Olimpiada Matemática Argentina - Torneo de Computación y Matemática

Criterios generales de corrección

Agosto 2007

Información para los miembros de los Jurados del Certamen Intercolegial.

Aquí damos criterios generales de corrección para las pruebas del Torneo de Computación y Matemática (CyM). Esperamos que este material sirva de guía para entender globalmente los criterios específicos y resolver los casos no contemplados. Los criterios específicos de corrección, que analizan detalladamente cada uno de los problemas tomados, se distribuirán unos días después de la prueba. Este material también puede ser útil para los profesores de los participantes.

Una aclaración

Si tuviéramos que resumir todos estos criterios en una sola frase, sería:

> *CyM es una competencia de resolución de problemas de matemática, con la ayuda de la computadora.*

El resto son sólo detalles de implementación. :-)

Secciones

- **Soluciones y Respuestas**

Qué esperamos que aparezca en las respuestas. Dónde y cómo se deben escribir. Qué se considera *papel*.

- **Problemas y Programas**

Cuántos programas hay que entregar con la solución de cada problema. Cómo considerarlos.

- **Detalles de la Corrección**

Puntajes posibles de cada problema. Errores grandes y pequeños.

- **Motivaciones**

En esta sección repetimos los conceptos de las secciones anteriores, pero ordenándolos según las razones que los motivan. Esperamos que ayude a aclarar las razones detrás de cada uno de los criterios específicos.

Secciones útiles de *Preguntas más o menos Frecuentes*

Existe un documento titulado “Preguntas más o menos frecuentes”, disponible en la página web de CyM en la sección “Detalles” o “Más detalles”, con material relacionado con las pruebas del torneo. Una parte importante de ese material tiene que ver con detalles organizativos y previos a la prueba. Actualmente este material está desactualizado.
Sin embargo puede ser útil para los correctores parte de la información de las siguientes partes:

- **Información general:** Un vistazo general sobre el torneo (las fechas están desactualizadas).
- **Organización de la Corrección:** Detalles sobre la composición del jurado (las fechas están desactualizadas). Las instrucciones para comunicar los resultados de las pruebas también están desactualizadas. *Las nuevas fueron enviadas junto a las pruebas.*
- **Recomendaciones sobre lo que deben entregar los participantes:** Qué archivos y hojas deberían entregar al finalizar la prueba, cómo entregarlos, ...

Olimpiada Matemática Argentina - Torneo de Computación y Matemática

Criterios generales de corrección

Agosto 2007

Soluciones y Respuestas

¿Qué es una solución?

Las soluciones tienen (en general) varias partes

- La respuesta: en general un numerito o "sí/no es posible"
- Los razonamientos escritos
- Los programas

¿Qué se evalúan?

Las tres cosas.

¿La respuesta tiene que estar escrita en *papel*?

Queremos que la respuesta esté escrita en forma clara, precisa y concisa en *papel*. Al fin y al cabo hay que encontrar *la respuesta* del problema planteado.

¿Qué se considera *papel*?

Cualquier información entregada por los participantes que se pueda leer sin tener que ejecutar los programas de los participantes. Por ejemplo cosas escritas:

- en papel de verdad (por ejemplo hojas de cuaderno)
- como archivos de texto (.txt, .doc, ...) (pueden ser cosas escritas en el teclado por los participantes, o salidas de los programas ejecutadas durante la prueba)
- como comentarios adentro del programa.
- en cadenas de texto adentro del programa. (por ejemplo en Print "Da 204")

No es *papel* la salida en pantalla o en archivo de los programas cuando se los ejecuta durante la corrección.

Esto parece ser contradicho por el cuarto ítem, pero no lo es: el corrector puede ver la solución **sin** correr el programa, ¡pues lee el código fuente!. (Este método no es recomendable para los participantes ya que es más fácil que la respuesta pase desapercibida durante la corrección.)

¿Qué es necesario justificar en *papel*?

¿¡Todo!? Como en cualquier justificación, qué grado de detalle es apropiado varía según el caso. En algunos casos el programa hace exactamente lo que pide el problema y la justificación es mínima. En otros casos se deben realizar cuentas auxiliares y razonamientos para simplificar o hacer más rápido el programa. Estas cosas deben escribirse como parte de la justificación en papel.

