

Preparación para la ACM ICPC

Autores: Joaquín Planells Lerma, Sergio García Barea, Mario Rodríguez Molins

Introducción

La ACM ICPC [1] es una competición internacional de programación por equipos que se celebra todos los años en la que compiten estudiantes universitarios de todo el mundo. Se organizan fases regionales de clasificación (en nuestro caso la SWERC [2]) y los mejores equipos de cada región compiten en la fase final.

En la página de Salvador España [3] se puede leer una breve explicación sobre la competición y el resultado que obtuvo el equipo de la Facultad de Informática hace unos años.

Ediciones anteriores

Que nosotros sepamos la UPV sólo ha participado durante los últimos 3 años en la SWERC y estos son los resultados obtenidos:

- 2003-2004 : El equipo **UPV Valencia Penguins** consiguió resolver 1 problema y quedó en el puesto 22 de 52 equipos participantes. [4]
- 2004-2005: El equipo **FIUPV Valencia Penguins** acabó con 1 problema resuelto en el puesto 36 de 57 participantes. [5]
- 2005-2006: Participan 2 equipos de la Universidad, puesto 29 y 3 problemas resueltos para el **UPV-EI** de la Escuela de Informática y puesto 47 para **UPV-FI** de la Facultad de Informática. Participaban un total de 55 equipos. [6]

Preparación

La mejor manera de entrenar para esta competición es resolver muchos problemas del mismo tipo que los que ponen en el concurso. También es necesario conocer muy bien la sintaxis y el funcionamiento del lenguaje de programación que utilicemos (C, C++ o JAVA). La únicas librerías que se permiten son las estándar que acompañan a los compiladores. Es importante conocer los distintos tipos de problemas que se presentan y esto se consigue con la práctica. A menudo suelen aparecer problemas de los siguientes tipos:

- Matemáticos: Calculos numéricos
 - Aritmética con números grandes
 - Números primos
 - Aritmética modular
 - Teoría de números
 - Otros (Fibonacci, Goldbach...)
- Teoría de grafos
 - dfs, bfs
 - Búsqueda del camino mínimo
 - Árbol de recubrimiento mínimo
 - Network Flow
 - Ordenación topológica
- Geometría
 - Líneas, rectas, circunferencias... etc
 - Envoltura convexa
- Programación dinámica
- Algoritmos voraces

- Simulación: Problemas en los que simplemente hay que seguir una serie de reglas indicadas en el enunciado. Ej: Dirigir un robot por un mapa siguiendo las órdenes escritas en un fichero.
- Combinatoria: Calcular el número de combinaciones de un conjunto, muchas veces se pueden resolver con programación dinámica.
 - Recurrencias
 - Coeficientes binomiales
- Ordenación: Ordenar una serie de elementos complejos (por ejemplo elefantes)
- Problemas de entrada/salida: La única complicación es tratar con la entrada o la salida del problema. Por ejemplo dibujar letras con asteriscos

Páginas con problemas

Los dos principales sitios para practicar son la página de USA Computing Olympiad (USACO [7]) y el Problem Set Archive de la Universidad de Valladolid (Uva [8]).

La USACO tiene una gran cantidad de problemas ordenados (más o menos) por su dificultad. Para acceder a nuevas secciones de problemas hay que resolver las anteriores. Las secciones suelen venir acompañadas de algún texto introductorio sobre algoritmos que es interesante leer. Los problemas de esta página son en general sencillos de entender, están bien definidos en su enunciado y no tienen trampas.

La página de la UVa tiene más de mil de problemas del mismo tipo que los que ponen en la SWERC. Se puede acceder a todos los enunciados desde el principio (no hay "secciones" como en USACO) y no hay ninguna clasificación para los problemas, los más sencillos están mezclados con los más difíciles. Los problemas de esta página hay que leerlos muy atentamente para descubrir todos los detalles y las posibles trampas en el enunciado.

