UVA ONLINE JUDGE

La arena de programación de la Universidad de Valladolid es una de las plataformas de programación más populares, entre otras porque es la recolección de problemas de programación más grande del mundo. Cuenta con una gran base de datos de problemas clasificados por volúmenes, competiciones anteriores y niveles de dificultad. Los tipos de problemas propuestos en la arena pueden ser:

- Estructuras de datos
- Cadenas de caracteres
- Ordenación
- Aritmética y álgebra
- Combinatoria
- Teoría de Números
- Rastreo Exhaustivo
- Grafos
- Programación dinámica
- Mallados
- Geometría

La UVA cuenta con un Juez virtual que funciona como evaluador de las propuestas de solución a los problemas planteados, que pueden ser codificados en los lenguajes de programación:

- ANSI C 4.5.3 - GNU C Compiler with options: -lm -lcrypt -O2 -pipe -ansi -DONLINE_JUDGE
- JAVA 1.6.0 - Java Sun JDK
- C++ 4.5.3 - GNU C++ Compiler with options: -lm -lcrypt -O2 -pipe -DONLINE_JUDGE
- PASCAL 2.4.0 - Free Pascal Compiler

La entrada y salida de los problemas es estándar, para todos los lenguajes de programación.

La UVA cuenta con un foro de discusión alrededor de las competiciones y problemas en el que se pueden encontrar sugerencias para la solución de los mismos y detalles técnicos relacionados con la implementación y lenguajes de programación.

Definición de la estrategia de uso de la arena

- **Proceso de inscripción**

Para realizar el proceso de inscripción en el juez virtual UVA hay que seguir estos pasos:

1. Dirigirse a la pagina del juez http://uva.onlinejudge.org/

2. Ubicarse en el panel izquierdo, luego dar click en el link Register.

3. Acto seguido debemos llenar el formulario, datos los cuales conformarán nuestro perfil virtual en UVA,(Datos Importantes: username , password).

- **Publicación de los trabajos**

Para publicar un trabajo o solución en el juez ingresamos al juez con nuestra cuenta, luego nos dirigimos al panel izquierdo al botón **quicksubmit**, allíi encontraremos una casilla con el nombre ProblemID que corresponde al número de problema dentro del juez (Para revisar el número de problema usar el botón **browse problems** del panel izquierdo), luego escogemos el lenguaje de programación correspondiente a nuestra solución, pegamos el código en el último espacio y por ultimo damos click en **submit** .

- **Evaluación por parte del juez**

Para ver el veredicto nos dirigimos a la opción My Submissions allí aparecerá una tabla , la decisión del juez se mostrará en la columna vereditc para cada una de las publicaciones realizadas, los veredictos se clasifican de la siguiente forma:

- Accepted(AC) Indica que la solución es correcta

- In Queue(QU) Indica que el juez está ocupado y no puede procesar la solicitud, la solicitud será procesada lo más pronto posible.

- Presentation Error(PE) Hay algun problema con la salida (impresión) se recomienda revisar el formato de salida del problema.

- Wrong Answer(WA) Indica que alguno de los casos de prueba no obtuvo la respuesta correcta.

- Compile Error(CE) el compilador no puedo compilar la solución , los errores de compilación serán especificados via Mail.

-RuntimeError(RE) si se presenta algún error en tiempo de ejecución.

- Time Limit Exceeded (TL) si la solución tarda en responder más de lo especificado.

- Memory Limit Exceeded(ML) si la solución excede la cantidad de memoria usada en tiempo de ejecución.

- Output Limit Exceeded (OL) puede deberse a que la aplicación está mostrando más información de la requerida.

- Submission Error (SE) la publicación de la solución presenta algún error, se recomienda volver a publicar.

- Restricted Function (RF) el juez considera que alguna de las funciones usadas en la solución del problema es “prohibida” dentro de los estándares del problema. se exige usar las bibliotecas estándar.

- Can't Be Judged (CJ) el juez no pose casos de prueba para este problema y no puede ser juzgado, puede deberse a que la referencia del problema ha sido eliminada o considerada obsoleta.

