INSTRUCCIONES

Bienvenido al examen práctico de la fase estatal de la 8ª Olimpiada Mexicana de Informática. En este examen tendrás que desarrollar programas que sean capaces de resolver un problema dado.

El examen consta de 3 problemas. Para cada uno de ellos deberás entregar un archivo ejecutable que realice las operaciones que se te piden. Los tres problemas tienen un valor diferente de puntaje de acuerdo a su dificultad. Los valores de los problemas son:

- **Primos 10 puntos.**
- **Tin Marín 20 puntos.**
- **Tesoro 30 puntos.**

El examen práctico tiene un valor total de 60 puntos que se sumarán con el puntaje que hayas obtenido en la parte teórica.

El examen práctico tiene una duración total de 4 horas.

Para resolver los problemas del examen práctico podrás utilizar lenguaje C/C++ o Pascal. Los compiladores a utilizar serán Turbo Pascal 7.0 y Turbo C/C++ 3.0. Ambos compiladores deberán estar instalados en tu máquina de trabajo y podrás utilizar el que mejor te parezca.

Para obtener tu puntaje se utilizarán tus archivos ejecutables, **NO SE UTILIZARÁN CÓDIGOS**. Es muy importante que te asegures de que estas generando un archivo ejecutable. Si programas en C/C++, tu archivo ejecutable se generará de manera automática. Si programas en Pascal es necesario que en el menú ***Compile*** en la sección ***Destination*** tengas el valor ***Disk***. **DE OTRO MODO NO GENERARÁS EJECUTABLE Y OBTENDRÁS CERO PUNTOS.**

Tus archivos ejecutables deberan llamarse PRIMOS.EXE, MARIN.EXE y TESORO.EXE para cada uno de los problemas respectivamente

En los problemas del examen práctico se te pide que leas los datos de archivos de texto, de igual forma se te pide que escribas tus resultados a archivos de texto. Los compiladores cuentan con ayuda en la que puedes ver como se leen y escriben archivos de texto. De cualquier forma a continuación se anexan las rutinas que permiten leer y escribir datos desde un archivo de texto.

LENGUAJE C (Leer de un archivo de texo).

```
FILE *arch;          /* Asi se define una variable de archivo */

arch = fopen("INPUT.TXT","rt"); /* Asi se abre un archivo de texto para lectura */
fscanf(arch,"%d",&v);       /* Con fscanf se pueden leer datos del archivo de texto */
fclose(arch);          /* Por ultimo, cuando termines tienes que cerrar el archivo*/
```

LENGUAJE C (Escribir a un archivo de texo).

```
FILE *arch;          /* Asi se define una variable de archivo */

arch = fopen("OUTPUT.TXT","wt"); /* Abre el archivo de texto para escritura */
fprintf(arch,"%d",v); /* Con fprintf se pueden escribir datos a un archivo de texto */
fclose(arch);        /* Por ultimo, cuando termines tienes que cerrar el archivo*/
```

LENGUAJE Pascal (Leer de un archivo de texo).

```
Arch : text;          /* Asi se define una variable de archivo */

Assign(arch,'INPUT.TXT');  //  Primero hay que asignar el archivo
Reset(arch);               // Abre el archivo para lectura
Readln(arch,v);            // Con readln, pudes leer datos de un archivo de texto
Close(arch);               // Por ultimo, cuando termines hay que cerrar el archivo
```

LENGUAJE Pascal (Escribir a un archivo de texo).

```
Arch : text;          /* Asi se define una variable de archivo */

Assign(arch,'OUTPUT.TXT');      //  Primero hay que asignar el archivo
Rewrite(arch);             // Abre el archivo para escritura
Writeln(arch,v);           // Con writeln, puedes escribir datos a un archivo de texto
Close(arch);               // Por ultimo, cuando termines hay que cerrar el archivo
```

Es muy importante que sólo leas datos del archivo de entrada y que sólo escribas datos al archivo de salida. No debes leer ningún dato del teclado ni escribir nada a la pantalla, ya que si lo haces el evaluador no te dará ningún punto. Evita utilizar interfaces de usuario como "Introduzca el dato: ", también evita utilizar funciones como *getch()* en C o *readln;* en Pascal, ya que estas esperan a que un usuario presione una tecla y el evaluador tomará esto como que tu programa se ha trabado.

También es muy importante que utilices las menos librerías posibles, evita utilizar librerías como *crt, graphics, etc* en Pascal, de hecho trata de no utilizar ninguna librería a menos que lo consideres estrictamente necesario. Igualmente en C trata de limitarte a usar <stdio.h> y <stdlib.h>, evita usar <conio.h> y otro tipo de librerías.

Por último, concéntrate, haz tu mejor esfuerzo y trata de generar ejecutable para los tres problemas aunque no estés completamente seguro de que tu solución sea la correcta o la óptima.

El Comité Olímpico Mexicano de Informática te desea MUCHA SUERTE!

Primos

Los organizadores de la Olimpiada de Informática últimamente se han interesado por los números primos. Un número primo es aquel que sólo se puede dividir entre 1 y él mismo sin dejar residuo (el número 1 no es un primo), por ejemplo el 2, 3, 5, etc.

A los organizadores les interesa hacer un programa que sea capaz de encontrar todos los números primos que hay en un rango, por ejemplo todos los primos que hay entre 50 y 1500.

Problema

Tienes que hacer un programa que sea capaz de encontrar todos los números primos en un cierto rango que siempre estará entre 1 y 10000.

Entrada

Tu programa deberá leer del archivo de texto de entrada INPUT.TXT dos números. Cada número estará en una línea. Estos números indicarán el límite inferior y superior del rango en el que tienes que buscar los primos.

