Curso Programación Competitiva 2021

Estadísticas y Soluciones

Programación Dinámica

Clasificación de los problemas

Problema	**Categoría**
A - Dora la programadora	Adhoc, mapas, bucles, arrays.
B - Teclado bancario	Adhoc, mapas, bucles, condicionales.
C - 21	Programación dinámica.
D - Moléculas Atómicas	Programación dinámica.
E - Noche de cometas	Matemáticas, MCM, MCD.
F - Yincana	BackTracking.

Estadísticas

Problema	**# casos de prueba**	**Espacio en disco**
A - Dora la programadora	10	5MB
B - Teclado bancario	21	2MB
C - 21	6	2.3MB
D - Moléculas Atómicas	4	1MB
E - Noche de cometas	5	1.1MB
F - Yincana	33	8.3MB
- **Total**	79	19.7MB (+-)

Estadísticas*

Problema	Primer equipo en resolverlo	Tiempo
A - Dora la programadora	so.milagro.2018_CP	1'
B - Teclado bancario	j.justo.2017_CP	19'
C - 21	-	-
D - Moléculas Atómicas	-	-
E - Noche de cometas	x.liu1.2020_CP	36'
F - Yincana	-	-

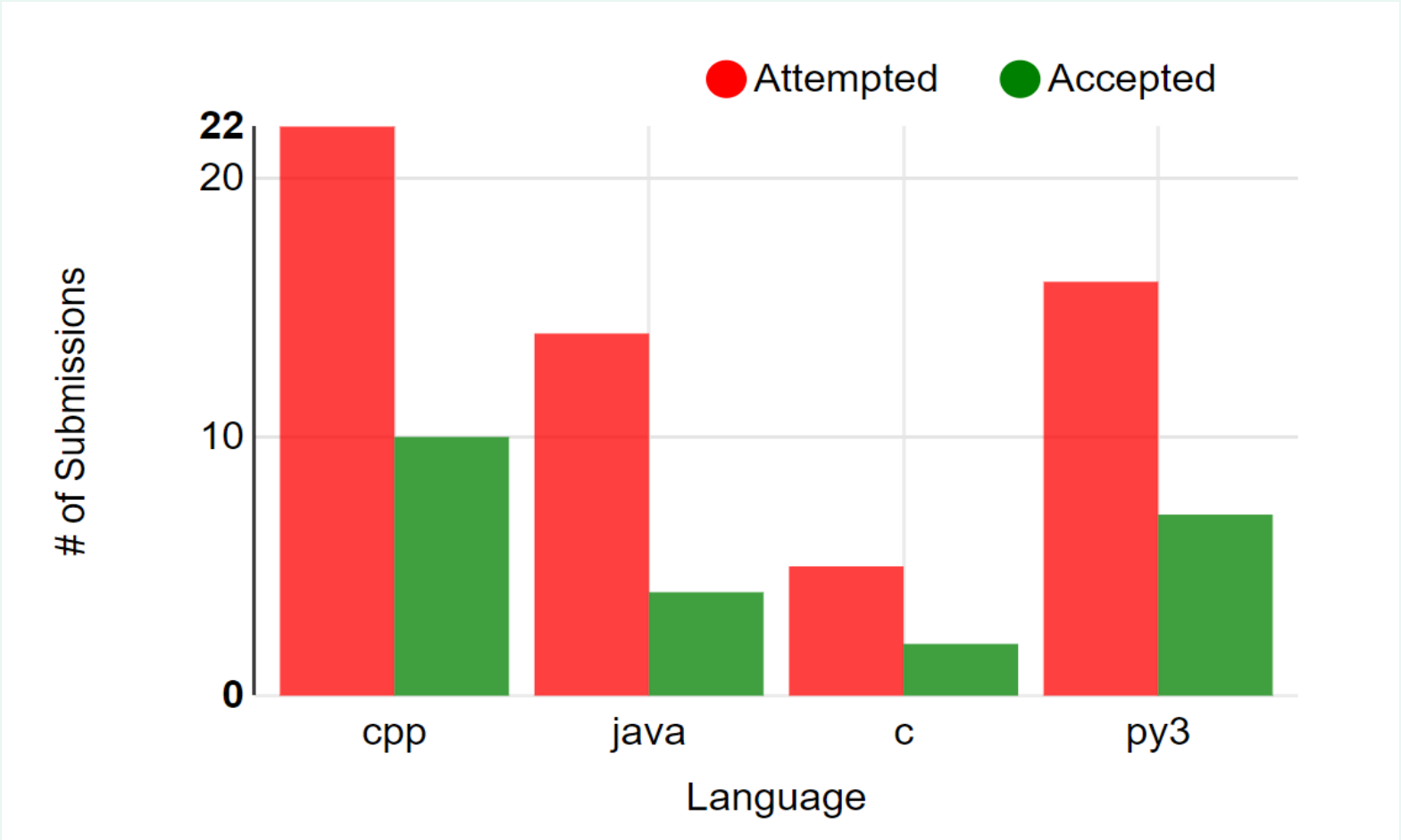
*Antes de congelar el marcador.

