

CURSO DE PROGRAMACIÓN COMPETITIVA

URJC - 2025



CURSO DE PROGRAMACIÓN COMPETITIVA

URJC - 2025

Organizadores:

- Isaac Lozano (isaac.lozano@urjc.es)
- Sergio Salazar (sergio.salazar@urjc.es)
- Adaya Ruiz (am.ruiz.2020@alumnos.urjc.es)
- Eva Gómez (e.gomezf.2020@alumnos.urjc.es)
- Lucas Martín (lucas.martin@urjc.es)
- Iván Penedo (ivan.penedo@urjc.es)
- Alicia Pina (alicia.pina@urjc.es)
- Sara García (sara.garcia@urjc.es)
- Raúl Fauste (r.fauste.2020@alumnos.urjc.es)
- Alejandro Mayoral (a.mayoralg.2020@alumnos.urjc.es)
- David Orna (de.orna.2020@alumnos.urjc.es)



CURSO DE PROGRAMACIÓN COMPETITIVA

URJC - 2025

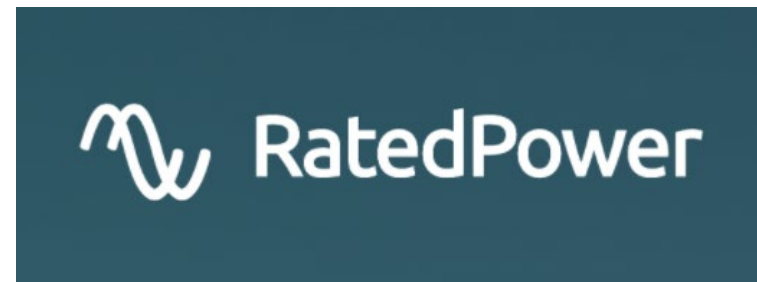
¿Quién somos?



CURSO DE PROGRAMACIÓN COMPETITIVA

URJC - 2025

¿En que trabajamos?



Motivación

- ¿Qué aprenderás?
 - diseño de algoritmos
 - estructuras de datos
 - nociones de complejidad
 - ...aprobar asignaturas!!! (ED, EDA, IP, POO, LP, DAA, P1, P2, AJ, ...)



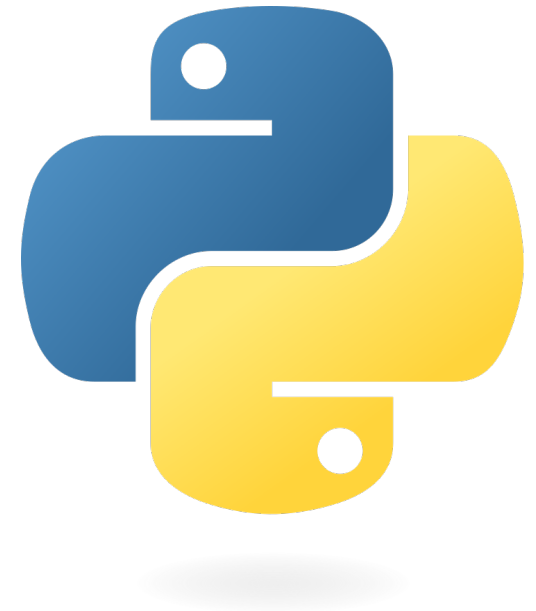
OBJETIVOS EN PROGRAMACIÓN COMPETITIVA

- Resolver los problemas en el menor tiempo posible
- Tener nociones intuitivas:
 - Tipos de problemas, algoritmos...
 - **Complejidad** vs límite de tiempo (eficiencia)
 - Estructuras de datos necesarias
- **Trabajo en equipo** (nombres creativos)
- Representar tu institución, país...



Motivación

- Empresas patrocinadoras
 - cazatalentos
 - concursos internos
 - entrevistas de trabajo



¿EMPRESAS QUE TRABAJÓ/A GENTE QUE COMPLETÓ EL CURSO?

amazon


HUAWEI

accenture

IBM

 **Eleven
Paths**


avanade

Deloitte.