Por ejemplo:

- Si el problema pide todos los números naturales con cierta condición y un programa sólo prueba hasta 10000, hay que justificar que no es necesario analizar el resto.
- Si se buscan todas las soluciones de una ecuación en enteros por tanteo es necesario justificar que los rangos utilizados son los correctos.
- En cambio si se pide sólo una solución de una ecuación entre todas las posibles, los participantes pueden realizar simplificaciones, restricciones de rangos y estrategias heurísticas siempre y cuando finalmente encuentren una solución y vean que realmente es una solución del problema original.

Olimpiada Matemática Argentina - Torneo de Computación y Matemática

Criterios generales de corrección

Agosto 2007

Problemas y Programas

¿Cómo se consideran los programas?

Son una parte acelerada de la justificación. Así que se debe analizar el código fuente (.bas, .pas, .c, .cpp) de la misma manera que se analizan los razonamientos en *papel*. También se los debe ejecutar para ver qué resultados dan y cuánto tardan.

No se revisan los programas ejecutables (.exe); es más, se pide que no los entreguen.

¿Cómo es el caso típico?

Normalmente cada participante entrega un programa para cada problema, y al finalizar el programa queda escrita en la pantalla el resultado del problema (y quizás algunos datos auxiliares). Sin embargo esto no es obligatorio. Las excepciones en general aparecen sólo en los niveles y rondas más avanzadas.

¿Es obligatorio entregar un programa?

En realidad no es *necesario* hacer programas para resolver los problemas. Los programas los consideramos solamente una parte acelerada de la justificación.

- En algunos casos hay formas de resolver los problemas a mano (que son quizás un poco más laboriosas) que están bien.
- En otros casos el programa se utiliza sólo para hacer algunos tanteos previos que permitan intuir un camino hacia la solución, y luego resolver el problema con justificaciones y cálculos en *papel*. (De todas maneras nos gusta mucho ver qué programas usaron como herramientas.)

¿El programa debe dar la respuesta?

No, el programa es sólo una parte. Los participantes pueden operar y analizar los datos antes o después del programa si lo consideran conveniente.

¿Un solo programa por problema?

No, pueden usarse todos los que quieran.

- Se puede usar uno para hacer tanteos previos y otro como parte de la demostración.
- Se puede separar en casos y analizar cada caso con un programa especifico.
- Se puede resolver la primera mitad de un problema, pensar un rato en papel, y resolver el resto con otro programa.
- ¡Puede no haber programa!
- O cualquier otra combinación que le resulte útil al participante.

Por todo esto se pide que den la respuesta en forma clara y escrita en *papel*.

Las entradas y salidas

En general no es necesario que los programa tengan entradas (“entrada de datos”). Además las salidas en general son cortas y se pueden copiar a mano en *papel*. Así que en general no es necesario que los programas manejen archivos (pero no está prohibido).

Olimpiada Matemática Argentina - Torneo de Computación y Matemática

Criterios generales de corrección

Agosto 2007

Detalles de la Corrección

¿Qué notas se puede obtener en cada problema?

Hay tres notas posibles: 'Bien', 'Regular' y 'Mal'. Los errores pequeños se indican agregando menos '–' (pueden ser varios). (En algún caso *excepcional* se puede agregar un '+'.)

Cada Bien vale 1 punto, cada Regular ½ punto, y cada Mal 0 puntos. Para aprobar es necesario tener dos puntos, por ejemplo se puede tener dos problemas Bien, o uno Bien y dos Regulares. No importa la cantidad de más y menos que aparezcan.

¿La solución tiene que ser igual a la solución oficial?

No es necesario. Si el problema esté bien resuelto (y bien justificado) está Bien.

¿Hay un tiempo máximo de ejecución de los programas?

No hace falta que los programas sean instantáneos. El máximo sería igual a la duración de la prueba, o sea de unas 3 horas (hay que tener en cuenta que el tiempo de ejecución varía con la velocidad de la computadora). Repetimos: el problema debe ser *resuelto* durante la prueba.

¿Y si hay un programa que está "bien" pero tarda demasiado?

A veces se encuentran programas que al analizarlos uno se da cuenta de que darían la respuesta correcta al terminar, pero que tardan mucho (o sea más de 3 horas, por ejemplo 3 años). Una solución de este tipo **no** se considera bien. La idea es que encuentren un método para resolver el problema durante el tiempo de prueba. Cómo el tiempo varía de computadora en computadora, al corregir es difícil saber el tiempo exacto que le tomo al participante. Esta es otra razón por la que es importante que den la respuesta en forma clara escrita en *papel*.