Estadísticas*

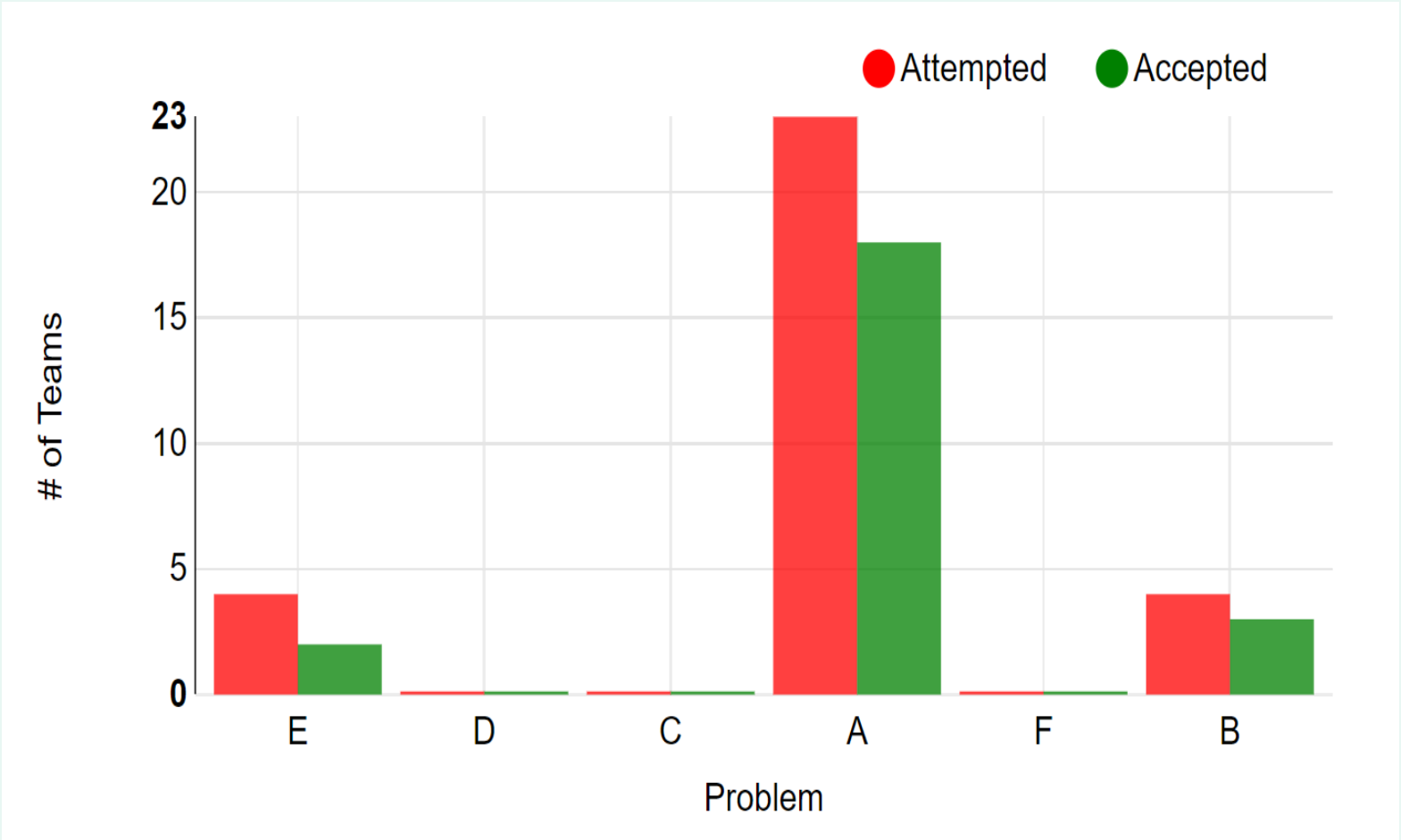
Problema	Envíos	Válidos	% éxito
A - Dora la programadora	37	17	46 %
B - Teclado bancario	4	3	75 %
C - 21	-	-	-
D - Moléculas Atómicas	-	-	-
E - Noche de cometas	8	2	25 %
F - Yincana	-	-	-

