CodeWars 2019 LEÓN

benvinguts
καλωσόρισμα
hoakipa ana
bonvenon
welcome
i mirépritur
karşılama
warmukelekile
bienvenidos
nau mai
angietorri
benvido
tatiga

#HPCodeWars
#CodeWars19

CodeWars 2019 LEÓN

# HP SCDS

CodeWars 2019 León

GUÍA DE PROBLEMAS

SCDS

# Contenidos

[2] En tiempos de Guerra – 10 Puntos ..... 3
[3] Funciones recursivas – 14 Puntos6
[4] ¿Cuánto queda? – 6 Puntos8
[5] Apunteeeeeen ... Fuego! – 10 Puntos ..... 10
[6] Lindas mariposas – 5 Puntos ..... 13
[7] Resistencias equivalentes – 6 Puntos..... 16
[8] ¡El radio me desaparece! – 8 Puntos..... 20
[9] Ada Lovelace – 9 Puntos ..... 22
[10] XOR variable – 15 Puntos .....25
[11] Coaliciones – 11 Puntos .....31
[12] Primos-RSA – 8 Puntos .....35
[13] Sectoriza sista! – 7 Puntos.....37
[14] Apollo 11 y las importancia de las prioridades – 18 Puntos ..... 40
[15] El corta-pega genético – 5 Puntos.....45
[16] Teorema en el espejo – 3 Puntos.....50
[17] Show me the money (o toma el dinero y corre) – 7 Puntos .....52
[18] Mary Anderson y los limpiaparabrisas– 9 Puntos .....54
[19] Donde está mi planeta? – 3 Puntos .....58

## [2] En tiempos de Guerra – 10 Puntos

Hedy Lamarr fue una actriz e ingeniera de origen austriaco que en 1941 desarrolló un sistema de transmisión novedoso para el guiado de torpedos en la II Guerra Mundial. La peculiaridad de dicho sistema fue que, en vez de transmitir todo el mensaje sobre una misma frecuencia (canal), el mensaje era troceado en partes y cada parte emitida en una frecuencia (o canal) distinto, para que su intercepción fuera más complicada.

Tienes que crear un programa que simule dicho sistema de transmisión y reconstruya el mensaje final analizando la información que va en cada uno de los canales transmitidos. Las siguientes líneas representan un ejemplo de datos de entrada:

*RETIRADA3xxxxxxxxxx CORTO0*

*xxxxxxxxxxxxxxxxxxxxxxxxxx*

*xxxxxxxx INMEDIATA.1xxxxxx*

- Cada línea representa un canal.
- La decodificación del mensaje siempre empieza en el canal 1 (línea 1).
- Al final del trozo de mensaje que contiene un canal viene un número que indica qué canal contiene la siguiente parte del mensaje. Por ejemplo “RETIRADA3” contiene el mensaje “RETIRADA” y el 3 indica que la siguiente parte del mensaje está en el canal 3.
- La siguiente parte del mensaje en el nuevo canal siempre empieza en la misma posición donde acabó la parte anterior (p.e. RETIRADA3 acaba en la posición 8 – partiendo de 0 – por lo que en el canal 3 la siguiente parte del mensaje empieza en dicha posición 8).
- Puede haber hasta 9 canales (1-9).
- Un 0 (p.e. CORTO0) indica que el mensaje se ha terminado.

Para las entradas anteriores, el mensaje final de salida sería “RETIRADA INMEDIATA. CORTO”.

**ENTRADA**
*RETIRADA3xxxxxxxxxx CORTO0*

*xxxxxxxxxxxxxxxxxxxxxxxxxx*

*xxxxxxxx INMEDIATA.1xxxxxx*

**SALIDA**
*RETIRADA INMEDIATA. CORTO*

**SOLUCION**
Escrita en Java:

```
import java.util.*;
```

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;

public class SistemaTransmision {

    private static int posicion = 0;
    private static int canalActual = 0;
    private static ArrayList<String> canales;
    private static StringBuilder mensajeFinal;

    public static void main(String[] args) {

        try {
            BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));

            canales = new ArrayList<String>();
            mensajeFinal = new StringBuilder();

            String input = null;
            while((input = reader.readLine()) != null){
                System.out.println(input);
                canales.add(input);
            }

            canalActual = 1;
            while(canalActual != 0){
                ProcesarCanal();
            }

            System.out.println(mensajeFinal.toString());
        }
        catch(IOException ioex)
        {
            ioex.printStackTrace();
        }
    }

    private static void ProcesarCanal(){

        String canalAProcesar = canales.get(canalActual - 1);
        boolean procesado = false;

        while(true){

            char item = canalAProcesar.charAt(posicion);
```

```
            int intValue = ParseInteger(item);

            if(intValue == -1){
                mensajeFinal.append(item);
            }
            else{
                canalActual = intValue;
                break;
            }

            posicion++;
        }
    }

    private static int ParseInteger(char c){
        int result;
        try{
            result = Integer.parseInt(String.valueOf(c));
        }
        catch (NumberFormatException e){
            result = -1;
        }

