	30/ 50	✓
Q5	Numeros diferentes > Coding	17 min	30/ 50	✓

QUESTION 1

✗ Wrong Answer

Score 0

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 DDRR o DRDR).

▼ Sample Case 1

Sample Input For Custom Testing

```
2
U
UUU
```

Sample Output

```
1
3
```

CANDIDATE ANSWER

Language used: **Java 8**

```
class Result {

    /*
     * Complete the 'calcularMaximoRetorno' function below.
     *
     * The function is expected to return an INTEGER_ARRAY.
     * The function accepts STRING_ARRAY instruccion as parameter.
     */

    public static List<Integer> calcularMaximoRetorno(List<String>
instruccion) {
    List<String> movimientos = new ArrayList<>();
    List<Integer> cantidad ;
    int suma = 0;
    for(String i : instruccion){
        for(String m : movimientos){
            if(!m.equals(i))
                suma+=2;
            else
                suma++;
        }
        movimientos.add(i);
    }

    }

}
```

Result: Compilation Failed

Compile Message

```
                Solution.java:37: error: missing return statement
    }
    ^
1 error
```

No Comments

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

Ricardo podrá cumplir con su asignación.

Para ello el arquitecto te entregará un numero 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 numero N que representa el numero de posibilidades que el arquitecto del equipo quiere evaluar.

Luego una línea con el numero 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: **Java 8**

```java
class Result {

    /*
     * Complete the 'podraCumplir' function below.
     *
     * The function is expected to return a BOOLEAN_ARRAY.
     * The function accepts 2D_INTEGER_ARRAY caso as parameter.
     */

    public static List<Boolean> podraCumplir(List<List<Integer>> caso) {
        List<Boolean> resultados = new ArrayList<>();
        for(List<Integer> lista : caso){
             int cont = 1, acum = 0;
             while(cont < lista.get(0)){

                 acum += lista.get(2) - lista.get(3);

                 cont++;
             }
             acum += lista.get(2);
             resultados.add(acum < lista.get(1) ? false : true);
        }
        return resultados;

    }

}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Sample	Easy	Sample case	Success	0	0.1974 sec	30.4 KB
Sample 2	Easy	Sample case	Success	0	0.1635 sec	30.2 KB
All true	Easy	Hidden case	Success	15	0.2695 sec	32.6 KB
Random	Easy	Hidden case	Success	15	0.2605 sec	34.6 KB
Bordes	Easy	Hidden case	Success	20	0.2041 sec	30.1 KB

No Comments

QUESTION 3

Wrong Answer

Score 0

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.

```
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.

```
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: **Java 8**

```java
class Result {

    /*
     * Complete the 'maximoNumeroHoras' function below.
     *
     * The function is expected to return an INTEGER_ARRAY.
     * The function accepts 2D_INTEGER_ARRAY equipo as parameter.
     */

    public static List<Integer> maximoNumeroHoras(List<List<Integer>> equipo)
{
        List<Integer> resultados = new ArrayList<>();
        int promedio = 0;
        for(List<Integer> lista: equipo){
            promedio = lista.get(1)+ lista.get(2);
            resultados.add(promedio);
        }
        return resultados;
    }

}

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Sample	Easy	Sample case	⊗ Wrong Answer	0	0.1589 sec	30.2 KB

Sample 2	Easy	Sample case	Wrong Answer	0	0.1947 sec	30.3 KB
Random	Easy	Hidden case	Wrong Answer	0	0.2124 sec	30.4 KB
Random 2	Easy	Hidden case	Wrong Answer	0	0.1532 sec	30.4 KB
Simple Test	Easy	Hidden case	Wrong Answer	0	0.168 sec	29.9 KB

No Comments

QUESTION 4

Correct Answer

Score 30

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 ≤ deuda ≤ 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 especifico

▼ 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: **Java 8**

```java
class Result {

    /*
     * Complete the 'calcularDias' function below.
     *
     * The function is expected to return a LONG_INTEGER_ARRAY.
     * The function accepts LONG_INTEGER_ARRAY prestamo as parameter.
     */

    public static List<Long> calcularDias(List<Long> prestamo) {
        Long contUni = 1l;
        int mult= 1;
        List<Long> resultados = new ArrayList<>();
        for(int i = 0; i< prestamo.size() ; i++){
                while(mult < prestamo.get(i)){
                    mult += (mult+1);
                    contUni++;
                }
                resultados.add(contUni);
                contUni = 1l;
                mult = 1;
        }
        return resultados;
    }

}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Sample	Easy	Sample case	Success	0	0.2004 sec	30 KB
Sample 0	Easy	Sample case	Success	0	0.2039 sec	30.1 KB
<1000	Easy	Hidden case	Success	10	0.151 sec	30.1 KB
< 1000000	Easy	Hidden case	Success	10	0.2122 sec	30 KB
< 100000000	Easy	Hidden case	Success	10	0.173 sec	30.2 KB
Max	Easy	Hidden case	Terminated due to timeout	0	4.0339 sec	199 KB

No Comments

QUESTION 5

Correct Answer

Score 30

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 ≤ N ≤ 100.000
0 ≤ Xi ≤ 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 Xi de la lista

▼ Sample Case 0

Sample Input For Custom Testing

```
5
5
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: **Java 8**

```java
class Result {

    /*
     * Complete the 'cantidadMinimaDeDias' function below.
     *
     * The function is expected to return an INTEGER.
     * The function accepts INTEGER_ARRAY x as parameter.
     */

    public static int cantidadMinimaDeDias(List<Integer> x) {
        List<Integer> unicos = new ArrayList<>();
        int band;
        for(Integer num : x){
            band = 0;
            for(Integer uni: unicos){
                if(uni==num){
                    band = 1;
                    break;
                }
            }
            if(band == 0){
                unicos.add(num);
            }
        }
        return unicos.size();
    }

}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Basic Sample	Easy	Sample case	Success	0	0.1561 sec	29.8 KB
Basic non repeated	Easy	Hidden case	Success	10	0.1865 sec	29.9 KB
Basic all repeated	Easy	Hidden case	Success	10	0.1559 sec	30.1 KB
Basic 50/50	Easy	Hidden case	Success	10	0.2147 sec	30 KB
Advance Random	Medium	Hidden case	Terminated due to timeout	0	4.012 sec	56.1 KB
Advance 50/50	Medium	Hidden case	Terminated due to timeout	0	4.0075 sec	56.8 KB
Sample 2	Easy	Sample case	Success	0	0.1795 sec	29.9 KB

No Comments