*Antes de congelar el marcador.

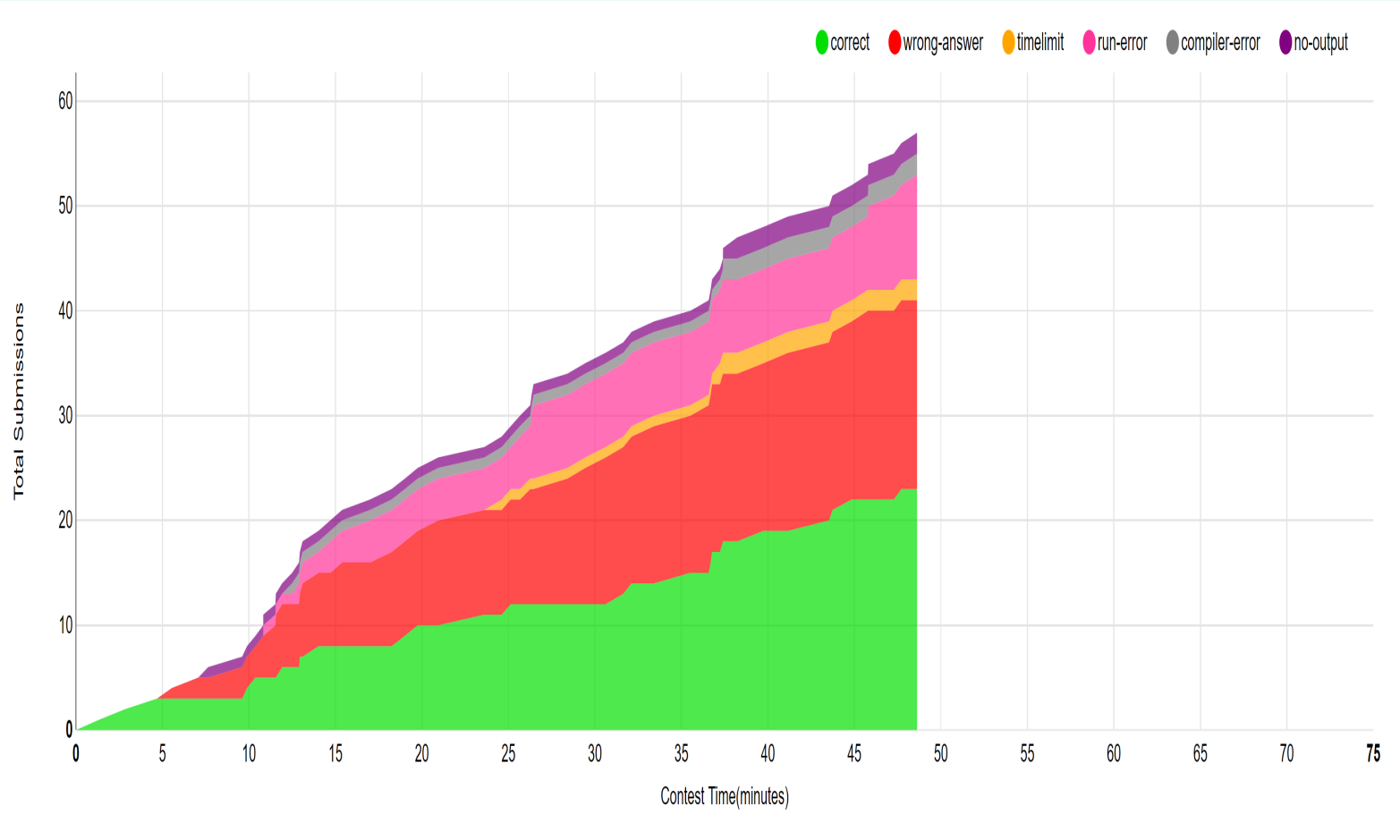
Estadísticas varias



Estadísticas varias



Estadísticas varias



A. Dora la programadora

Envíos	Válidos	% éxito
37	17	46 %

A. Dora la programadora

¿Histograma? ¿Qué estructura de datos?

- Array de 10 posiciones, dónde sumamos el número de aparición e imprimimos cuando sea necesario.
- Mapa, problema para practicar esta estructura de datos e imprimimos cuando sea necesario.

B. Teclado Bancario

