

# Programación Competitiva

Por Fabián Olivares



folivares13@alumnos.usalca.cl



@Jestter

# ¿Qué hacemos acá?

- Resolver problemas que ya tienen solución
- Si, puede sonar extraño! Si ya están resueltos ¿para qué hacerlo de nuevo no?
- ...
- La diferencia es que estos problemas deben ser resueltos lo más rápido posible y en un ambiente de competencia (la persona que tiene la respuesta no se las dará).



# Conocimientos previos

- Saber programar en lo que sea: **C, C++, Java, Python, Kotlin**, javascript, Pascal, C#, Go, Haskell, Scala, Rust, Ruby... en serio, lo que sea (siempre y cuando el juez de la competencia acepte el lenguaje)
- Conocimiento de inglés nivel: puedo leer dos páginas sin estar usando Google translate en cada oración.

# Competitive Programming 3



[https://github.com/Jestter/Training\\_ICPC2018/blob/master/Recursos/Competitive.Programming.3rd.Edition.pdf](https://github.com/Jestter/Training_ICPC2018/blob/master/Recursos/Competitive.Programming.3rd.Edition.pdf)



# Plan de entrenamiento

- Temas:
  - Introducción (hoy)
  - Estructuras de datos y librerías
  - Búsqueda completa, dividir y conquistar, Greedy
  - Programación dinámica
  - Grafos
  - Matemáticas
  - Procesamiento de strings
  - Geometría computacional
  - Otros tópicos avanzados

# Plan de entrenamiento

- Competencias:

- Red de programación competitiva ([redprogramacioncompetitiva.com](http://redprogramacioncompetitiva.com))

- Calendario:

- Actividad 1: Febrero 10 de 2018
- Actividad 2: Marzo 10 de 2018
- Actividad 3: Abril 7 de 2018
- Actividad 4: Abril 28 de 2018
- Actividad 5: Mayo 12 de 2018
- Actividad 6: Junio 9 de 2018
- Actividad 7: Junio 23 de 2018

- Actividad 8: Julio 28 de 2018
- Actividad 9: Agosto 11 de 2018
- Actividad 10: Septiembre 1 de 2018
- Actividad 11: Septiembre 29 de 2018
- Actividad 12: Octubre 20 de 2018
- Actividad 13: Noviembre 3 de 2018
- Actividad 14: Noviembre 24 de 2018



# Tips

- Acostumbrarse a escribir código (rápido) y verificar en jueces.
- Identificar los problemas.
- Hacer el análisis del tiempo que toma en correr la solución (complejidad algorítmica) antes de empezar a codear.
- Tomar un par de lenguajes y aprenderlos bien (llegar al punto de tan solo necesitar la api para programar las soluciones).
- Acostumbrarse a hacer casos de prueba propios y debugging.
- Darse el tiempo de practicar de forma autónoma fuera de los entrenamientos.
- TRABAJO EN EQUIPO!

# Problemas de esta sesión - [a2oj.com](https://a2oj.com)

- Maximum in Table (<http://codeforces.com/problemset/problem/509/A>)
- Fence(<http://codeforces.com/problemset/problem/363/B>)
- Ilya and Sticks(<http://codeforces.com/problemset/problem/525/C>)
- Searching for  
nessy([https://uva.onlinejudge.org/index.php?option=onlinejudge&page=show\\_problem&problem=1985](https://uva.onlinejudge.org/index.php?option=onlinejudge&page=show_problem&problem=1985))
- Sales([https://uva.onlinejudge.org/index.php?option=com\\_onlinejudge&Itemid=8&page=show\\_problem&problem=3701](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=3701))
- Appleman and Easy Task(<http://codeforces.com/contest/462/problem/A>)
- Vito's  
family([https://uva.onlinejudge.org/index.php?option=onlinejudge&page=show\\_problem&problem=982](https://uva.onlinejudge.org/index.php?option=onlinejudge&page=show_problem&problem=982))
- Forming Quiz  
Teams([https://uva.onlinejudge.org/index.php?option=onlinejudge&page=show\\_problem&problem=1852](https://uva.onlinejudge.org/index.php?option=onlinejudge&page=show_problem&problem=1852))