        return result;
    }
}
```

## [3] Funciones recursivas – 14 Puntos

Rózsa Politzer fue una matemática húngara que es conocida por ser la madre de la teoría de la recursión.

La recursión ocurre cuando algo este definido dentro de sí mismo. Por ejemplo, cuando estamos en una sala de espejos vemos en un espejo los demás espejos de forma recursiva, que se van perdiendo a lo lejos haciéndonos cada vez más pequeños.

Rózsa Politzer estaba entusiasmada con funciones como la siguiente:

$$F(n) = \frac{n}{10} \cdot F(n-1) + n$$

Sabiendo que F(n) = 1 cuando n ≤ 0.

Si os dais cuenta, para obtener el valor de F(n) hay que saber antes el valor de F(n-1) ... se trata de una función recursiva.

Veamos un Ejemplo. ¿Como se calcula F(3)?

$$F(3) = \frac{3}{10} \cdot F(2) + 3$$

Entendido, ¿no? ...ya... pero ¿y que es F(2)? Pues F(2) es:

$$F(2) = \frac{2}{10} \cdot F(1) + 2$$

...pues estamos en las mismas, ¿y que es F(1)?

$$F(1) = \frac{1}{10} \cdot F(0) + 1$$

...jolín, no acabamos nunca... y que es F(0)? ¡¡Ah!! Aquí termina la recursión. ¡La recursión siempre tiene que terminar porque si no, seria infinita! En nuestro caso, si os fijáis, en el enunciado ponía:

*"Sabiendo que F(n) = 1 cuando n ≤ 0. "*

¡Ah! Pues entonces podemos calcular los F(n) que veíamos antes:

$$F(1) = \left(\frac{1}{10} \cdot 1\right) + 1 = 1.10$$

$$F(2) = \frac{2}{10} \cdot F(1) + 2 = \left(\frac{2}{10} \cdot 1.10\right) + 2 = 2.22$$

$$F(3) = \left(\frac{3}{10} \cdot 2.22\right) + 3 = 3.67$$

Con números pequeños es más o menos fácil calcular el valor de esta función, pero ¿Y si quisiéramos saber el valor de F(30)? Para ello necesitamos que escribas un programa que reciba un número entero como parámetro 'n', y calcule el valor de la función recursiva F(n) vista anteriormente para este valor.

El valor introducido como argumento será un numero entero y la salida deberá tener 2 decimales (incluso si se trata de un numero entero).

## ENTRADA

- 10
- 20
- 30

## SALIDA

- 36.60
- 5340.34
- 58426080.00

## SOLUCION

Escrita en Java:

```
import java.util.Scanner;
class test
{
    public static float F(int n)
    {
        if (n<=0)
            return (float)1.0;
        else
            return (float)(((n/10.0) * F(n-1))+n);
    }

    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        n=(int)n;

        float respuesta=F(n);
        System.out.printf("%.2f", respuesta);
    }
}
```

## [4] ¿Cuánto queda? – 6 Puntos

Isabella Bird fue una exploradora inglesa que hizo, en el siglo XIX, diferentes viajes que la llevaron a Australia, América, Hawái, Japón, Corea, China, Vietnam, Singapur, malasia, India, Persia, Turquía y Marruecos. Isabella siempre estaba deseando volver a embarcarse (entonces se viajaba en barco) en otra de sus aventuras.

A Isabella le hubiera venido muy bien (si hubiera habido ordenadores) un programa que le dijera cuantos días le quedan para su próxima aventura. ¿Podrías programarle un programa que reciba una fecha y te diga cuantos días quedan para la misma?

En caso de que la fecha no sea válida o se trate de una fecha anterior a la de hoy el programa deberá mostrar el siguiente mensaje:

*Esa fecha no es valida*

**ENTRADA**

- 15/05/2016 06:00:00
- 15/05/2019 06:00:00
- 15/15/2019 06:00:00

**SALIDA**

- Esa fecha no es valida.
- Quedan 86 días
- Esa fecha no es valida.

**SOLUCION**

Escrita en Java:

```
import java.util.Scanner;
import java.time.*;
import java.text.*;
import java.util.*;
import java.time.temporal.ChronoUnit;

class test
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        String fecha=sc.nextLine();

        Date result;
        SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy kk:mm:ss");
        LocalDateTime localDate;
```

```
        sdf.setLenient(false);
        try{
            result =sdf.parse(fecha);
            localDate =
result.toInstant().atZone(ZoneId.systemDefault()).toLocalDateTime();
            int year=localDate.getYear();
            if (year<2019)
            {
                throw new Exception("Invalid Date");
            }
        }
        catch (Exception e)
        {
            //System.out.println(e);
            System.out.print("Esa fecha no es valida.");
            return;
        }

        LocalDateTime now = LocalDateTime.of(2019, 02, 17, 11, 12, 23);
        LocalDateTime travelDateTime =
LocalDateTime.of(localDate.getYear(),localDate.getMonthValue(),localDate.
getDayOfMonth(),localDate.getHour(),localDate.getMinute(),localDate.getSe
cond());

        LocalDateTime tempDateTime = LocalDateTime.from( now );

        long days = tempDateTime.until( travelDateTime, ChronoUnit.DAYS);
        tempDateTime = tempDateTime.plusDays( days );

        long hours = tempDateTime.until( travelDateTime,
ChronoUnit.HOURS);
        tempDateTime = tempDateTime.plusHours( hours );

        long minutes = tempDateTime.until( travelDateTime,
ChronoUnit.MINUTES);
        tempDateTime = tempDateTime.plusMinutes( minutes );

        long seconds = tempDateTime.until( travelDateTime,
ChronoUnit.SECONDS);