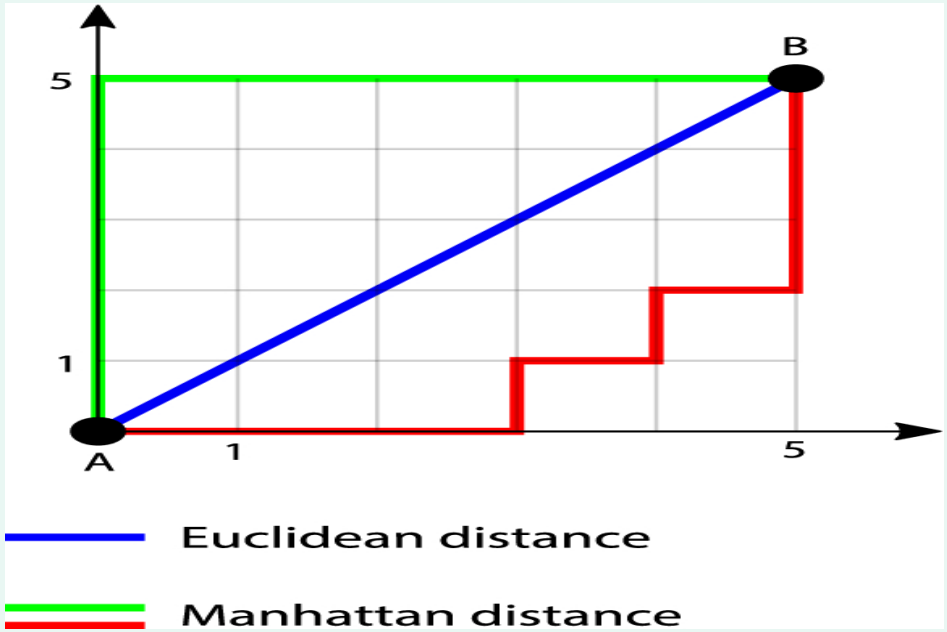
Envíos	Válidos	% éxito
4	3	75 %

B. Teclado Bancario

Una vez que tenemos el teclado, para ir de una letra a otra, necesitamos conocer la distancia de Manhattan.

¿Qué es la distancia de Manhattan?

$d(M,P)|M_x - P_x| + |M_y - P_y|$



B. Teclado Bancario

De tal forma que si desde cada sitio del teclado en el que estamos, hacemos la distancia de Manhattan hacia la siguiente letra, tenemos el recorrido y multiplicando por 100 el tiempo final.

