

CURSO DE PROGRAMACIÓN COMPETITIVA

URJC - 2020



CURSO DE PROGRAMACIÓN COMPETITIVA

URJC - 2020

Al final se deberá firmar en papel

CURSO DE PROGRAMACIÓN COMPETITIVA

URJC - 2020

Curso:

- 11 sesiones
- 07 de enero - 24 abril (inclusive)
- Viernes: 17:00 - 19:00
- Aulario 1 - 002 (Teoría) y Laboratorio 1 - 109 (Práctica)

Organizadores:

- David Morán (david.moran@urjc.es)
- Sergio Pérez (sergio.perez.pelo@urjc.es)
- Juan Quintana (juandavid.quintana@urjc.es)
- Jesús Sánchez-Oro (jesus.sanchezoro@urjc.es)
- Isaac Lozano (isaac.lozano@urjc.es)
- Raúl Martín (raul.martin@urjc.es)

CURSO DE PROGRAMACIÓN COMPETITIVA

URJC - 2020

Fechas importantes (revisar PDF de horario):

- Ada Byron local: 12 de febrero
- Google Hashcode: 20 de febrero
- Ada Byron Regional: viernes 28 y 29 de febrero
- SWERC 2020 (diciembre-enero?)
 - Clasificación equipos URJC el ??/??/??

Motivación

- Programming skills
 - diseño de algoritmos
 - estructuras de datos
 - nociones de complejidad
 - ...aprobar asignaturas!!! (ED, EDA, DAA, ...)

Motivación

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

Motivación

- Participación en concursos
 - SWERC '20 en ????: ? equipos
 - AdaByron
 - Google Hashcode
 - 12 Uvas
 - ... premios y honor!!!



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...

PLANIFICACIÓN DEL CURSO

- Cada bloque se dividirá en dos sesiones; una teórica y una práctica
- En la teórica, los docentes explicarán los algoritmos y los diferentes usos, así como resolver algún problema de ejemplo tipo
- En la práctica, se levantará un juez y se hará un concurso de prueba con enunciados cortos para fortalecer lo dado la semana anterior

PLANIFICACIÓN DEL CURSO

Objetivos:

- Que los alumnos se sientan cómodos en un ambiente competitivo
- Que se familiaricen con los jueces (y a ser posible con sistemas UNIX)
- Que entrenen de manera dinámica sus habilidades de programación

PLANIFICACIÓN DEL CURSO

- Bloque 1: Introducción (07/02)
- Bloque 2: Estructuras de Datos (14/02, 28/02)
- Bloque 3: Algoritmos de búsqueda y voraces (06/03, 13/03)
- Bloque 4: Grafos (20/03, 27/03)
- Bloque 5: Grafos ponderados (03/04, 10/04)
- Bloque 6: Programación Dinámica (17/04)
- Concurso Final (24/04)


TIPOS DE COMPETICIONES

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

TIPOS DE COMPETICIONES

SWERC 2019-2020

final standings

RANK	TEAM	SCORE	A ●	B ●	C ●	D ●	E ●	F ●	G ●	H ●	I ●	J ●	K ●	L ●
1	 EP Chopper École Polytechnique	11 1170	103 1 try	4 1 try	11 1 try	276 1 try		21 1 try	150 2 tries	233 2 tries	6 1 try	112 2 tries	61 1 try	133 1 try
2	 UPC-1 Universitat Politècnica de Catalunya	11 1289	133 3 tries	6 1 try	9 1 try	165 1 try	293 3 tries	15 1 try	145 5 tries		4 1 try	77 1 try	39 2 tries	223 1 try
3	 mETH ETH Zürich	10 1176	183 3 tries	10 1 try	20 1 try	161 1 try		32 2 tries	229 3 tries		4 1 try	97 1 try	76 1 try	264 1 try
4	 EP Rouje École Polytechnique	10 1302	88 4 tries	6 1 try	12 1 try	220 2 tries		26 2 tries		252 2 tries	3 1 try	117 1 try	141 2 tries	297 1 try
5	 ENS Ulm 1 École Normale Supérieure de Paris	10 1317	182 3 tries	20 1 try	24 1 try	220 1 try		38 3 tries		116 1 try	10 1 try	48 1 try	282 2 tries	277 1 try
6	 LaStatale Blue Università Degli Studi di Milano	9 1076	157 4 tries	10 1 try	18 1 try	261 2 tries		30 1 try		78 1 try	12 1 try	256 1 try	174 1 try	
7	 ETHanol ETH Zürich	9 1150	222 4 tries	10 1 try	16 1 try	239 1 try		58 3 tries	270 1 try		5 1 try	119 2 tries	91 1 try	
8	 ISTo é por vocês Instituto Superior Técnico	8 953	149 1 try	15 2 tries	23 1 try			31 1 try	273 1 try		6 1 try	260 2 tries	136 2 tries	1 try
9	 Moradonellani Politecnico di Milano	8 1060	193 2 tries	7 1 try	15 1 try			38 2 tries	294 3 tries		10 1 try	269 2 tries	94 3 tries	
10	 UPC-2 Universitat Politècnica de Catalunya	8 1071	257 2 tries	8 1 try	14 1 try	8 tries		21 2 tries	269 3 tries		4 1 try	189 3 tries	129 4 tries	