Otra página interesante es The 2000's ACM-ICPC Live Archive Around the World [9]. Pertenece también a la Universidad de Valladolid y tiene una base de datos con los problemas que han aparecido en el ACM ICPC durante los últimos años. Tiene cientos de problemas de todas las fases clasificatorias del mundo y también de las fases finales. Lamentablemente esta página no tiene solución para muchos de los problemas que contiene por lo que no es tan útil como las otras dos para practicar.

Concursos de programación

Las universidades de los equipos mejor clasificados en la ACM ICPC se involucran y esfuerzan en la selección de los estudiantes que les van a representar. Por ejemplo, la Universitat Politècnica de Catalunya [10], la Universidad Politécnica de Madrid [11] o la Universidad Autónoma de Madrid [12], todas ellas mucho mejor clasificadas que la UPV, tienen asignaturas de libre elección para la formación de sus equipos y organizan concursos individuales para elegir a los representantes.

La Universidad de Valladolid, a parte de organizar su propia fase local, mantiene una página [13] donde cada pocas semanas se organizan concursos accesibles para cualquier persona que quiera participar.

La USACO también organiza competiciones online cada cierto tiempo en esta página [14].

En la UPV, el Concurso de algoritmos de la asignatura EDA [15] que se realiza en la Facultad de Informática, aparte de sólo ser inaccesible para alumnos de otras carreras, solamente premia la velocidad de ejecución del programa y no la velocidad de codificación como se requiere en la ACM ICPC.

Cómo practicar

Para conseguir un resultado decente es necesario resolver muchos problemas (cuantos más mejor) y participar en todos los concursos online que sea posible. Se puede alternar la resolución de problemas de la USACO y de la UVa. Los problemas de la USACO están bien para ir mejorando progresivamente pero uno se puede quedar atascado en un problema y ya no poder continuar. La UVa en cambio no tiene los problemas clasificados por dificultad pero podemos utilizar la utilidad next2solve [16] de Igor Naverniouk para conseguir un listado aproximado de los problemas según su dificultad.

Los concursos online de la UVa normalmente son los sábados y se anuncian con antelación en la web del Contest Hosting Service. Estas competiciones tienen un horario fijo y suele haber 5 horas para resolver entre 5 y 10 problemas.

En la USACO sin embargo los concursos son de 3 horas pero están accesibles durante todo el fin de semana, el tiempo empieza a correr en el momento que accedemos a la página con los problemas. Esta libertad de horario hace más fácil participar en esta competición que en las de la UVA. Además, los concursos de la USACO están divididos en 3 niveles (oro, plata y bronce), tienes que quedar en una buena posición en tu nivel para que te inviten a participar en el siguiente. Los problemas de nivel bronce son bastante sencillos (problemas de cálculo, grafos sencillos, dfs, bfs... etc), en la división plata tenemos problemas más complicados (programación dinámica, grafos más complejos) y por último la división de oro que tiene problemas de dificultad extrema. Se ha creado también la división level1 que tiene problemas mucho más sencillos orientados a gente que está empezando a programar.

Es importante participar en todos los concursos que sea posible para así comprobar la evolución personal y ganar experiencia.

Nuestro entrenamiento

Desde finales de agosto que formamos el equipo hasta el día de la competición. Nuestros entrenamientos han consistido en:

- Entrenamiento personal: Cada miembro del equipo, por su cuenta, resolvía problemas de la USACO o la UVa. Entre los 3 hemos resuelto casi 300 problemas de la UVa y varias secciones de la USACO. También hemos dedicado tiempo en leer libros sobre programación y algoritmos cada uno a su ritmo. Cada miembro ha participado en los concursos que le ha sido posible (entre 5 y 10 concursos cada uno).
- Entrenamiento en equipo: Nos hemos reunido siempre que hemos podido para participar en concursos online. No es fácil que todos estén disponibles las horas que dura la competición y a veces tampoco es sencillo encontrar un sitio para reunirnos y hacerla. Hemos entrenado en equipo en unas 5 ocasiones durante 3 meses.

Estimaciones del tiempo de entrenamiento

Contando 10 semanas de entrenamiento, si hemos quedado unas 2 horas cada semana para entrenar en equipo y además hemos realizado entre 4 y 8 horas de entrenamiento personal podríamos decir que cada miembro ha practicado entre 60 y 100 horas para la competición.

