

# CS3101: Programación Competitiva

Sesión Introductoria

Profesor:

Angel Gerardo Napa Bernuy  
[anapa@utec.edu.pe](mailto:anapa@utec.edu.pe)



**Autor:**

**Angel Napa**

1

# Información sobre el curso de Programación Competitiva

UTEC

## Sistema de Evaluación:

EVALUACIÓN	TEORÍA	PRÁCTICA Y/O LABORATORIO
	Examen 1 <b>E1 (10%)</b> Examen 2 <b>E2 (10%)</b>	Evaluación continua <b>C1 (40%)</b> Evaluación continua <b>C2 (40%)</b>
	20%	80%
	100%	

# Reglas para evaluaciones de Teoría

- C: evaluaciones continuas
- Semanalmente se realizarán contests
- 2 Exámenes escritos
- La nota mínima para aprobar el curso es 10.5

# 2

## Unidad 1: Introducción al curso

UTEC

## ¿Qué es programación competitiva?

- Es un deporte mental que combina 2 tópicos
- Diseños de algoritmo:
  - Algoritmos eficientes
  - Combinación de métodos conocidos de nuevos insights
- Implementación de algoritmos:
  - No es software tradicional
  - No es Hackathon
  - Programas cortos, deben ser escritos rápidos, no necesarios luego de la competencia
  - C++, Python, Java

## ¿Qué es programación competitiva?

- ICPC The International Collegiate Programming Contest
- IEEEExtreme
- International Olympiad of Informatics
- Online Contests: Codeforces, AtCoder, CodeChef, CS Academy, HackerRank, Topcoder, LeetCode
- Facebook Hacker Cup, Google Code Jam, Yandex Algorithm, GoogleKickStart





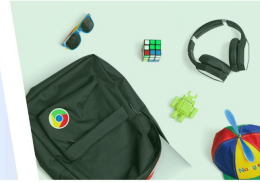




**code jam**



**hash code**



































**kick start**

























































































































































































































































*advancing the art and sport  
of competitive programming*



#	User/Site	Name	A	B	C	D	E	F	G	H	I	J	K	L	M	Total
1	teamsope007/PE	[UTEC] SamuraisDelUwu	 3/239		 1/182	 1/28	 3/72				 1/94		2/-	2/-	 1/153	6 (848)
2	teamsope001/PE	[UNI-FC] TLEnjoyers			 1/222	 1/13	 1/113				 2/152			1/-	 3/99	5 (659)
3	teamsope009/PE	[UNI-FIIS] ADHOCKERS				 1/18	 4/163				 1/214				8/-	3 (455)
4	teamsope015/PE	[UNSA-CC] OpenCubitos			 5/218	 1/17	 4/81				2/-					3 (456)
5	teamsope011/PE	[UCSP] Los galácticos	1/-			 1/12	9/-			2/-	 1/70	2/-				2 (82)
6	teamsope008/PE	[UPC] GPT is your turn				 1/29	4/-				 1/127					2 (156)
7	teamsope012/PE	[UTEC] O(1)				 1/75	1/-				 2/170					2 (265)
8	teamsope005/PE	[UPC] C0d3Br0th3rs				 1/22	5/-				 1/247				1/-	2 (269)
9	teamsope006/PE	[UPC] DigitalCodeX	1/-	1/-		 1/15	4/-				2/-					1 (15)
10	teamsope014/PE	[UTEC] 404				 1/31	4/-									1 (31)
11	teamsope010/PE	[ ] Deep Work				 1/34	5/-									1 (34)
12	teamsope013/PE	[ ] UNAP - AutoChess Coders				 1/40	3/-									1 (40)
13	teamsope003/PE	[UNSAAC] La banda Binaria				 1/52	3/-									1 (52)
14	teamsope004/PE	[UNSAAC] The competidor Jhajuset				 2/36	5/-									1 (56)
15	teamsope002/PE	[UNSAAC] MaQueenTosh				 3/145	2/-				2/-					1 (185)