Telefonica

 **vodafone**

gm[®]
INNOVATING SOLUTIONS



Motivación

- Participación en concursos
 - SWERC '25 en ?????: ? Equipos **NO TENEMOS EQUIPOS**
<https://urjc-cp.github.io/urjc-cp/swerc.html>
 - AdaByron '25 Madrid ¿?
<https://ada-byron.es/2025/reg/madrid/>
 - AdaByron Nacional – 4 y 5 de julio
<https://ada-byron.es/2025/nac/>
 - 12 Uvas -> 31/12
<https://las12uvas.es/>



CURSO DE PROGRAMACIÓN COMPETITIVA

URJC - 2023

Curso:

- **9 sesiones**
- 7 de febrero - 4 de abril (incluidos)
- Viernes: 17:00 - 19:00
- Aula 111 - Laboratorios 3



PLANIFICACIÓN DEL CURSO

- **Sesiones teórica y práctica.**
- En la teórica, los docentes explicarán los algoritmos y resolveremos problemas.
- En la práctica, tendremos un juez y se hará un **concurso de prueba** con enunciados cortos para fortalecer lo dado la semana anterior.



PLANIFICACIÓN DEL CURSO

- Bloque 1: Introducción (07/02)
- Bloque 2: Estructuras de Datos (14/02, 21/02*)
- Bloque 3: Estructuras de datos II (28/02)
- Bloque 4: Algoritmos de búsqueda y voraces (07/03)
- Bloque 5: Grafos (14/03, 21/03*)
- Bloque 6: Programación Dinámica (28/03)
- Concurso Final (04/04*)

Los concursos tendrán premios relacionados con merchand de la universidad y libros de programación competitiva. No sabemos cantidad pero aseguramos mínimo 1 premio no acumulativo (si alguien ganó, pasará al siguiente).

* concurso por equipos



CANALES Y FORMULARIO DE EQUIPOS



CP-2025



Dijkstraideos



Formulario



NO TENGO EQUIPO!!

Los concursos oficales FUERZAN que sean **3 personas**, en caso de ser menos o más no permiten participar.

Usad la clase para buscar otros compañeros.
Usad los grupos.

Preguntadnos, pero ser 3 para poder participar en cualquier evento.



¿YMIS CRÉDITOS?

- 9 sesiones - asistir a MÍNIMO 7 de ellas = CRÉDITOS
- Concursos (AdaByron, SWERC) = CRÉDITOS



¿CÓMO SE VALIDA ASISTENCIA?

- **Sesiones teóricas:** Aplicación de la URJC de Asistencia en un momento aleatorio de la clase.
- **Sesiones prácticas:** Mínimo de 1 envío por persona/equipo revisando IP del envío + Aplicación URJC en un momento aleatorio.
- No rellenar el formulario = NO ASISTENCIA



¿Podemos preguntar cualquier duda?

- Sí y no..



TIPOS DE COMPETICIONES

[ADA BYRON](#)



- ACM-ICPC:
 - 5 horas de duración
 - Equipos: 3 personas (1 ordenador)
 - Puntuación: problemas resueltos (0/1)
 - Empates: tiempo + penalizaciones