Si la respuesta del programa está en *papel* y es correcta, las demostraciones y el programa son correctos, entonces podemos aceptar que el participante corrió el programa durante la prueba, y por lo tanto resolvió el problema. En caso contrario, si la respuesta no está, entonces el jurado puede razonablemente suponer que el participante no terminó de correr el programa, y por lo tanto no resolvió completamente el problema.

¿La respuesta tiene que estar escrita en *papel*?

Sí. En caso de que no esté escrita pero los programas sean rápidos y claros (y estén el resto de las justificaciones) en vez de considerarse 'Bien', la nota será de 'Bien –' ó 'Bien – –'. Si no es así puede ser que el problema se considere 'Regular' o 'Mal'.

¿Se dan puntos por programas rápidos/ cortos/ estructurados?

No, aunque son más agradables. La idea es ayudarse a obtener la solución del problema, no programar *lindo* (de todas manera es bueno programar *bien* y *elegante*).

¿Se dan puntos por colores y gráficos?

No, no se tiene en cuenta para la corrección.

¿Es necesario hacer diagramas de flujo/bloques, pseudocódigo, ...?

No.

¿Alcanza con hacer diagramas de flujo/bloques, pseudocódigo, ...?

No, se necesita escribir y ejecutar el programa realmente para obtener la solución. Repetimos: un problema debe estar completamente resuelto para estar Bien.

Olimpiada Matemática Argentina - Torneo de Computación y Matemática

Criterios generales de corrección

Agosto 2007

(Detalles de la Corrección - continuación)

¿Es necesario explicar los programas?

El programa es parte de la justificación y debe ser claro y entendible. No hace falta explicar línea por línea, pero es bueno algún comentario sobre qué está haciendo el programa cada 5 o 10 líneas. Como en cualquier justificación, qué grado de detalle es apropiado varía según el caso. Lo ideal sería que siempre haya una explicación del programa, de longitud proporcional a la complejidad del mismo. Una oración breve podría ser suficiente. En algunos casos es necesario aclarar o justificar en *papel* las ideas y algoritmos que se utilizan.

¿Qué pasa si hay algún error de cuentas en el *papel*?

Si el error es sólo de cuentas y no afecta el desarrollo del resto de la solución se puede "arreglar" y en vez de considerarse 'Bien', la nota será de 'Bien –' ó 'Bien – –'.

¿Qué pasa si el programa tiene algún error pequeño?

Idem.

¿Qué es un error pequeño?

A veces es posible "arreglar" un programa con unas pocas modificaciones, por ejemplo

- Arreglar un número mal tipeado.
- Cambiar un tipo de dato (cambiar integer por long).
- Reinicializar una variable (agregar "total = 0").
- Cambiar un mayor por menor o viceversa (depende del problema) .
- Agregar o sacar un punto y coma (;) .
- Cambiar un igual por dos (= por ==) .

¡La idea es hacer algún pequeño arreglo, no reescribir completamente el programa!

Si los errores son pequeños y pocos, y no afectan el desarrollo del resto de la solución se puede "arreglar" y en vez de considerarse 'Bien', la nota será de 'Bien –' ó 'Bien – –'.

La motivación detrás del concepto del "error pequeño" es ser compasivo si es obvio que el participante se "confundió". Pero ver siguiente ítem.

¿Y si hay un error que no cambia el resultado?

Un error es un error. Dependiendo de la gravedad puede ser que sólo agregue unos menos(–) o que el problema esté directamente 'Regular' o Mal.

Por ejemplo: En algunos problemas se pide encontrar *todas* las soluciones de una ecuación. A veces logran encontrarlas a todas por tanteo, pero no justifican por qué alcanza con tantear en ese rango. En ese caso no pueden asegurar que encontraron *todas* las soluciones.

¿Puede ser grave un error pequeño?

Sí, a veces un error pequeño cambia los razonamientos que hay que hacer para terminar la solución. No es grave si se pueden "arrastrar" las correcciones sin que cambien los razonamientos involucrados en las justificaciones.