Construcción del Banco de Problemas

PIMB

Lista de Problemas

Nombre	No. Problema	Tema	Dificultad	Interes
Minesweeper	10189	Manejo de estructuras de datos	Medio	Alto
3n+1	100	Recurrencia	bajo	Medio
Factors and	160	Teoria de	Alto	Medio

factorials		numeros		

Problemas Importantes

Nombre	Numero	Razón
Minesweeper	10189	El problema requiere un manejo adecuado de las estructuras de datos en este caso de las matrices, por consiguiente obliga al estudiante a adquirir dominio sobre ellas así como al mismo tiempo se familiariza con la utilidad de estas estructuras a la hora de enfrentarse a problemas que involucran la manipulación de cantidades considerables de información.
3n+1	100	Este problema supone un ejemplo claro de un problema de recurrencia, ya que para la solución del mismo se requiere entender el concepto, esto motivará al estudiante a desafiar el modelo iterativo (por ciclos) de solución frente a un modelo recursivo.
Factors and Factorials	160	Lo interesante de este problema es la aplicación de conceptos de teoria de numeros a un contexto de programación, esto le permitirá al estudiante observar la manera correcta de transformar algoritmos propios de la matemática a algoritmos propios de la informática.

PIMO

Lista de Problemas

Nombre	No. Problema	Tema	Dificultad	Interes
Bicoloring	10004	Grafos	Media	Alta
All road lead where	10009	Recorrido de grafos	Media	Alta
Tower Of Cubes	10051	Grafos	Alta	Media
Freckles	10034	Recorrido de grafos	Alta	Media

Problemas Importantes

Dados en orden de mayor importancia a mejor.

Nombre	Número	Razón
Bicoloring	10004	Es un problema por medio del cual los estudiantes pueden aprender sobre los grafos, sus propiedades y sus formas de modelamiento. Por medio de este, ellos pueden perfeccionar sus conocimientos aprendidos en grafos y obtener mayores conocimientos a los que se obtienen actualmente en la materia.
All road lead where	10009	Inicialmente es un problema que podría despertar un mayor interés en los estudiantes, debido a que va más relacionado con la realidad. Este problema afianza los conocimientos en recorridos de grafos; además de que permite a los estudiantes darse cuenta de la eficiencia que puede existir al representar la información por medio de grafos si esta requiere de búsquedas complejas y soluciones eficientes.

Tower Of Cubes	10051	Este problema puede llevarle a los estudiantes un trabajo fuerte, sobre todo de consulta. Sin embargo, por medio de este se pueden dar cuenta, desde esta materia, que siempre es importante buscar hacer los programas lo más eficientes posibles, ya que no sería lo mismo armar una torre de 2 cubos que una de 10, ni el tiempo que armarlas requiera es el mismo en cualquier implementación.
Freckles	10034	Este problema sería el de mayor dificultad para ellos pues se requiere el uso de algoritmos voraces, por lo tanto son conocimientos que hasta este momento muchos podrían no tener. Por lo tanto se haría necesaria una etapa fuerte de consulta. Sin embargo, este les permite desarrollar capacidades para detectar cuando una solución puede no ser la mejor; y además les permite obtener conocimientos importantes para su formación como Ingenieros.

TPRO

LISTA DE PROBLEMAS

Nombre	**No. Problema**	**Tema**	**Dificultad**	**Interes**
Is bigger smarter?	10131	Programación dinámica	Media	Alto
Cutting sticks	10003	Programación dinámica	Media	Alto
Tower of cubes	10051	Programación dinámica	Alta	Alto

PROBLEMAS IMPORTANTES

Nombre	Número	Razón
Is bigger smarter?	10131	Este problema se propone para ampliar los conocimientos con los que cuentan los estudiantes, para ayudarlos a perfeccionar sus habilidades en la resolución de problemas complejos incluyendo técnicas de diseño de algoritmos, herramientas y metodologías, que les permiten resolver una amplia gama de problemas e implementar algoritmos eficientes aplicados a problemas de gran escala.
Cutting sticks	10003	Este problema de programación dinámica requiere de una recurrencia bastante intuitiva el cual ayudará a los estudiantes a realizar soluciones óptimas para cualquier problema que se le plantee disminuyendo así el tiempo de ejecución de cada uno de los problemas.
Tower of cubes	10051	Para este problema como para los anteriores se requiere programación dinámica en la cual es importante comprender, definir y realizar la solución del problema de manera que quede lo más simplificada posible reutilizando aquellos cálculos que ya se habían realizado a medida que se avanza en la solución.