Tercer Concurso Local de Programación ACM ICPC

Universidad Autónoma Metropolitana Unidad Azcapotzalco

Examen eliminatorio, 13 a 16 de Octubre de 2006

Instrucciones: Abajo encontrarás los enunciados de cinco problemas. Cada problema tiene un valor entre 20 y 60 puntos, los cuales se obtendrán de las salidas que su programa entregue para cada una de 10 entradas distintas. **Todos los programas se evaluarán en Linux.** Que tu programa funcione con el ejemplo no quiere decir que funcionará siempre.

Para cada problema que resuelvas deberás enviar por correo el código fuente de un programa a la dirección `uam06acm@gmail.com` indicando claramente tu nombre completo. De preferencia envía todos tus códigos en un correo. El nombre de los archivos que envíes debe ser de la forma `xxx.zzz`, donde `xxx` es el nombre del problema y `zzz` es la extensión, según el lenguaje que hayas usado. Si así lo deseas, puedes resolver cada problema en un lenguaje distinto. Tu programa deberá leer e imprimir exactamente los datos que se indican (ni más ni menos) usando la entrada y la salida estándar. En particular, tu programa no deberá borrar la pantalla, escribir ningún tipo de letrero adicional, usar la biblioteca `conio`, etc. Quienes programen en C deberán usar el gcc, quienes programen en C++ deberán usar el g++ y quienes programen en Java deberán usar el javac de la siguiente manera:

```
gcc -ansi -o nombre_del_ejecutable nombre_del_fuente.c -lm
g++ -ansi -o nombre_del_ejecutable nombre_del_fuente.cpp -lm
javac nombre_del_fuente.java
```

Deberás enviar tus códigos fuente a más tardar el 16/10/2006 a las 16:00.

`/* :-) Los organizadores te deseamos éxito en tu examen (-: */`

Problema 1: Soluciones enteras de una ecuación (20 puntos)

Código fuente: `see.c, see.cpp, see.java`

Escribe un programa que determine la cantidad **n** de soluciones enteras que tiene la ecuación **a**x + **b**y = **c** con **x** ≥ 0, **y** ≥ 0. Observa que es posible que esta ecuación no tenga ninguna solución (es decir **n** = 0) o que tenga un número infinito de soluciones (en cuyo caso debes reportar **n** = -1). Por ejemplo, si **a** = 2, **b** = 5 y **c** = 20 entonces **n** = 3 ya que la ecuación 2**x** + 5**y** = 20 tiene las soluciones: **x** = 0, **y** = 4; **x** = 5, **y** = 2; **x** = 10, **y** = 0.

Entrada: Tres números enteros **a**, **b**, **c** separados por espacios y todos ellos en el intervalo de -1,000,000 a 1,000,000 (incluyéndolos).

Salida: Un número entero **n**.

Evaluación: 2 puntos por el valor correcto de **n**.

Ejemplo de entrada	Ejemplo de salida
2 5 20	3

Problema 2: Diferencias de cuadrados (30 puntos)
Código fuente: `ddc.c, ddc.cpp, ddc.java`

Algunos enteros se pueden representar como diferencia de dos enteros al cuadrado, por ejemplo $15 = 4^2 - 1^2 = 8^2 - 7^2$. Escribe un programa que determine la cantidad de formas distintas **t** en las que se puede escribir un número entero **n** como diferencia de dos cuadrados $\mathbf{a}^2 - \mathbf{b}^2$ donde $\mathbf{a} \geq \mathbf{b} \geq 0$.

Entrada: Un número entero **n** tal que $0 \leq \mathbf{n} \leq 1{,}000{,}000$.

Salida: Un número entero **t**.

Evaluación: 3 puntos por el valor correcto de **t**.

Ejemplo de entrada	Ejemplo de salida
15	2

Problema 3: Números en binario al revés (40 puntos)
Código fuente: `nbr.c, nbr.cpp, nbr.java`

Considere un entero positivo **n** que se escribe $\mathbf{b}_m\mathbf{b}_{m-1}...\mathbf{b}_1\mathbf{b}_0$ en binario, donde $\mathbf{m} \geq 0$ y $\mathbf{b}_m = 1$. El complemento de **n** es el entero que se escribe $\mathbf{a}_m\mathbf{a}_{m-1}...\mathbf{a}_1\mathbf{a}_0$ en binario, donde $\mathbf{a}_i = 1 - \mathbf{b}_i$ para toda $0 \leq \mathbf{i} \leq \mathbf{m}$. El reverso de **n** es el entero que se escribe $\mathbf{b}_0\mathbf{b}_1...\mathbf{b}_{m-1}\mathbf{b}_m$ en binario. Escribe un programa que calcule el complemento **p** y el reverso **q** de un entero **n**. Por ejemplo, si **n** = 2006 (11111010110 en binario) entonces su complemento es **p** = 41 (00000101001 en binario) y su reverso es **q** = 863 (01101011111 en binario).