Para determinar el sitio, nos guardamos un mapa con las letras determinando la posición en la que se encuentra y usando el mapa tenemos la siguiente posición del teclado.

E. Cometas

Envíos	Válidos	% éxito
8	2	25 %

E. Cometas

El mínimo común múltiplo (mcm) es el número positivo más pequeño que es múltiplo de dos o más números.
El máximo común divisor (m.c.d. o mcd) de dos o más números es el mayor número que divide a todos exactamente.
Si queremos saber cuando coinciden todos los números ¿qué necesitamos?
Aunque tiene un pequeño truco...

E. Cometas

Importante el long long!!

https://isaaclo97.github.io/resources/codes/GCDLCM.html

```
long long int gcd(long long int a, long long int b) {
    while (b > 0) {
        long long int temp = b;
        b = a % b;
        a = temp;
    }
    return a;
}
long long int lcm(long long int a, long long int b){
    return a*(b/gcd(a,b));
}
```

C. 21

Envíos	Válidos	% éxito
-	-	-

C. 21

Nos piden decir cuantas partidas posibles se pueden jugar de tal manera que nosotros seamos los ganadores (al completar la cuenta correctamente) en una partida en la que fuimos también los primeros en cantar.

Como una idea inicial ese problema se puede plantear de manera recursiva. Una función que recibe el número actual, la persona a la que le toca y la longitud de la cadena que ha dicho el último jugador.

Esa función comprueba si se ha completado la cuenta: si fue así y además lo ha hecho el primer jugador, devuelve un 1 y si ha sido otro jugador o se ha sobrepasado el límite se devuelve un 0.

En caso de no completarse la cuenta se llama recursivamente a si misma con dos llamadas. Con las dos longitudes que no ha usado el jugador anterior.

C. 21

Pero la aproximación recursiva puede llevar a exceder el límite de tiempo, ya que crece rápidamente con la longitud de la palabra y repite muchas veces cálculos que ya ha realizado.
Esto se puede remediar usando técnicas de programación dinámica para almacenar los cálculos ya realizados y no repetirlos.
La memoria que usaríamos para resolver este problema tiene los mismos parámetros que los parámetros de la función recursiva descrita anteriormente.
memoria[numeroActual][personaActual][longitudAnteriorCadena] que en el peor caso se corresponden con [20000][10][3].

D. Moleculas Atómicas

Envíos	Válidos	% éxito
-	-	-

D. Moleculas Atómicas

En este problema se nos pide maximizar la separación de una molécula en dos teniendo en cuenta la energía que producen teniendo en cuenta las siguientes condiciones:

- La separación ocurre al menos con dos moléculas; una de tamaño uno y otra de tamaño N-1 donde N es el número de moléculas encadenadas
- La energía generada está dada por la entrada y *pueden* producir dos energías, su suma, es la función objetivo que queremos maximizar

D. Moleculas Atómicas

Este problema entonces cumple las características básicas de un algoritmo de programación dinámica, por lo que procedemos a definir las metas básicas del mismo:

- Una molécula de tamaño 0 generará 0 energía
- Una molécula de tamaño 1 es inseparable y genera la energía que se le asigna
- De resto, comparar la respuesta que tenemos actualmente con dividir la cadena en un punto i, comprendido en un tamaño de 1 hasta i (¡inclusive!)

F. Yincana

Envíos	Válidos	% éxito
-	-	-

F. Yincana

Se tienen actividades con instante de inicio, de fin, y un beneficio obtenido por realizar dicha actividad. Las actividades son compatibles entre sí si no se solapan entre sí, teniendo en cuenta que actividades una actividad puede terminar a una hora y la siguiente empezar a dicha hora.

El programa debe dar el número de actividades realizadas y el beneficio obtenido.

Problema de las actividades ponderadas -> BackTracking.
Optimización posible de ramifica y poda.