TIPOS DE COMPETICIONES

ICPC Southwestern Europe Regional Contest (SWERC) 2024

final standings

Filter ▾

RANK	TEAM	SCORE	A	B	C	D	E	F	G	H	I	J	K	L	M
1	  Xepelin Ecole Polytechnique	10 1432		46 1 try		18 1 try	120 2 tries		35 1 try	6 1 try	10 1 try	251 2 tries	298 9 tries	295 8 tries	13 1 try
2	  flag[10] Università di Pisa	9 839		45 1 try	296 2 tries	21 1 try	116 2 tries		25 1 try	5 1 try	29 1 try		6 tries	192 4 tries	10 1 try
3	  UXT Ecole Polytechnique	9 905		51 1 try	241 2 tries	29 1 try	144 1 try	244 2 tries	56 3 tries	4 1 try	14 1 try	2 tries	2 tries		22 2 tries
4	  morETHanusual ETH Zürich	9 1046	4 tries	196 3 tries	212 2 tries	26 1 try	256 2 tries		12 1 try	14 1 try	34 1 try			209 1 try	7 1 try
5	  TempName Tel Aviv University	9 1164	286 5 tries	94 2 tries	3 tries	33 2 tries	180 3 tries	232 3 tries	31 3 tries	17 1 try	45 1 try	1 try	1 try	3 tries	6 1 try
6	  Error-Prone Function Lovers Ecole Polytechnique Fédérale de Lausanne	8 691		43 1 try	242 8 tries	26 1 try	160 2 tries		31 1 try	4 1 try	17 1 try			1 try	8 1 try
7	  UPC-1 Universitat Politècnica de Catalunya	8 740		41 1 try	9 tries	16 1 try	256 7 tries	204 4 tries	25 1 try	3 1 try	10 1 try				5 1 try
8	  forETHought ETH Zürich	8 817		192 3 tries	163 4 tries	39 2 tries	202 1 try		21 1 try	6 1 try	31 1 try		1 try	3 tries	23 2 tries
9	  ENS Ulm 1 École Normale Supérieure de Paris	8 855		65 2 tries	1 try	35 1 try	274 6 tries	1 try	33 1 try	8 1 try	16 1 try	254 3 tries			10 1 try
10	  Schwebler's Peacocks The Open University of Israel	8 871		186 4 tries	11 tries	26 1 try	249 4 tries		22 1 try	11 1 try	9 1 try			195 3 tries	13 1 try

<https://swerc.vps.tecnico.ulisboa.pt/domjudge/public>



TIPOS DE COMPETICIONES

- ACM-ICPC (Proceso de selección)
 - Eliminatorias en la **universidad** si hay más de tres equipos
 - Eliminatorias en el conjunto de **países** que forman una región (South-Western Europe)
 - Eliminatorias entre los potenciales candidatos en todo el **continente** (Super regional europeo (Beta))
 - **Final Mundial**



TIPOS DE COMPETICIONES

- [Codeforces](#) y [Topcoder](#)
 - Concursos muy rápidos y frecuentes
 - Libre para cualquiera
 - Tres o cuatro **divisiones** para novatos y expertos
 - De 95 a 120 minutos de duración
 - Puedes ver y ‘romper’ el código de otros
 - Sistema de puntuación (mientras más tardes en resolver problemas, más te penalizan en puntos)



TIPOS DE COMPETICIONES

- Facebook Hacker Cup y [Google Code Jam](#)
 - Evento de gente masiva online
 - Al menos 4 rondas
 - Suele haber ronda de clasificación, 2 rondas de filtro y luego la fase final
 - Dos tipos de evaluación (small y large)
 - El caso small se corrige automáticamente
 - El caso large se corrige al terminar la competición
 - Se permite cualquier tipo de solución (incluso manual ó *hardcodeada*) que permita llegar al output



TIPOS DE COMPETICIONES

- USACO/COCI/IOI
 - Concursos dirigidos a alumnos de bachiller/secundaria
 - ¡NO SON TAN FÁCILES!
 - Son evaluados con sistemas de puntuación (no binario ni penalizando tiempo de solución)
 - Resultados después de la competición
 - Funcionan por temporadas (de noviembre a abril) por ser eliminatorias para el IOI (International Olympiads in Informatics)



KATTIS



KATTIS

- Log in / Registro



- Encontrar problemas

- Ranking por dificultad de [Kattis](#)
- Propuestos por nosotros

5.5 Hard

4.2 Medium

1.7 Easy

- IDE

- Kattis directamente
- Pycharm o IDE externo -> RECOMENDADO



CARACTERÍSTICAS DE UN PROBLEMA

Enunciado: Se explica el problema con una narración que lo justifica

Análisis del Problema: Se requiere una solución determinista para el problema (siempre encontraremos una solución óptima y válida)