Entrada: Un número entero **n** tal que $1 \leq \mathbf{n} \leq 4{,}000{,}000{,}000$.

Salida: Dos números enteros **p**, **q** separados por un espacio.

Evaluación: 2 puntos por cada uno de **p** y **q**.

Ejemplo de entrada	Ejemplo de salida
2006	41 863

Problema 4: Rectángulos blancos y negros (50 puntos)
Código fuente: `rbn.c, rbn.cpp, rbn.java`

Considere una rejilla de **m** por **n** puntos blancos y negros. La rejilla tiene un rectángulo blanco si los cuatro puntos en los vértices de un rectángulo con sus lados paralelos a los de la rejilla son todos blancos. Los rectángulos negros se definen de forma similar. Escribe un programa que encuentre el

número **p** de rectángulos blancos y el número **q** de rectángulos negros en una rejilla. El ejemplo mostrado abajo tiene siete rectángulos blancos con coordenadas (1,1,2,5), (1,1,3,3), (1,1,3,5), (1,3,3,5), (2,1,3,4), (2,1,3,5) y (2,4,3,5), pero no tiene ningún rectángulo negro. (**a**, **b**, **c**, **d**) significa que los cuatro puntos con coordenadas (**a**, **b**), (**a**, **d**), (**c**, **b**) y (**c**, **d**) tienen el mismo color.

Entrada: Dos números enteros $2 \leq \mathbf{m}, \mathbf{n} \leq 100$, separados por un espacio, seguidos de **m** renglones con **n** números separados por espacios en cada renglón. Un 1 representa un punto blanco y un 0 representa un punto negro.

Salida: Dos números enteros **p**, **q** separados por un espacio.

Evaluación: 2 puntos por cada uno de **p** y **q**. 1 punto si ambos son correctos.

Ejemplo de entrada	Ejemplo de salida
3 5 1 0 1 0 1 1 0 0 1 1 1 0 1 1 1	7 0

Problema 5: Simulando un juego de cartas (60 puntos)

Código fuente: `sjc.c, sjc.cpp, sjc.java`

Se tiene una pila de **n** cartas, cada una de ellas con un número del 1 al **n**. Un movimiento del juego consiste en fijarse en el número **a** anotado en la carta hasta arriba de la pila e invertir la posición de las **a** cartas hasta arriba de la pila. El juego termina cuando se repite alguna configuración de cartas. Por ejemplo, si las cartas estaban en el orden 3, 1, 4, 1, 5 después de un movimiento se deberán invertir las 3 primeras cartas y quedan 4, 1, 3, 1, 5. Después de otro movimiento las cartas quedan 1, 3, 1, 4, 5. El juego termina en el tercer movimiento, puesto que las cartas vuelven a quedar 1, 3, 1, 4, 5. Escribe un programa que determine el número **m** de movimientos para que termine un juego y cuál es la configuración final de las cartas.

Entrada: Un número entero **n** con $1 \leq \mathbf{n} \leq 9$, seguido de un renglón con **n** números enteros $\mathbf{a}_1$, $\mathbf{a}_2$, ..., $\mathbf{a}_n$ en el intervalo 1 a **n** y separados por espacios. El primero de estos números representa la carta hasta arriba de la pila, etc.

Salida: Un número entero **m** seguido de un renglón con **n** números enteros $\mathbf{b}_1$, $\mathbf{b}_2$, ..., $\mathbf{b}_n$ en el intervalo 1 a **n** y separados por espacios. El primero de estos números representa la carta hasta arriba de la pila al terminar, etc.

Evaluación: 2 puntos por el valor de **m** y 4 puntos por la configuración final.

Ejemplo de entrada	Ejemplo de salida
5 3 1 4 1 5	3 1 3 1 4 5