        System.out.print( "Quedan "+ days + " dias, " + hours + " horas,
" +  minutes + " minutos y " +
        seconds + " segundos.");
    }
}
```

## [5] Apunteeeeeen … Fuego! – 10 Puntos

Juana de Arco fue una heroína francesa famosa nacida en 1412 por haber provocado el sitio y posterior captura de la ciudad de Orleans durante la guerra de los 100 años. Para finalizar el sitio de la ciudad, Juana de Arco utilizó multitud de catapultas, conocidas allí como Trebuchets. Sin embargo, poner estos Trebuchets en posición no era una tarea fácil. Aparte de su enorme peso, que les permitían lanzar enormes piedras de hasta 30 kilos a distancias de hasta 500m.

Sin embargo, en la época de Juana de Arco no se conocían las fórmulas que permitían calcular distancias, grados y alturas de lanzamiento. A Juana de Arco le hubiera venido muy bien un programa como el que te proponemos. Se pide desarrollar un programa que permita calcular los grados a los que un trebuchet debe lanzar una piedra de 30kg para que alcance las murallas de la ciudad de Orleans a una altura de 5m desde una distancia que se lea por teclado.

Las fórmulas que se proponen son las siguientes:

1. Calculo de la distancia máxima a la que el trebuchet puede lanzar una piedra:

$$D_{max} = \frac{V^2 \cdot sin\,(2 \cdot \alpha_{lanzamiento})}{9.8}$$

- Es importante recalcar que la máxima distancia siempre se consigue a 45 grados.
- V es la velocidad de lanzamiento en m/s.
- La Distancia D obtenida está en metros.
- El ángulo $\alpha$ está en radianes (1 radian equivale a 57,29578 grados)

2. Altura de destino
   La segunda fórmula nos permite calcular la altura a la que estará la piedra, dada una distancia (que debe ser menor que la distancia máxima antes calculada).

$$Altura = D_{obj} \cdot tan(\alpha_{lanzamiento}) - \frac{9.8 \cdot D_{obj}}{2 \cdot V^2 \cdot cos^2(\alpha_{lanzamiento})}$$

- La altura obtenida es en metros.

Se dispone de Trebuchets que solo permiten lanzamientos a una velocidad de 50m/s. Dichos trebuchets se pueden calibrar para que lancen a distintos ángulos entre 0 y 180, pero únicamente con grados completos. El ángulo de lanzamiento no se podrá descomponer en minutos y segundos. Desarrollar un programa que lea la distancia y la altura de golpeo y nos diga con que ángulo debemos de lanzar la piedra.

Los valores de distancia se mostrarán con tres decimales.

En caso de introducir una distancia demasiado larga, se mostrará la frase “A esa distancia no llegas.”.

**ENTRADA**

- 240

## SALIDA

- Esa fecha no es valida.
- Quedan 86 días
- Esa fecha no es valida.

## SOLUCION

Escrita en Java:

```
import java.util.Scanner;
import java.text.*;
import java.util.*;

class test
{
    public static void main(String args[])
    {
        double grados, velocidad, distancia_objetivo, maxdistance,
verticalpos, altura, minheightdifference;
        int best_angle=-1;

        Scanner sc=new Scanner(System.in);
        //System.out.print("Grados: ");
        //grados=sc.nextDouble();
        //System.out.print("Velocidad (m/s): ");
        //velocidad=sc.nextDouble();
        velocidad=50.0;
        //System.out.print("Altura de Golpe (m): ");
        //altura=sc.nextDouble();
         altura=5.0;

        //System.out.print("Distancia al objetivo (m): ");
        distancia_objetivo=sc.nextDouble();
        DecimalFormat df = new DecimalFormat("###.###");

        maxdistance=Math.pow(velocidad,2)*Math.sin(2*
               Math.toRadians(45))/9.8;
        minheightdifference=altura;
       // System.out.println(df.format(maxdistance));
        if (distancia_objetivo<maxdistance)
        {
            for (int i=0; i<90; i++)
            {
                grados=(double)i;

verticalpos=distancia_objetivo*Math.tan(Math.toRadians(grados)) -
((9.8*distancia_objetivo*distancia_objetivo)/(2*velocidad*velocidad*Math.
cos(Math.toRadians(grados))*Math.cos(Math.toRadians(grados)) ));
```

```
                if (Math.abs(verticalpos-altura) < minheightdifference)
                {
                    minheightdifference=Math.abs(verticalpos-altura);
                    best_angle=i;
                }
            }

            //System.out.println("Lanzando el trebuchet con
"+df.format(best_angle)+" grados, alcanzaras el objetivo con
"+df.format(minheightdifference)+ " metros de margen de error.");
           System.out.println(df.format(best_angle));
        }
        else
        {
            System.out.println("A esa distancia no llegas.");
        }
   }
}
```

## [6] Lindas mariposas – 5 Puntos

Maria Merian (1647-1717) fue probablemente la primera naturalista de la historia. Era alemana y le apasionaban los insectos (que raro ¿verdad?).

Escribió publicaciones acerca de la metamorfosis de las mariposas desde su estado larvario a su imponente colorido final. Para aumentar sus conocimientos incluso promovió un viaje, financiado por ella misma, a Surinam donde describió muchos insectos y plantas.

Se pide escribir un programa que dibuje una mariposa con números. El programa recibirá el tamaño como parámetro y dibujará una mariposa tan grande como se le pida. El tamaño mínimo de la mariposa será 4 y el máximo será 9. En caso contrario deberá aparecer exactamente la siguiente frase:

*El tamaño de la mariposa debe estar comprendido entre 4 y 9.*

El aspecto de la mariposa de tamaño 9 será el siguiente:

```
1                 1
12                21
123               321
1234              4321
12345             54321
123456            654321
1234567           7654321
12345678          87654321
123456789987654321
12345678          87654321
1234567           7654321
123456            654321
12345             54321
1234              4321
123               321
12                21
1                 1
```

La mariposa tendrá por tanto una altura de 2N-1 y un ancho de 2N.

**ENTRADA**

- 3
- 5

**SALIDA**

- *El tamaño de la mariposa debe estar comprendido entre 4 y 9.*
- ```
  1           1
  12         21
  123       321
  1234     4321
  123455 4321
  1234     4321
  123       321
  12         21
  1           1
  ```

**SOLUCION**

Escrita en Java:

```
import java.util.Scanner;
import java.text.*;
import java.util.*;

public class test
{
public static void main(String[] args) {

    Scanner sc=new Scanner(System.in);
    int size=sc.nextInt();

    if (size<4 || size>9)
    {
        System.out.println("El tamaño de la mariposa debe estar
comprendido entre 4 y 9.");
    }else
    {

        int j;
        for (int i = 0; i <= size-1; i++) {
            for ( j = 1; j <= i + 1; j++) {
                System.out.print(j);
            }
            System.out.print(String.format("%"+((size + 1 - j)*2 !=
0 ? (size + 1 - j)*2 : "")+"s", ""));

            for ( j = i + 1; j >= 1; j--) {
```
```

```
                System.out.print(j);
            }
            System.out.println();

        }
        for (int i = size -1; i >= 1; i--) {
            for ( j = 1; j <= i ; j++) {
                System.out.print(j);
            }

            System.out.print(String.format("%"+((size + 1 - j)*2 !=
0 ? (size + 1 - j)*2 : "")+"s", ""));

            for ( j = i; j >= 1; j--) {
                System.out.print(j);
            }
            System.out.println();

        }
    }
}}
```

## [7] Resistencias equivalentes – 6 Puntos

Kathleen McNulty fue la primera mujer que empezó a trabajar, en 1945, en uno de los proyectos estrella de la armada americana: la computadora ENIAC. En aquella época los programadores tenían que saber más de electrónica que de programación puesto que los ordenadores funcionaban con válvulas de vacío, diodos, relés, resistencias y condensadores.

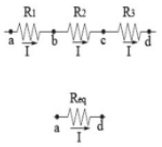
Kathleen más que conocimientos de C o Java (que por otra parte aún no existían) tenía que tener conocimientos de electrónica.

Se pide hacer un programa (que ya hubiera querido Kathleen tener delante) que calcule la resistencia eléctrica equivalente usando los siguientes datos como líneas de entrada.

- Primera línea, “s” o “p” en función de la configuración deseada, s = serie, p = paralelo.
- Las siguientes líneas sucesivas serán los valores de las resistencias en ohmios, en formato de punto flotante.
- La última línea estará marcada con una “f” en lugar de un número para señalizar que hemos acabado de introducir datos.

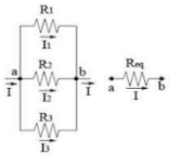
Recordemos algunas fórmulas de electrónica para calcular estas resistencias equivalentes en función de si estas están conectadas en serie o en paralelo.

Resistencia serie equivalente




$$R = R1 + R2 + R3 + \cdots$$

Resistencia paralela equivalente




$$\frac{1}{R} = \frac{1}{R1} + \frac{1}{R2} + \frac{1}{R3} + \cdots$$

Recordemos que en una configuración de resistencias paralelas donde una de ellas sea cero, es decir una conexión directa, la resistencia equivalente es cero. ¡Mucho cuidado con las divisiones por cero!

### ENTRADA

*Ejemplo 1:*

```
s
f
```

*Ejemplo 2:*

*p*
*1.5*
*3.8*
*47.5*
*f*

*Ejemplo 2:*

*s*
*10.0*
*5.2*
*0.0*
*3.4*
*f*

**SALIDA**

*Ejemplo 1:*

*0*

*Ejemplo 2:*

*1.05166*

*Ejemplo 3:*

*18.6*

## SOLUCION

Escrita en C++:

```
#include <iostream>
#include <string>

using namespace std;

static long double resistenciaTotal = 0.0;
```

```
long double calcularSerie(double Ra, double Rb)
{
    return Ra + Rb;
}

long double calcularParalelo(double Ra, double Rb)
{
    if ((Ra == 0.0) || (Rb == 0.0)) return (long double)0.0;
    return (long double)(1/((1/Ra)+(1/Rb)));
}

int main(int argc, char *argv[])
{
    std::string input;;
    cin >> input;

    if (input == "s")
    {
        cin >> input;
        while (input != "f")
        {
            resistenciaTotal = calcularSerie(resistenciaTotal,
std::stold(input));
            cin >> input;
        }
    }
    else if (input == "p")
    {
        bool primeraResistencia = true;
        cin >> input;
        while (input != "f")
        {
            if (primeraResistencia)
            {
                resistenciaTotal = std::stold(input);
                primeraResistencia = false;
            }
            else
            {
                resistenciaTotal = calcularParalelo(resistenciaTotal,
std::stold(input));
            }
            cin >> input;
        }
    }
    else{
        cout << "ERROR" << endl;
    }
```

```
    cout << resistenciaTotal << endl;
    return 0;
}
```

## [8] ¡El radio me desaparece! – 8 Puntos

Marie Curie es la única persona que ha conseguido ganar dos premios nobel diferentes en dos ramas diferentes (Física y Química) entre 1903 y 1910. Sus hallazgos fueron pioneros en el campo de la radiactividad. Descubrió dos nuevos elementos químicos radioactivos (el polonio y el radio), desarrolló técnicas para el aislamiento de isotopos radioactivos e investigó el uso de la radiactividad con fines medicinales.

Un problema que tenía Marie Curie era que los productos radiactivos se le desintegraban a tal ritmo que no tenía tiempo para realizar sus experimentos. Tras muchos estudios descubrió que el número de átomos restantes de un elemento tras un periodo de tiempo responde a la fórmula:

$$N = N_0 \cdot e^{-l \cdot t}$$

$$l = \frac{ln2}{t'}$$

- N es el número de átomos restantes
- $N_0$ es el número de átomos que se tenía inicialmente
- l es la constante radioactiva, diferente para cada elemento
- t es el número de días que han pasado
- t' es el tiempo de semidesintegración. Se trata del número de días que tardan la mitad de los átomos de un elemento en desintegrarse.

Marie trabajaba con el Radio-223, que tiene un tiempo de semidesintegración t' de 7,8 días. Cuando hacía un pedido mensual de Radio-223 le llegaban 10 mg de Radio, que correspondían con $5{,}067 \cdot 10^8$ átomos. Se pide desarrollar un programa que le ayude a Marie Curie a descubrir cuantos gramos de Radio-223 le quedarán después de X días.