Olimpiada Matemática Argentina - Torneo de Computación y Matemática

Criterios generales de corrección

Agosto 2007

Motivaciones

Queremos contarles algunas de nuestras motivaciones al realizar el torneo y ver cómo se reflejan en los criterios de corrección de las pruebas y en algunos otros puntos organizativos.

- Estimular la actividad mental, en particular la matemática.
  - Los problemas no son triviales (aunque tienden a ser simples en las primeras rondas y niveles, y aumentan en dificultad a medida que aumenta la ronda y el nivel).
  - Los participantes tienen que tener tiempo y condiciones aptas para pensar (y una mesa para poder escribir).
  - Los problemas son de matemática, y se deben corregir como problemas de matemática.
  - Tienen que aparecer todos los razonamientos, justificaciones y cuentas intermedias necesarias para llegar hasta la respuesta.
  - No se evalúa sólo el resultado. Si el razonamiento está bien, pero hay algún error *pequeño* (de cuentas, en el programa, etc.) el problema está "casi bien resuelto" aunque el resultado sea erróneo. De la misma manera si el resultado está bien pero no está justificado o la justificación es errónea el problema está "mal resuelto".
  - Los programas son solamente una parte de la justificación; son una justificación acelerada (todo lo que hace el programa se puede hacer a mano pero tarda mucho más).
- Desarrollar la capacidad para resolver problemas.
  - Los problemas deben estar resueltos en forma completa. En particular tiene que estar la respuesta del problema en forma clara.
  - Un esquema de resolución (o sea las ideas o pasos generales que uno debería seguir para terminar de resolver el problema) no es una solución. Hay que escribir la solución realmente y completar todos los detalles intermedios hasta obtener el resultado.
  - De la misma manera un programa que tarda demasiado no es una solución. (Es el equivalente a decir "No me alcanzó el tiempo. Lo único que me falta hacer es probar a mano con la calculadora todos los números entre 1 y 1000000 y ver cuál es el que sirve." Los programas deben ser efectivos.
  - Un programa escrito en pseudocódigo, diagrama de flujo/bloques tampoco alcanza.
- Aprovechar las ventajas que tiene la computadora como una herramienta la resolución de problemas de matemática.
  - En general se utiliza mucha capacidad de cálculo. Pero no tanta, no hace falta una computadora último modelo.
  - Es sólo una herramienta más. No hay una relación directa entre problemas y programas.
  - En muchos problemas es necesario pensar y utilizar herramientas matemáticas para simplificar el problema y lograr obtener uno equivalente para el cual lo programado funcione en el tiempo que dura la prueba.
  - No es obligatorio usar la computadora. A veces se pueden resolver a mano.
  - En algunos casos los programas sólo se usan para investigar algunos casos particulares y poder intuir alguna propiedad o solución, que se justifica sin utiliza la computadora.
  - No se evalúa la presentación gráfica, colores, efectos especiales y otras decoraciones. Sólo se tiene en cuenta lo que conduce hacia la solución de los problemas.

Criterios generales de corrección

Agosto 2007

(Motivaciones - continuación)

- Desarrollar la habilidad de intercambiar ideas en forma clara.
  - Tienen que justificar sus razonamientos.
  - Pedimos que expliquen cómo funcionan los programas que utilizan. (No línea por línea. Pero los programas deben ser entendibles.)
- Aprendan a tener más claras las sus propias ideas.
  - Los lenguajes tienen muy pocas instrucciones, que son muy simples, y no tienen herramientas matemáticas avanzadas. Así que hay que tener claros los algoritmos y heurísticas que se utilizan para resolver los problemas manualmente, para poder escribir los programas respectivos.
  - La computadora no "entiende": realiza las instrucciones que se le dan exacta y literalmente. Por eso exige claridad y precisión. No acepta ambigüedad o vaguedad.
- Aprendan una herramienta útil para otras actividades.
  - Utilizamos programas estándar (aunque un poco viejos porque son simples).
  - Hay varios lenguajes, así cada uno puede usar el que le resulta más cómodo.
- Otra oportunidad para resolver problemas
  - Nos gusta resolver problemas. Esperamos que les gusten nuestros problemas.
  - Nos gustaría que los participantes vean cómo los ayudaría una computadora a resolver problemas de OMA. (Quizás haya que agrandar algunos números.)
  - Y viceversa. O sea, que analicen si es posible resolver los problemas planteados en CyM sin una computadora. (Quizás haya que achicar algunos números.)