Entrada: Se especifica lo que nuestro programa debe leer

Salida: Se especifica lo que nuestro programa debe mostrar

Ejemplos I/O: Muestras de entrada/salida con el comportamiento esperado para el programa

Límites [Opcionales]: Lo máximo ó mínimo en cuanto a variables que nuestro programa debe tomar en cuenta

[Problema de ejemplo](#)




CARACTERÍSTICAS DE UN PROBLEMA

- Tipos de Lectura:
 - Un caso: Se lee un caso de prueba y a partir de la entrada se genera una salida y termina la ejecución
 - Múltiples casos: Se leen varios casos de pruebas y, dadas múltiples entradas, se generan múltiples salida
- No hace falta guardar todos los resultados y mostrarlos al final
- ¡Cuidado con reutilizar estructuras de datos!



P YCHARM

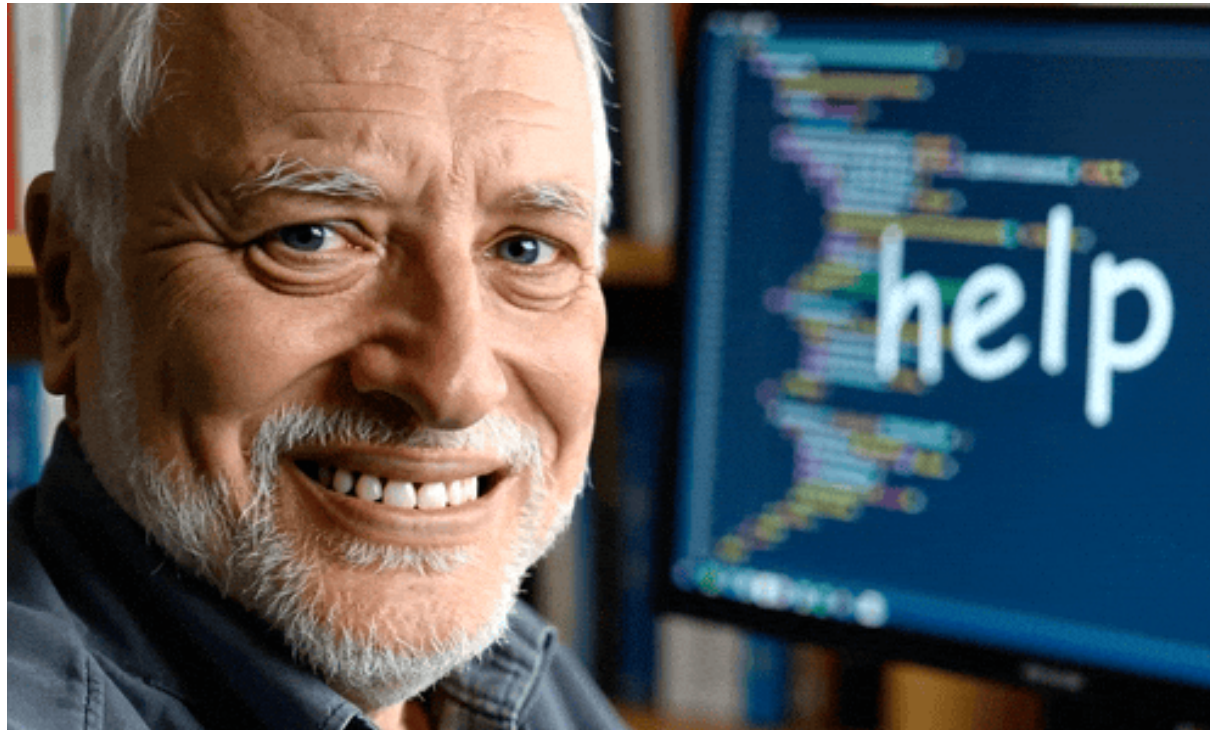
- Crear un Proyecto
- Empezar un código
 - main + TAB
- Lectura de 1 solo caso



```
if __name__ == '__main__':  
    name = input()  
    age = int(input())
```



CODING TIME!