El programa recibirá como parámetro el número de miligramos originales de Radio-223, y el número de días que se esperarán. El programa deberá generar una salida con el número de miligramos (con 3 decimales) que le restan a Marie Curie tras esos días de espera. En caso de introducir un número negativo de días o miligramos, el programa deberá mostrar el mensaje:

*Valores introducidos inválidos*

Nota: El número de miligramos se mostrará con 3 números decimales.

**ENTRADA**

- 11
  11
- 12
  30
- 11
  -1

**SALIDA**

- 4.139
- 0.834

- Valores introducidos inválidos

**SOLUCION**

En Java:

```
import java.util.Scanner;
import java.text.*;
import java.util.*;

class test
{
    public static void main(String args[])
    {
        double mg;
        int dias;
        Scanner sc=new Scanner(System.in);
        System.out.print("Miligramos de Radio-223: ");
        mg=sc.nextDouble();
        System.out.print("Dias que van a pasar: ");
        dias=sc.nextInt();

        if (mg<0 || dias <0)
        {
            System.out.println("Valores introducidos inválidos");
        }
        else
        {
            long atomos_originales=(long)(506700000*mg/10);
            double lambda=java.lang.Math.log(2)/7.8;

            long atomos_despues=(long)(atomos_originales *
java.lang.Math.exp(-lambda*dias));
            double available_miligrams=(atomos_despues * 10.0  /
506700000.0);

            System.out.printf("%5.3f", available_miligrams);
        }
   }
}
```

## [9] Ada Lovelace – 9 Puntos

Ada Lovelace fue una matemática, informática y escritora británica, célebre sobre todo por su trabajo con la denominada máquina analítica, probablemente la primera calculadora mecánica, inventada en 1837. Entre sus notas sobre la máquina, se encuentra lo que se reconoce hoy como el primer algoritmo destinado a ser procesado por una máquina, por lo que se la considera como la primera programadora de ordenadores en la historia.

Este primer algoritmo tenía como objetivo obtener los números de Bernoulli que son una sucesión de números racionales que se basan en la fórmula:

$$B[k] = -\sum_{i=0}^{k-1} \binom{k}{i} \cdot \frac{B[i]}{k+1-i}$$

Que saca los siguientes resultados:

| n | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| **B[n]** | 1 | −1/2 | 1/6 | 0 | −1/30 | 0 | 1/42 | 0 | −1/30 |

Nótese que, para todo n impar por encima de 3, B[n] siempre es 0.

El programa que hizo Ada Lovelace realizado con tarjetas, tenía el siguiente pseudocódigo:

```
main (parms):
   read n
   call bernouillis(n)
Fin main

void bernouillis(max):
   b =: []
   b[1] =: 1/6
   print 'B[1]=' + b[1]
   For n =: from 2 upto max
    k =: 2*n - 1
    n2 =: 2*n
    a0 =: - (n2 - 1) / ( 2* (n2 + 1) )
    a1 =: n2 / 2
    x =: a0 + (b[1] * a1)
    aj =: a1
```

```
For j =: from 3 upto k-1 step 2
  aj =* (n2 - (j-2)) / j
  aj =* (n2 - (j-1)) / (j + 1)
  x =+ (b[j] * aj)

b[k] =: -x
print "B[" k "]=" + b[k]
Fin Bernouillis
```

Hoy en día hay formas más eficientes de calcular esta serie. Pero es que, además, Ada hizo alguna trampa:

Es trivial calcular que para k=0, B[k]=1 y para k=1, B[1]=-1/2... y así lo sabía ella, por lo que comenzó a contar por el tercer valor y colocó en B[1] el valor 1/6, que realmente corresponde a B[2]. Es decir, calculó la serie de tal modo que los valores quedaban descolocados respecto a donde deberían quedar. Y, por tanto, para ella las posiciones a 0 eran los pares. Pero al margen de esto, su programa funciona perfectamente.

Se pide escribir un programa que obtenga la serie de Bernoulli usando este seudocódigo desarrollado por Ada Lovelace. El programa leerá la cantidad de números que se quieren calcular por teclado y mostrará esa cantidad de números de la serie de Bernoulli, pero SOLO las posiciones impares (que según el seudocódigo de Ada deben ser distintos de 0).

La anterior estructura será la del texto que se comprobará. Si el usuario introdujera un número inferior a 1, el programa simplemente leería otra vez el número (sin mensaje de ningún tipo).

Nota: Los valores de la serie que se muestren tendrán siempre 4 dígitos decimales.

**ENTRADA**

- 5

**SALIDA**

- B[1]=0.1666
  B[3]=-0.0333
  B[5]=0.0238
  B[7]=-0.0333
  B[9]=0.0757

SOLUCION

En Java:

```
import java.util.Scanner;
import java.util.concurrent.TimeUnit;
```

```
public class test{
    static Scanner sc=new Scanner(System.in);
    static double[] b;

    public static void main(String args[])
    {
        int num=-1;
        while (num<0)
        {
            num=sc.nextInt();
        }
        bernouilli(num);
    }

    static void bernouilli(int max)
    {
        b=new double[2*max];
        b[1]=(double)1/6;
        int k;
        double n2,a0, a1, aj,x;
        System.out.printf("B[1]=%3.4f\n",b[1]);
        for (int n=2; n<=max; n++)
        {
            k=(2*n)-1;
            n2=2*n;
            a0=(double)(-(n2-1.0)/((2.0)*(n2+1.0)));
            a1=n2/2;
            x=a0+(b[1]*a1);
            aj=a1;
            for (int j=3; j<=k-1; j=j+2)
            {
                aj *= (n2 - (j-2)) / j;
                aj *= (n2 - (j-1)) / (j + 1) ;
                x+=(b[j] * aj);
            }
           b[k] = -x;
           System.out.printf("B[%d]=%3.4f\n",k,b[k]);
        }
    }
}
```

## [10] XOR variable – 15 Puntos

Hedy Lamarr, fue una reconocida actriz y estrella de Hollywood de los años 30, fue la inventora del "espectro ensanchado por salto de frecuencia", tecnología inicialmente pensada para el envío de mensajes cifrados durante la segunda guerra mundial y que posteriormente ha sido utilizado en las comunicaciones GPS, Bluetooth y Wi-Fi.