<https://swerc.eu/2019/theme/scoreboard/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
 - Dos o tres 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)

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

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 salidas
- ¡Cuidado con reutilizar estructuras de datos!
- No hace falta guardar todos los resultados y mostrarlos al final

CARACTERÍSTICAS DE UN PROBLEMA: LECTURA DE T CASOS

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

```
Scanner sc = new Scanner(System.in);
int t = sc.nextInt();
for(int i=0; i<t; i++) {
    //- leer datos de cada caso
    //- codigo + generar salida
}
```

```
int t;
scanf("%d", &t);
for(int i=0; i<t; i++){
    // - leer datos de cada caso
    // - codigo + generar salida
}
```

CARACTERÍSTICAS DE UN PROBLEMA: LECTURA HASTA EOF

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

```
Scanner sc = new Scanner(System.in);  
while(sc.hasNextInt()) {  
    int n = sc.nextInt();  
    // codigo  
}
```

```
int n;  
while(scanf("%d", &n) != EOF){  
    //codigo  
}
```

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)

```
Scanner sc = new Scanner(System.in);
int n = sc.nextInt();
while(n!=0) {
    //codigo
    n = sc.nextInt();
}
```

```
int n;
while(scanf("%d", &n) != EOF && n!=0){
    //codigo
}
```

LENGUAJES DE PROGRAMACIÓN

- C
- C++
- Java
- Python (muy nuevo)

Ejemplo Problema: ¡Hola mundo!

<https://www.aceptaelreto.com/problem/statement.php?id=116>

¡Hola mundo!

Tiempo máximo: 1,000 s



Memoria máxima: 2048 KiB

Dicen los viejos que en este país el latín era una asignatura obligatoria por la que todos los estudiantes de bachillerato tenían que pasar.

Dicen los viejos que el primer día de clase de latín cualquiera esperaba que los alumnos salieran sabiendo el "rosa rosae".

Dicen los viejos que esa era la primera declinación.

Quizá, dentro de muchos años, nosotros seamos los viejos que contemos batallitas de cómo se enseñaba informática. Y quizá entonces, digamos que en la primera clase de cualquier curso en el que se explicara un lenguaje de programación, se tenía que salir habiendo escrito "un hola mundo".

Y eso es lo que vamos a hacer. Escribir un programa que escriba tantos "hola mundo" como nos pidan.

Entrada

La entrada consta de una única línea que contiene un número n , $0 \leq n \leq 5$, que indica cuántos mensajes hay que emitir.

Salida

Cada mensaje a escribir aparecerá en una única línea y será la cadena "Hola mundo."

Entrada de ejemplo

3

Salida de ejemplo

Hola mundo.
Hola mundo.
Hola mundo.

Ejemplo Problema: Test

<https://www.spoj.com/problems/TEST/>

TEST - Life, the Universe, and Everything

#basic #tutorial #ad-hoc-1

Your program is to use the brute-force approach in order to *find the Answer to Life, the Universe, and Everything*. More precisely... rewrite small numbers from input to output. Stop processing input after reading in the number 42. All numbers at input are integers of one or two digits.

Example

Input:

1
2
88
42
99

Output:

1
2
88

Entrenamiento

- <https://docs.google.com/spreadsheets/d/1kbepsA3yIQ9HpLq8WAOhSLf5QMZK0KRdKQ2pxEBCQkQ/edit?usp=sharing>
- Páginas de entrenamiento Acepta el Reto - Spoj

Semana que viene

- Día 12 Concurso Clasificatorio Local AdaByron - equipos de 3 personas (Aula por decidir, recibiréis correo el Domingo)
 - Inscripción: shorturl.at/bmtR8
- Viernes aula de teoría: Nociones básicas, estructuras de datos y complejidad en tiempo.

Hasta la vista

Baby



Telegram: shorturl.at/cFOUV

Noticias: shorturl.at/zBG27

Twitter: @URJC_CP