#	User/Site	Name	A	B	C	D	E	F	G	H	I	J	K	L	M	Total
1	 teamsoar001/AR	[UBA - FCEN] una ma y no inchamo ma	 1/88	 1/197	 1/63	 1/8	 2/69	 1/268	 3/230	 1/161	 1/45	1/-	 1/148	 1/33	 2/95	12 (1485)
2	 teamsoar005/AR	[UBA - FCEN] Prim Floyd	 1/141	 2/236	 1/188	 1/18	 1/14			 2/270	 1/52		3/-	 2/123	 1/80	9 (1182)
3	 teamsoch014/CL	[PUC-Ing] Laranjas.clear()	 1/135	5/-	 1/186	 1/13	 3/57	1/-	 1/267	 2/209	 1/119		 1/227		 1/152	9 (1425)
4	 teamsoar020/AR	[UNR] Don Gato	 1/254		 1/180	 1/12	 1/107		 1/156	 1/277	 3/237				 1/218	8 (1481)
5	 teamsobo001/BO	[UMSS] Club de Frontón 2880	 1/143		 2/197	 1/15	 2/42				 1/88			 3/261	 1/118	7 (944)
6	 teamsoar002/AR	[FCE - UNLP] Estufa En Piloto	 6/226		 2/295	 1/16	 3/58				 1/85			 2/193	 1/198	7 (1251)
7	 teamsoch015/CL	[PUC-Ing] McNanoL	 1/190			 1/19	 6/195			 3/217	 4/128		 2/277		 5/128	7 (1454)
8	 teamsoch002/CL	[UCHile] Dijkstra Stan Army	 1/92			 1/12	 2/59			 3/283	 1/72				 1/123	6 (701)
9	 teamsope007/PE	[UTECH] SamuraisDelUwu	 3/239		 1/182	 1/28	 3/72				 1/94		2/-	2/-	 1/153	6 (848)
10	 teamsope001/PE	[UNI-FC] TLEnjoyers			 1/222	 1/13	 1/113				 2/152			1/-	 3/99	5 (659)
11	 teamsoar014/AR	[UBA - FCEN] Que la sigan mandando	1/-		 1/213	 1/14	 4/111				 2/124				 1/153	5 (695)
12	 teamsoar015/AR	[UTN Argentina - Santa Fe] Nanopartículas	 3/261		5/-	 1/22	 3/121				 2/113				 5/213	5 (910)
13	 teamsobo002/BO	[UMSA] Los Maquinolas	2/-			 1/12	 1/73			1/-	 1/45				 3/150	4 (320)
14	 teamsoar023/AR	[UN La Plata] Es Todo Un Tema			 3/226	 1/18	 1/54				 1/202				1/-	4 (540)
15	 teamsobo003/BO	[USB] Jala-Peños	 1/246			 1/21	 3/91				 1/174				5/-	4 (572)

#	User/Site	Name	A	B	C	D	E	F	G	H	I	J	K	L	M	Total
1	 teamsoar001/AR	[UBA - FCEN] una ma y no inchamo ma	 1/88	 1/197	 1/63	 1/8	 2/69	 1/268	 3/230	 1/161	 1/45	1/-	 1/148	 1/33	 2/95	12 (1485)
2	 teambrbr005/BR	[UFRJ] Lebenslangerschicksalsschatz	 1/117	 1/156	 1/48	 1/21	 1/29	 1/269		 2/206	 1/54		 1/295	 1/83	 1/133	11 (1431)
3	 teambrbr002/BR	[UFMG] Humuhumunukunukuapua'a	 2/127	 2/237	 1/99	 1/11	 2/49		 1/254	 1/176	 1/36	1/-	 1/191	 1/148	 2/106	11 (1514)
4	 teamnoco001/CO	[UNAL Bogotá] phiUN	 1/167	 1/270	 1/87	 1/19	 1/45			 2/209	 1/38		 1/191	 1/141	 1/152	10 (1339)
5	 teambrbr001/BR	[UNICAMP] Você beijaria Matheus Leal Viana?	 1/140	 1/122	 1/198	 1/6	 2/90	 1/83		 1/237	 1/101		 2/270	1/-	 1/59	10 (1346)
6	 teambrbr003/BR	[UNICAMP] Inimigos do Hungaro	 2/80	 1/260	 1/46	 1/17	 2/54	1/-		 1/104	 1/62		 3/141	9/-	 1/29	9 (873)
7	 teamsoar005/AR	[UBA - FCEN] Prim Floyd	 1/141	 2/236	 1/188	 1/18	 1/14			 2/270	 1/52		3/-	 2/123	 1/80	9 (1182)
8	 teamsoc014/CL	[PUC-Ing] Laranjas.clear()	 1/135	5/-	 1/186	 1/13	 3/57	1/-	 1/267	 2/209	 1/119		 1/227		 1/152	9 (1425)
9	 teamcbcu001/CU	[UH] UH Top	 1/149		 4/102	 1/17	 2/72			 5/278	 1/65		 2/259	 1/170	 1/153	9 (1445)
10	 teambrbr008/BR	[IME] 12k Club	 1/44		 1/86	 1/12	 1/19	2/-	 1/201	 1/166	 1/32	1/-			 1/61	8 (621)
11	 teambrbr011/BR	[UnB] FLAMENGO	 1/65		 2/140	 1/14	 1/37			 1/201	 1/79		1/-	 2/274	 1/107	8 (957)
12	 teamnoco002/CO	[EAFIT] Fast and Fourier	 3/125		 1/146	 1/15	 1/50			 1/249	 1/62		2/-	 1/212	 1/94	8 (993)
13	 teammxmx001/MX	[UG-CIMAT] OWO	 1/51	 1/205	 1/185	 1/8	 2/64	1/-		 5/277	 2/83				 1/108	8 (1101)