A grandes rasgos el invento consiste en que la comunicación "salta" de frecuencia cada cierto tiempo, un tiempo y una nueva frecuencia que solo el emisor y el receptor conocen ya que la nueva frecuencia y el momento del salto se envían codificados dentro de la comunicación.

Nosotros os proponemos realizar una simulación de este salto de frecuencia, desarrollando un programa que encripte un texto en claro, haciendo un "salto" de clave cada cierto tiempo. Es decir, los primeros X caracteres se encriptarán con una clave, los siguientes Y con otra y los últimos Z caracteres con otra. Pero no tiene que limitarse a 3 saltos, pueden ser desde 0 hasta N-1, siendo N la longitud del mensaje.

La encriptación la vamos a hacer usando la operación XOR. La operación XOR sobre un número, con una clave nos da otro número diferente, pero si volvemos a hacer esa operación XOR sobre el número que resultó, volveríamos a tener el número original. Por ejemplo:

*XOR(13,clave)=5 || XOR(5,clave)=13*

La clave va a ser otro número, por ejemplo, el 7. Así XOR(13,7) =10 y XOR(10,7)=13, pero si hiciéramos XOR(10,4) no nos daría el valor 13 original. Entonces:

XOR(num_original, clave) =num_encriptado

XOR(num_encriptado, clave) =num_original

Esto lo podemos aplicar a la posición de cada letra. Así, la 'A' o la 'a', corresponderían al número 1 y la 'L' o la 'l' corresponderían al número 12. Si la letra es un número, cogeremos ese número y si no es ni letra, ni número, la clave será el número 27.

Todos los lenguajes de programación incluyen la XOR como una operación estándar, igual que la suma, la resta o la división.

Nosotros simularemos el cambio de frecuencia de Hedy Lamarr, cambiando la clave con la que hacemos la operación XOR. Ahora, para hacerlo más real, tenemos que poner un tiempo "aleatorio", variable en cada mensaje.

Bueno, pues el tiempo que el mensaje se mantendrá en cada "frecuencia", es decir, usando cada clave de cifrado, la obtendremos de la última letra en claro que hemos encriptado. Si se acaba un tiempo de frecuencia y la última letra que hemos encriptado es la 'c', el siguiente tiempo de frecuencia es 3. Ese 3 se refiere a 3 letras. Es decir, la nueva "frecuencia" se usará en las próximas 3 letras.

Como la clave depende de una letra anterior a la actual... ¿qué clave le ponemos a la PRIMERA letra? Bueno, se recibirá como parámetro inicial al programa.

Una vez explicado todo esto, se pide escribir un programa que, dado un mensaje en claro y una clave de cifrado inicial (como parámetros de entrada) nos encripte el mensaje en claro mediante este algoritmo, y ofrezca una salida con el mensaje codificado.

Vamos a ver un ejemplo con el mensaje "Acabaremos" y la clave inicial 8.

*Durante 1 letra cifro 'A' con XOR-8 y nos da 'I'  <-- La primera letra se cifra con la clave leída por teclado (en este caso 8). El tiempo de uso de esa clave es siempre 1 (letra). Al cifrar nos da 'I'.*

*Durante 1 letra cifro 'c' con XOR-9 y nos da 'j'  <-- El 1 es por la A mayúscula de la primera letra del mensaje, el 9 es por la I cifrada que obtuvimos antes. XOR(c,9) =j*

*Durante 3 letras cifro 'a' con XOR-10 y nos da k  <-- El 3 es por la c minúscula de la letra anterior del mensaje en claro. El 10 es por la j que obtuvimos en el XOR anterior. XOR(a,10) =k*

