**Problema 1:**

Dos números enteros se dicen coprimos si y sólo si el m.c.d. entre ellos es 1. Por ejemplo:

m.c.d.(17,91) = 1 luego 17 y 91 son coprimos.

Escribir un programa que, dado un entero n, determine una matriz nxn de manera que:

*aij = 1 si i y j son coprimos*

*aij = 0 si i y j no son coprimos*

Indicar la CC del algoritmo que lo resuelve con notación O.

**Problema 2:**

Implemente un método que dado un archivo de entrada (archivo que será leído) y uno de salida (archivo que será escrito), especificados con su ruta, y dos palabras: palabraAReemplazar y nuevaPalabra, devuelva en el archivo de salida el contenido del archivo de entrada, pero con la palabra a reemplazar reemplazada por la nueva palabra.

*public void reemplazar(String archivoOrigen, String archivoDestino, String palabraAReemplazar, String nuevaPalabra){*

*...*

*}*

Ejemplo:

Supongamos el contenido del archivo origen:

“Un kit también puede formar parte de otro kit, por lo que un kit es un grupo de elementos de cualquiera de los tipos de elementos (simples o kit). Cada tipo de elemento tiene un código, nombre y un precio.”

La palabra a reemplazar “kit” y la palabra nueva “set”, el método escribirá en el archivo destino (archivoDestino) el contenido:

“Un set también puede formar parte de otro set, por lo que un set es un grupo de elementos de cualquiera de los tipos de elementos (simples o set). Cada tipo de elemento tiene un código, nombre y un precio.”

**Problema 3:**

Escribe un programa que lea un archivo de texto (texto.txt) y devuelva el número de palabras distintas que hay en dicho texto junto con la frecuencia de aparición de cada una de ellas.

Nota: la frecuencia se calcula como el número de veces que aparece la palabra en el texto dividido entre el número total de palabras del texto.

**Problema 4:**

Una cerradura con combinación tiene las siguientes propiedades básicas: la combinación (una secuencia de tres números) está oculta, la cerradura sólo se puede abrir si se le provee la combinación correcta, para cerrar la cerradura no hace falta la combinación, la combinación se puede cambiar pero sólo si se conoce la combinación vieja. Implementar en Java la clase CerraduraConCombinación y hacer una clase para testearla.

**Problema 5:**

Dado un archivo “números.in” que contiene N números enteros (pueden estar repetidos), codifique un programa que permita encontrar los K valores más pequeños. En caso de que existan muchos valores repetidos, cada uno de estos debe considerarse una sola vez.

Indicar la complejidad computacional del algoritmo que resuelve el problema.

Datos de entrada:

Primera línea la cantidad N de datos a procesar seguido de la cantidad K de valores buscados, separados por un blanco.

Siguientes N líneas, los valores a procesar

Datos de salida:

Primera línea la cantidad K de valores obtenidos.

Siguientes K líneas, los K valores más pequeños obtenidos.

|  |  |
| --- | --- |
| numeros.in | pequenios.out |
| 10 3  -3  5  67  87  -12  14  -3  1  1  1 | 3  -12  -3  1 |

[Solución](https://github.com/leoblautzik/K-mas-pequenios/blob/master/KmasPequenios.java)

**Problema 6**

Para cada par de enteros positivos (m,n) podemos definir una función f de la siguiente manera:

f(m,n) = [(m+n-1).(m+n-2)/2] + n

Esta función tiene dos propiedades importantes, cuya validez demuestran los matemáticos:

1- Si f(m,n) = f(s,t) entonces m = s y n = t

2- Dado cualquier entero positivo k existen dos enteros positivos m y n tales que f(m,n) = k.

Dicho en otras palabras, dado cualquier entero positivo k, existe un único par de enteros positivos (m,n) tal que f(m,n) = k. Por lo tanto, esta función permite demostrar el hecho sorprendente de que "hay tantos" elementos en NxN como en N. La demostración que hacen los matemáticos no es constructiva, es decir, prueba que existe el par (m,n) pero no "muestra" el par ni dice cómo calcularlo. Sin embargo, si se hace una tabla de los valores de f para m + n < 7:

--------------------------------------------------------------------------------

m 1 2 3 4 5

n

1 1 2 4 7 11

2 3 5 8 12

3 6 9 13

4 10 14

5 15

--------------------------------------------------------------------------------

veremos que lo que hace f es "numerar" los elementos de una matriz. Se trata entonces de escribir un programa que:

1- permita el ingreso de un par (m,n) y calcule f(m,n), emitiendo en pantalla un mensaje adecuado.

2- permita el ingreso de un entero positivo k y determine el par (m,n) tal que f(m,n) = k, emitiendo en pantalla el resultado.

**Problema 7**

Sin pensarlo demasiado, nos entrometemos en el mundo de los conejos y planteamos el siguiente ejercicio:

Supongamos que una pareja adulta de conejos produce una nueva pareja de conejos cada mes y que cada pareja de conejitos llega a la adultez al cumplir un mes de vida.

Si partimos entonces de una pareja recién nacida, al cabo de un mes tendremos una pareja adulta, 1 mes después habrá dos parejas (una adulta y una recién nacida) y así (suponiendo que los conejos no son mortales):

0 mes 1 pareja recién nacida

1 mes 1 pareja adulta

2 meses 2 parejas (1 adulta y 1 recién nacida)

3 meses 3 parejas (2 adultas y 1 recién nacida)

4 meses 5 parejas (3 adultas y 2 recién nacidas)

Se trata entonces de escribir un programa que permita el ingreso de un número entero positivo N y determine cuántos pares de conejos se habrán producido a partir de una sola pareja después de N meses.

**Problema 8:**

Escribir un método que almacena en una lista 100 número aleatorios entre 1 y 100 y escribir

métodos que resuelvan los siguientes puntos:

a) Mostrar en pantalla los elementos de la lista sin repetir.

b) Mostrar en pantalla los elementos de la lista sin repetir y en orden ascendente.

**Problema 9:**

Escriba un método recursivo que tome un String y devuelva como resultado el String

invertido. Calcule su eficiencia.

**Problema 10:**

Escriba un método recursivo que calcule el máximo común divisor entre dos números

enteros. Calcule su eficiencia.