<https://open.kattis.com/problems/hipphippurra>



CARACTERÍSTICAS DE UN PROBLEMA: LECTURA DE T CASOS

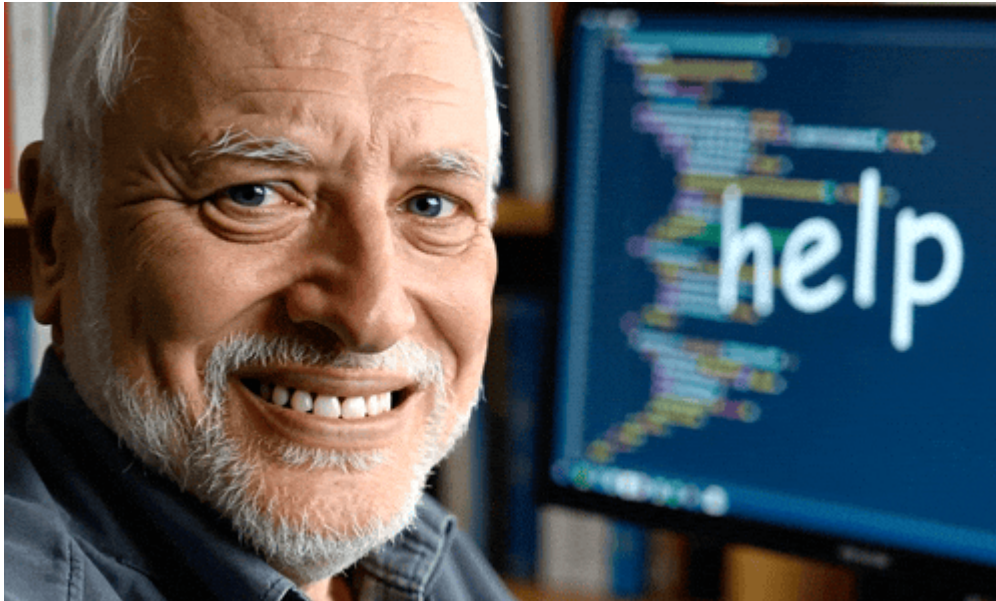
Se recibe un entero T y luego vendrán T casos de prueba

```
n = int(input())  
for i in range(n):  
    #- leer datos de cada caso  
    #- codigo + generar salida
```

Problema de ejemplo



CODING TIME!



```
n = int(input())
for i in range(n):
    #- leer datos de cada caso
    #- codigo + generar salida
```

<https://open.kattis.com/problems/oddities>



CARACTERÍSTICAS DE UN PROBLEMA: LECTURA HASTA EOF

Se leen los casos hasta leer la marca EOF (End-Of-File)

```
import sys

for line in sys.stdin:
    #codigo
```

- `line.split()` para separar por espacios y acceder con `map(int,line.split())` o `[posicion]`
- `line.strip()` para quitar el salto de línea

[Problema de ejemplo](#)



CARACTERÍSTICAS DE UN PROBLEMA: LECTURA HASTA CASO EN 0

Se lee el número de casos hasta que se consiga una condición de parada (generalmente cuando la entrada sea 0)

```
import sys

for line in sys.stdin:
    if(line=='0\n'):
        sys.exit()
    #codigo
```