Cada uno hemos participado entre 5 y 10 competiciones online de 4 horas de duración. Nos daría un mínimo de 20 horas y un máximo de 40 horas.

Así, la estimación del total de horas de resolución de problemas y de participación en competiciones sería de entre 80 y 140 horas de preparación totales.

Consejos y avisos

- Si nos quedamos atascados en un problema lo primero es no maldecir al juez que ejecuta nuestro código, tanto la UVa como la USACO son páginas que llevan años funcionando y es poco probable que encontremos un fallo en el juez. El problema estará en nuestro código con toda seguridad.
- Los ordenadores que juzgan los problemas suelen tener el sistema operativo Linux con alguna versión de GCC, hay que tener esto en cuenta a la hora de programar para saber qué funciones podemos usar. Cada página informa sobre qué versiones exactas de Linux y GCC utilizan sus jueces.
- Hay que usar las librerías estándar siempre que sea posible (STL o java.utils). No suele ser necesario reimplementar cada vez las estructuras de datos a no ser que necesitemos mucha eficiencia, pero esto no suele ocurrir.
- Utilizar siempre el algoritmo más simple que conozcamos, que sea capaz de resolver el problema dentro de los límites de tiempo. Gastar menos memoria o la velocidad de ejecución no se tiene en cuenta a la hora de juzgar, así que no es recomendable entretenerse optimizando al máximo un programa. Las funciones estándar de los lenguajes suelen ser suficientemente rápidas para su utilización en estas competiciones, siendo raros los casos en que haya que reescribir (por ejemplo en ensamblador) estas funciones.
- Si la fuerza bruta sirve para resolver un problema dentro de los límites de espacio y tiempo, suele ser el algoritmo más recomendable porque normalmente es el más rápido de escribir.
- No utilizar memoria dinámica si es posible. Reservar y liberar memoria dará muchos más problemas que declarar un simple array. Como en estas competiciones no hay ningún premio por ahorrar memoria es mejor reservar al principio un array de un millón de double, que ir haciendo llamadas a new o malloc(). Normalmente podremos utilizar varios MB de memoria en los programas, pero hay que tener cuidado con no pasar el límite que imponen en la competición.
- Leer el enunciado completamente, anotando el tamaño máximo de la entrada y la salida, los datos relevantes... etc.

Enlaces

- [1] ACM ICPC: <http://icpc.baylor.edu/icpc/>
- [2] SWERC: <http://icpc.baylor.edu/icpc/Regionals/SWERC05/Default.asp>
- [3] Página de Salvador España: <http://www.dsic.upv.es/~sespana/acm/>
- [4] Resultados SWERC 2003-2004: <http://www.polytechnique.edu/icpc2003/ranking.html>
- [5] Resultado SWERC 2004-2005: <http://www.polytechnique.edu/icpc2004/2004/results/>
- [6] Resultados SWERC 2005-2006: <http://www.polytechnique.edu/icpc2004/2005/ranking.html>
- [7] USACO: <http://ace.delos.com/usacogate>
- [8] UVa Problem Set Archive: <http://online-judge.uva.es/problemset/>
- [9] The 2000's ACM-ICPC Live Archive Around the World: <http://acmicpc-live-archive.uva.es/nuevoportal/>
- [10] Universitat Politècnica de Barcelona: <http://concurs.lsi.upc.edu/>
- [11] Universidad Politécnica de Madrid: <http://acm.asoc.fi.upm.es/contest/>
- [12] Universidad Autónoma de Madrid: <http://www.ii.uam.es/~swerc/inicio.php>
- [13] UVa Contest Hosting Service: <http://acm.uva.es/contest/>
- [14] USACO Contest Gate: <http://ace.delos.com/oiigate>
- [15] Concurso de algoritmos de la asignatura EDA:
<http://www.dsic.upv.es/asignaturas/facultad/eda/concurso/>
- [16] next2solve de Igor Naverniouk: <http://www.shygypsy.com/acm/>