*Durante 3 letras cifro 'b' con XOR-10 y nos da 'h'*

*Durante 3 letras cifro 'a' con XOR-10 y nos da 'k'*

*Durante 1 letra cifro 'r' con XOR-11 y nos da 'y'*

*Durante 18 letras cifro 'e' con XOR-25 y nos da '|'*

*Durante 18 letras cifro 'm' con XOR-25 y nos da 't'*

*Durante 18 letras cifro 'o' con XOR-25 y nos da 'v'*

*Durante 18 letras cifro 's' con XOR-25 y nos da 'j'*

**En este caso, la salida de la frase "Acabaremos" cifrado con una clave  inicial de 8 es "**ljkhky|tvj".

## ENTRADA

- En un lugar de la mancha de cuyo nombre no quiero acordarme…
  13

## SALIDA

- Hf(}f(wn|zi~;wz;vzuxsz~;xnbt;bcan~i,bc,}yei~c,moc~hm~ai"55

## SOLUCION

Escrita en java:

```
import java.util.Scanner;
import java.util.concurrent.TimeUnit;

public class test{
    static Scanner sc=new Scanner(System.in);
    static char[] plaintext_chararray;
    static char[] replainedtext_chararray;
    static char[] encryptedText_chararray;
```

```
    static int encrypt_key;
    //char vocabulario={' ',0,1,2,3,4,5,6,7,8,9,'a','b'
    public static void main(String args[])
    {

        //String plainText="En un lugar de la mancha, no ha mucho tiempo
que vivia un hidalgo de los de lanza en astillero, adarga antigua, rocín
flaco y galgo corredor.";
        //String plainText="Acabaremos";
        System.out.print("Introduzca mensaje en claro: ");
        String plainText=sc.nextLine();
        plaintext_chararray=plainText.toCharArray();
        encryptedText_chararray=new char[plainText.length()];



        System.out.print("Introduzca la clave inicial de cifrado: ");
        encrypt_key=-1;
        while (encrypt_key<1 || encrypt_key>26)
        {
            encrypt_key=sc.nextInt();
        }
        int original_encrypt_key=encrypt_key;
        encriptar(plainText);
        //descifrar(original_encrypt_key);
    }

    public static char convert(char enc)
    {
        char val=0;
        if (enc>='0' && enc <'A')
        {
            val=(char)(enc-'0'+1);
        } else if (enc>='A' && enc <'a')
        {
```

```
            val=(char)(enc-'A'+1);
        } else if (enc>='a' && enc <='z')
        {
            val=(char)(enc-'a'+1);
        }else
        {
            val=27;
        }

        return val;
    }

    public static void encriptar(String plainText)
    {
        //El primer salto sale de la primera letra en claro
        int encrypt_hop=convert(plaintext_chararray[0]);
        //Hacemos el primer XOR
        encryptedText_chararray[0]=(char) ((plaintext_chararray[0] ^
encrypt_key));
        //La primera clave variable sale del primer caracter cifrado con
la clave original
        encrypt_key=convert(encryptedText_chararray[0]);

        int position=1;
        while (position < plainText.length())
        {
            encryptedText_chararray[position]=(char)
((plaintext_chararray[position] ^ encrypt_key));
            //System.out.println("Durante "+encrypt_hop+" cifro
"+plaintext_chararray[position]+" con XOR-"+encrypt_key +" y nos da
"+encryptedText_chararray[position]);
            encrypt_hop--;
            if (encrypt_hop==0)
            {
```

```
                //Cuando se acaba la venta de cifrado recalculamos la
clave y el hop
                encrypt_hop=convert(plaintext_chararray[position]);

                encrypt_key=convert(encryptedText_chararray[position]);
            }
            position++;
        }

        System.out.println(new String(encryptedText_chararray));
    }

    public static void descifrar(int original_encrypt_key)
    {
        int text_length=new String(encryptedText_chararray).length();
        replainedtext_chararray=new char[text_length];

        //desciframos la primera letra con la clave introducida por
teclado
        replainedtext_chararray[0]=(char) ((encryptedText_chararray[0] ^
original_encrypt_key));

        //la nueva clave de cifrado es la ultima letra cifrada.
        int encrypt_key=convert(encryptedText_chararray[0]);
        //El nuevo salto de cifrado es la ultima letra en claro
        int encrypt_hop=convert(replainedtext_chararray[0]);

        int position=1;
        while (position < text_length)
        {
            replainedtext_chararray[position]=(char)
((encryptedText_chararray[position] ^ encrypt_key));
            //System.out.println("Durante "+encrypt_hop+" DEScifro
"+encryptedText_chararray[position]+" con XOR-"+encrypt_key +" y nos da
"+replainedtext_chararray[position]);
```

```
            encrypt_hop--;
            if (encrypt_hop==0)
            {
                //Cuando se acaba la venta de cifrado recalculamos la
clave y el hop
                encrypt_hop=convert(replainedtext_chararray[position]);

                encrypt_key=convert(encryptedText_chararray[position]);
            }
            position++;
        }
    }
}
```

## [11] Coaliciones – 11 Puntos

Elizabeth Garrett Anderson fue una médica y política inglesa nacida en 1836. Fue la primera mujer en obtener los títulos de doctora y cirujana. Fue cofundadora del primer hospital gestionado por mujeres, la primera mujer en ser elegida alcaldesa en Inglaterra y la primera magistrada de Inglaterra. Elizabeth fue también una muy activa sufragista por el voto de la mujer, todo ello antes de 1902.

Elizabeth era conocida por ser una fuerte defensora de la democracia e intentaba siempre hacer política con aquellos que no pensaban como ella.

Se pide desarrollar un programa que, dado un parlamento, determine aquellas coaliciones que permitirían gobernarlo sumando mayoría absoluta.

El programa recibirá inicialmente el número de escaños del parlamento para, a continuación, ir leyendo los diputados obtenidos en las elecciones por los distintos partidos políticos. Dicho listado de partidos políticos y sus diputados comenzará por los partidos situados más a la izquierda en el arco parlamentario, para acabar introduciendo los partidos situados más a la derecha del arco.

El programa no dejará de leer resultados hasta que se alcance el número de diputados indicados en la primera línea y si por lo que fuera nos pasáramos nos mostraría el siguiente mensaje:

*Resultados inválidos*

Si el listado suma 350 buscará aquellas coaliciones de 3 o menos partidos que puedan gobernar el hemiciclo. Los 3 partidos tienen que ser afines y por tanto tendrán que ser estar colocados de forma consecutiva en el listado introducido. En el ejemplo anterior, la mayoría absoluta se encuentra en 176 diputados. El programa mostrara todas las posibles coaliciones.

Importante:

- La salida mostrará las coaliciones con menor número de partidos políticos primero.
- Deja una línea en blanco entre resultados y coaliciones
- Para este ejercicio se va a suponer que la estructura de las líneas introducidas es correcta (primera línea un numero entero positivo y en las siguiente un partido = número entero).