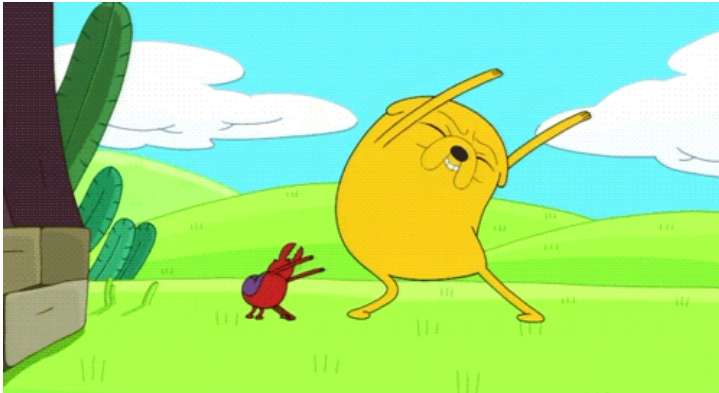
Problema de ejemplo

PYTHON



CARACTERÍSTICAS DE UN PROBLEMA: HE ESCRITO MI SOLUCIÓN ¿Y AHORA QUÉ?

AC



Tu solución es al menos tan buena como la esperada

- Imprimes correctamente todos los casos ocultos
- Tu código tarda menos en ejecutarse que el tiempo límite
- Tu código termina de ejecutarse sin problema

CARACTERÍSTICAS DE UN PROBLEMA: HE ESCRITO MI SOLUCIÓN ¿Y AHORA QUÉ?

WA

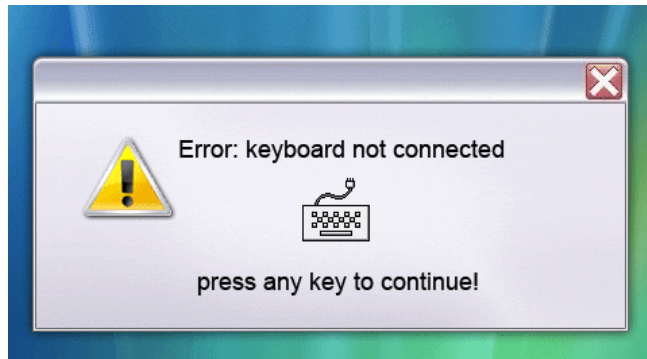
WRONG ANSWER

La solución que imprimes en algún caso no coincide con la esperada



CARACTERÍSTICAS DE UN PROBLEMA: HE ESCRITO MI SOLUCIÓN ¿Y AHORA QUÉ?

RTE



RUN TIME ERROR

Tu código muere por algún fallo en tiempo de ejecución

- Dividir entre cero
- Acceder a zonas de memoria no reservada
- Usar objetos en "null"
- Utilizar librerías externas (Python)

CARACTERÍSTICAS DE UN PROBLEMA: HE ESCRITO MI SOLUCIÓN ¿Y AHORA QUÉ?

TLE



TIME LIMIT EXCEPTION

Tu código tarda demasiado en ejecutarse

- Hay que reducir la complejidad del algoritmo!!!
- Puede esconder otros veredictos
- Seguramente no es la manera adecuada de resolverlo

CARACTERÍSTICAS DE UN PROBLEMA: HE ESCRITO MI SOLUCIÓN ¿Y AHORA QUÉ?

MLE



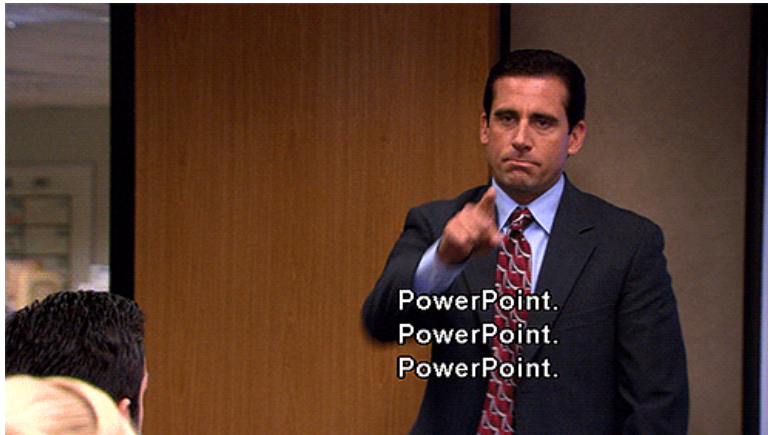
MEMORY LIMIT EXCEPTION

Tu código utiliza más memoria de la reservada

- Hay que reducir la memoria del algoritmo!!!
- Puede esconder otros veredictos

CARACTERÍSTICAS DE UN PROBLEMA: HE ESCRITO MI SOLUCIÓN ¿Y AHORA QUÉ?