Salida

Tu programa deberá escribir en el archivo de texto de salida OUTPUT.TXT todos los primos que haya encontrado en el rango que se pidió, poniendo cada primo en una línea. Los primos que encuentres los deberás escribir ordenados de menor a mayor.

Ejemplo

En este ejemplo se muestra un archivo de entrada y su respectivo archivo de salida.

El archivo de entrada INPUT.TXT tiene dos líneas con un número cada una. Como se describe en la sección Entrada, estos números representan el límite inferior y superior del rango de búsqueda.

En el archivo de salida OUTPUT.TXT debes escribir todos los números primos que estén en el rango que se pide, escribiendo un número en cada línea y ordenándolos de menor a mayor.

INPUT.TXT	OUTPUT.TXT
7 35	7 11 13 17 19 23 29 31

Tin Marin

¿Recuerdas cuando de pequeño jugabas a los encantados o a las escondidillas? Para elegir al que comenzaba siempre era necesario seleccionar con el famoso "tin marin de do pingüe" o alguna rima similar. En esos momentos que útil hubiera sido poder calcular de antemano en que orden iban a ir saliendo los jugadores y quien se iba a quedar hasta el final.

Ahora que estas en la Olimpiada de Informática esto resulta sencillo, ya que por medio de la computadora puedes calcular el orden exacto en el que irán saliendo los jugadores. Desgraciadamente lo más probable es que no le vayas a dar utilidad a tu programa, sin embargo siempre puedes ayudar a tus hermanos menores.

Problema

Tienes que desarrollar un programa que dado el número inicial de jugadores y el largo del "tin marín" te diga en que posición tienes que colocarte para ser el último en salir.

La primera vez que se cuente siempre se inicia por la primera posición y se va eliminando a la persona en donde termino el "tin marín".

Por ejemplo si hay 5 jugadores y con el "tin marín" cuentas a 8 jugadores, el desarrollo sería el siguiente:

Empiezas por la posición 1 y cuentas 8 lugares (1,2,3,4,5,1,2,3). Como solo hay 5 jugadores, al llegar al cinco vuelves al primero. De la cuenta anterior se eliminó al jugador 3. De nuevo vuelves a contar empezando por el que sigue del último eliminado (4,5,1,2,4,5,1,2), se elimina al jugador 2. De nuevo vuelves a contar (4,5,1,4,5,1,4,5), eliminas al 5. Cuentas (1,4,1,4,1,4,1,4), eliminas al 4 y el último que quedo es el jugador 1. Por lo tanto si querías quedarte al final tendrías que haberte puesto en la posición 1.

Entrada

Tu programa deberá leer del archivo de texto de entrada INPUT.TXT dos números, cada uno en una línea diferente. El número en la primera línea indica la cantidad de jugadores, y puede estar entre 2 y 10,000. El número en la segunda línea indica cuantos lugares avanzas con cada "tin marín".

Salida

Tu programa deberá escribir en el archivo de salida OUTPUT.TXT un único número que indique la posición en la que te debes colocar para ser el último que quede.

Ejemplo

Abajo se muestran cuales serían los respectivos archivos de entrada y salida para el caso que se describió.

INPUT.TXT	OUTPUT.TXT
5 8	1

Tesoro

Por azares del destino, te encuentras en la famosa Isla del Tesoro. Y no sólo eso, además gracias a tu suerte, tienes en tus manos el mapa del tesoro. Desgraciadamente, las instrucciones del mapa no están del todo claras. Las instrucciones del mapa dicen así:

*"Busca la palmera con 2 troncos y a partir de ahí camina **X** pasos hacía el norte, **Y** pasos hacia el este y finalmente camina **Z** pasos hacia el sur, cava y encontrarás el tesoro."*

Como puedes ver el mapa hace mención de unos números ***X, Y, Z***. Pero no dice sus valores.

Cuando conseguiste el mapa te dieron además unas fichas de mármol. Cada ficha tiene un dígito escrito en la parte delantera y un dígito diferente escrito en la parte trasera.

Después de varios días de investigaciones, y de recordar algunos viejos manuscritos. Deduces que los valores de ***X, Y, Z*** se pueden obtener formando números con los dígitos de las fichas de la siguiente forma.

X = El número primo más grande que se pueda formar utilizando las fichas.

Y = El número primo más chico que se pueda formar utilizando todas las fichas.

Z = El número par más grande que se pueda formar utilizando las fichas.

Obviamente una ficha solo se puede utilizar de un lado, es decir si decides utilizar el dígito en la parte delantera no podrás utilizar el dígito en la parte trasera.

Problema

Tienes que desarrollar un programa que conociendo los dígitos en la parte delantera y trasera de cada ficha calcule los números ***X, Y, Z***.

Entrada

Tu programa deberá leer del archivo de texto de entrada INPUT.TXT los datos de las fichas de la siguiente forma. En la primera línea del archivo hay un número ***N*** que indica el número de fichas que tienes. El número de fichas siempre esta entre 1 y 7. En las siguientes ***N*** líneas del archivo habrá 2 dígitos separados por un espacio en cada línea. Cada uno de estos números representa el dígito en la parte delantera y trasera de la ficha respectiva (Digito = 1..9).

Salida

Tu programa deberá escribir en el archivo de salida OUTPUT.TXT los números ***X, Y, Z*** que se forman con las fichas. Cada número deberá ir en una línea y deberán estar en ese orden, es decir, primero ***X***, después ***Y*** y por último ***Z***.

Ejemplo

Tesoro.ent	**Tesoro.sal**
2 1 3 7 1	73 11 0

NOTA: En caso de que sea imposible obtener un primo o un número par, tu programa deberá escribir 0 en el espacio correspondiente.