# ACM-ICPC

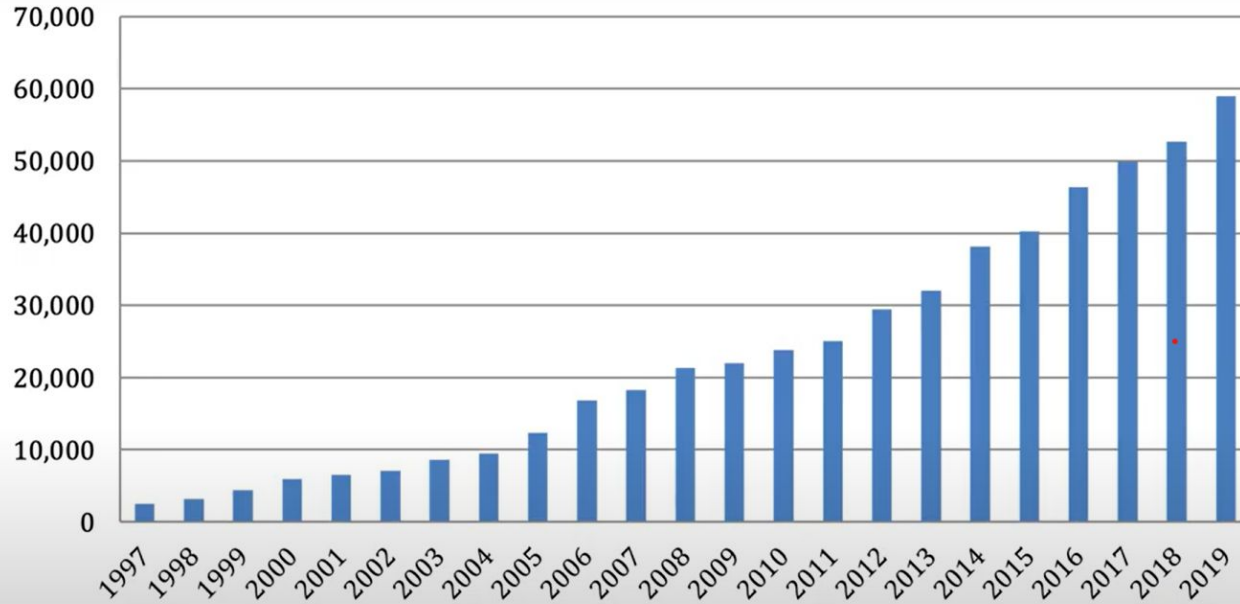
- Equipos de 3
- 5 horas de prueba
- ~10 problemas
- Dificultad variable y desconocida por cada problema
- **Live scoreboard:** Primeras 4 horas con información conocida del avance del resto de equipos
- **Frozen scoreboard:** Luego de la 5ta hora solo conoces el avance de tu equipo.



# Entrenamiento Grupal

- Simular un examen real
  - 5 horas
  - usar banco de problemas similar
  - 1 teclado
  - escribir código en papel.
  - “Imprimir” código

## Student Participation





# Meta

- Dado un problema, queremos:
  - Resolver de forma eficiente
  - Usando algoritmos y estructuras de datos
  - convertir nuestra solución en un programa
  - realizarlo de forma correcta
  - realizarlo lo más pronto posible
- Este curso se encargará de practicar este proceso

# ¿Cómo?

- Practicar diferentes tipos de problemas
- Resolver problemas clásicos que aplican algoritmos conocidos
- Practicar resolución de problemas
- Practicar programación
- Practice
- More practice

# Los problemas

- Un contest tiene un conjunto de problemas.
- La estructura de un problema contiene usualmente:
  - Descripción del problema
  - Descripción del Input
  - Descripción del Output
  - Ejemplos de Input/Output
  - Restricción de tiempo en segundos
  - Restricción de memoria en bytes

# Los problemas

- El problema consiste construir un programa que:
  - Resuelva el problema para una serie de válidos inputs
  - No exceda las restricciones de tiempo y memoria