PE



PRESENTATION ERROR

Tu código está bien, pero no lo has impreso de la manera correcta

- No siempre se contempla, en algunas ocasiones devolverá un WA

¿ YSI NO LO CONSIGO?

- **SIGUE PROBANDO!!**

Un mal veredicto no es excusa para rendirse, SIGUE!!!

- **INTERNET**

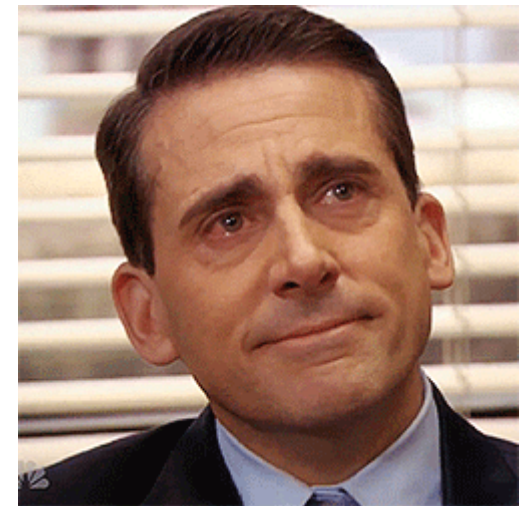
No lo tienes durante los concursos, pero entrenando no te quedes con la duda

- **CP3 y CP4**

La biblia de la programación competitiva (hay copias en la biblioteca)

- **Pide ayuda**

Usad el grupo de telegram para pedir ayuda



- [Ideone](#)
- [Diffchecker](#)



¿Podemos tener todos los casos de prueba?

¿En una empresa te dan casos de prueba para encontrar un fallo?

¿En un examen te los dan?

Si se necesitan tras mucho esfuerzo se os pasarán para aprender, pero aprender a encontrar fallos es algo muy importante.

Busca tus propios casos, pon los casos límites y prueba el algoritmo.




DOMjudge

- Plataforma utilizada en los concursos del curso
- Clarifications para preguntar
- Acceso por [correo](#)

DOMjudge

[Home](#) [Problemset](#) [Scoreboard](#) [Jury](#)

[Submit](#) [Logout](#) 4d 5:43:02

RANK	TEAM	SCORE	TORNEO
2	 e.gomezf Universidad Rey Juan Carlos	0 0	

Submissions

No submissions

Clarifications

No clarifications.

Clarification Requests

No clarification request.

request clarification



Entrenamiento

- Páginas de entrenamiento.
 - [Kattis](#)
 - [Acepta el Reto](#)
 - [Codeforces](#)
 - [OnlineJudge](#) y [uHunt](#)
 - [Spoj](#)

Recomendable tener un nick



Enlaces del curso

- El enlace del curso

<https://urjc-cp.github.io/urjc-cp/>

¿QUÉ AULA ERA?



- Enlace de otras ediciones (algunas grabadas)

<https://david8k.github.io/>

Este año el curso NO se grabará.



Tengo mucho interés, ¿cómo avanzar?

1. Practicar lo máximo posible
 - [Problemas propuestos](#) de Kattis (concurso semanal).
 - Concursos de [Codeforces](#) (semanales).
2. **Revisar material de otros años** de las siguientes clases y practicar
3. Leer, entender y practicar cada tema del **libro Competitive Programming 3 o 4** (disponible en biblioteca).

Quien haga estos pasos durante varios años asegura medalla en el Europeo.



¿Preguntas?



HASTA LA SEMANA QUE VIENE!



CP-2025



@Dijkstraídos