# Veredictos de los Judges

- Feedback de las soluciones es limitada
- Usualmente uno recibe uno de los siguientes mensajes:
  - (AC) Accepted
  - (WA) Wrong Answer
  - (CE) Compile Error
  - (RTE) Run Time Error
  - (TLE) Time Limit Exceeded
  - (MLE) Memory Limit Exceeded
- En los contests no se revela el banco de inputs con el que trabaja el Judge



# Tips

- Typing: **touch type code**
- Invertir tiempo.
- Cuidar la salud
- Sé honesto contigo mismo(registrar tu progreso)
  - #problemas
  - #horas
- Escoger el lenguaje a usar:
  - Uno que te familiarices bien o ya estés familiarizado
  - Aprender otros
  - Lee códigos

# ¿Cómo Entrenar?

- Entrenamiento “Tradicional”:
  - Practica “lento”
  - Practica “rápido”
  - Practica “normal”

# Entrenamiento Individual

- Qué hacer
  - Online Judges
  - Online Competitions
  - Pregunta
- Qué no hacer
  - Quedarte en tu zona de confort
  - Evitar resolver problemas “difíciles” para ti
  - Olvidar

<https://cses.fi/problemset/>

[CSES - Weird Algorithm](#)

**Time limit:** 1.00 s   **Memory limit:** 512 MB

Consider an algorithm that takes as input a positive integer  $n$ . If  $n$  is even, the algorithm divides it by two, and if  $n$  is odd, the algorithm multiplies it by three and adds one. The algorithm repeats this, until  $n$  is one. For example, the sequence for  $n = 3$  is as follows:

$3 \rightarrow 10 \rightarrow 5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$

Your task is to simulate the execution of the algorithm for a given value of  $n$ .

### Input

The only input line contains an integer  $n$ .

### Output

Print a line that contains all values of  $n$  during the algorithm.

### Constraints

- $1 \leq n \leq 10^6$

<https://cses.fi/problemset/>

[CSES - Weird Algorithm](#)

## Example

Input:

3

Output:

3 10 5 16 8 4 2 1

```
#include <iostream>

using namespace std;

int main() {
    int n;
    cin >> n;
    while (true) {
        cout << n << " ";
        if (n == 1) break;
        if (n%2 == 0) n /= 2;
        else n = n*3+1;
    }
    cout << "\n";
}
```

# Veredicto 1

test	verdict	time (s)
#1	ACCEPTED	0.06 / 1.00
#2	ACCEPTED	0.06 / 1.00
#3	ACCEPTED	0.07 / 1.00
#4	ACCEPTED	0.06 / 1.00
#5	ACCEPTED	0.06 / 1.00
#6	TIME LIMIT EXCEEDED	– / 1.00
#7	TIME LIMIT EXCEEDED	– / 1.00
#8	WRONG ANSWER	0.07 / 1.00
#9	TIME LIMIT EXCEEDED	– / 1.00
#10	ACCEPTED	0.06 / 1.00

# Veredicto luego de arreglar el code

test	verdict	time (s)
#1	ACCEPTED	0.05 / 1.00
#2	ACCEPTED	0.06 / 1.00
#3	ACCEPTED	0.07 / 1.00
#4	ACCEPTED	0.06 / 1.00
#5	ACCEPTED	0.06 / 1.00
#6	ACCEPTED	0.05 / 1.00
#7	ACCEPTED	0.06 / 1.00
#8	ACCEPTED	0.05 / 1.00
#9	ACCEPTED	0.07 / 1.00
#10	ACCEPTED	0.06 / 1.00



# Template

```
1  #include <bits/stdc++.h>
2
3  using namespace std;
4
5  int main() {
6      ios_base::sync_with_stdio(false);
7      cin.tie(nullptr);
8
9      exit(0);
10 }
11
```

- UVA judge
  - <https://onlinejudge.org/>
- Codeforces
  - <https://codeforces.com/>
- Beecrowd:
  - <https://www.beecrowd.com.br/judge/>
- Atcoder:
  - <https://atcoder.jp/>
- CSES:
  - <https://cses.fi/problemset/>
- CodeChef:
  - <https://www.codechef.com/>
- entre otros...

# Bibliografía

- Antti Laaksonen. ***Guide to Competitive Programming: Learning and Improving Algorithms Through Contests***. Springer 2018
- Steven Halilm, Felix Halim. ***Competitive Programming 3: The New Lower Bound of Programming Contests***, Volumen 3. Lulu.com, 2013
- Steven S. Skiena, Miguel A. Revilla. ***Programming Challenges: The Programming Contest Training Manual***. Springer Science & Business Media, 2006
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. (2009). ***Introduction to algorithms***. MIT press.

**¡Nos vemos en la  
siguiente clase!**