**ENTRADA**

- 350
  Partido_republicano=202
  Partido_conservador=148
- 650
  Partido_los_verdes=35
  Partido_laborista=262
  Partido_conservador=318
  Partido_liberal_democrata=23
  Partido_independentistas_escoceses=12

## SALIDA

- Partido_republicano suman 202 diputados.
  Partido_republicano + Partido_conservador suman 350 diputados.
- Partido_laborista + Partido_conservador suman 580 diputados.
  Partido_conservador + Partido_liberal_democrata suman 341 diputados.
  Partido_los_verdes + Partido_laborista + Partido_conservador suman 615 diputados.
  Partido_laborista + Partido_conservador + Partido_liberal_democrata suman 603 diputados.
  Partido_conservador + Partido_liberal_democrata + Partido_independentistas_escoceses suman 363 diputados.

## SOLUCION

Escrita en Java:

```
import java.io.*;
import java.util.*;

public class test{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);

        int diputados=sc.nextInt();
        int suma=0;
        String resultado;
        String[] parts=new String[3];
        String[] Partidos=new String[16];
        int[] Asientos=new int[16];

        int i=0;
        while (suma<diputados)
        {
                resultado=sc.next();
                parts=resultado.split("=");
                Partidos[i]=parts[0];
                Asientos[i]=Integer.parseInt(parts[1]);
                suma=suma+Asientos[i++];
        }
```

```
        int num_partidos=i;
        //Separador
        System.out.println("");

        int mayoria=(int)(Math.round(diputados/2)+1);

      //  System.out.println("La mayoria es "+mayoria);

        //Partidos unicos
        for (i=0; i<num_partidos; i++)
        {
                if (Asientos[i]>=mayoria)
                {
                    System.out.println(Partidos[i]+" suman
"+Asientos[i]+" diputados.");
                    break;
                }
        }

        //Coaliciones dobles
        for (i=0; i<num_partidos-1; i++)
        {
                if ((Asientos[i]+Asientos[i+1])>=mayoria)
                {
                    suma=Asientos[i]+Asientos[i+1];
                    System.out.println(Partidos[i]+" + "+Partidos[i+1] +"
suman "+suma+" diputados.");
                }
        }

        //Coaliciones triples
        for (i=0; i<num_partidos-2; i++)
        {
                if ((Asientos[i]+Asientos[i+1]+Asientos[i+2])>=mayoria)
                {
                    suma=Asientos[i]+Asientos[i+1]+Asientos[i+2];
```

```
                    System.out.println(Partidos[i]+" + "+Partidos[i+1] +
" + " +Partidos[i+2]+ " suman "+suma+" diputados.");
                }
            }
        }
}
```

## [12] Primos-RSA – 8 Puntos

Marie-Sophie Germain una matemática, física y filósofa francesa nacida en 1776.Fue una de las pioneras de la teoría de elasticidad hizo importantes contribuciones a la teoría de números; uno de sus trabajos más importantes fue el estudio de los que posteriormente fueron conocidos como números primos de Sophie Germain (números primos cuyo doble incrementado en una unidad es también un número primo).

Los números primos de Sophie Germain son utilizados en criptografía, ya que son los factores generalmente usados para obtener claves RSA. Asimismo, son usados para la generación de números pseudoaleatorios.

Se pide escribir un programa que imprima los números primos de Sophie Germain menores que uno dado.

Para poder realizar este ejercicio, se aconseja que usar el algoritmo de *sieve* para calcular los números primos. El algoritmo de *sieve* genera un array que almacena si la posición del array es un número primo o no. Algo asi:

[SI][SI][SI][NO][SI][NO][SI][NO][NO][NO]...

El pseudocódigo de dicho algoritmo es:

```
sieve(n)
 rellenar primo[] a true
 for p=2 hasta p*p<=n :
    si (primo[p]) {
       for i=2p hasta n:
            primo[i] = false;
            i=i+p
       }
    }
 }
```

Por tanto en este ejercicio tendréis que escribir un programa que imprima todos los números primos de Sophie Germain menores que un número n leído por teclado.

Un número primo es considerado un primo de Sophie Germain si también (2*p) +1 es también un número primo. Así, por ejemplo, el 11 es un número primo y 11*2+1=23 también es un número primo, por lo que 11 es un número primo de Sophie Germain.

Nota: La separación de los números primos se realizará con un espacio en blanco.

**ENTRADA**

- 25

**SALIDA**

- 2 3 5 11 23

## SOLUCION

Escrita en Java:

```
import java.io.*;
import java.util.*;

public class test{
    static void sieve(int n, boolean prime[])
    {
        for (int p = 2; p * p <= n; p++) {
            if (prime[p] == true) {
                for (int i = p * 2; i < n; i += p)
                    prime[i] = false;
            }
        }
    }

    static void printSophieGermanNumber(int n)
    {
        boolean prime[]=new boolean[2 * n + 1];
        Arrays.fill(prime,true);
        sieve(2 * n + 1, prime);

        for (int i = 2; i < n; ++i) {
            if (prime[i] && prime[2 * i + 1])  System.out.print( i + "
");
        }
    }

    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int num=-1;
        while (num<0) num=sc.nextInt();

        printSophieGermanNumber(num);
    }
}
```

## [13] Sectoriza sista! – 7 Puntos

Phoebe Sarah Hertha Ayrton, fue una ingeniera, matemática, física e inventora británica nacida en 1854. Fue galardonada con la Medalla Hughes de la Royal Society por sus estudios del arco eléctrico y la formación de ondas de las dunas y las olas del mar. Aunque trabajó mucho con arcos y formas sinusoidales, Hertha comenzó con las líneas rectas.

No en vano, en 1884 publicó su primera patente, un instrumento de dibujo de ingeniería para dividir una línea en cualquier número de partes iguales y para ampliar y reducir figuras. Fue su primer invento importante (Tuvo 26 patentes a lo largo de su vida). Aunque probablemente su uso principal fuera para artistas, también era útil para arquitectos e ingenieros.

Se pide escribir un programa que reciba una línea recta (escrita mediante guiones) por parámetro, y también el número de partes en que se quiere dividir la línea. El programa, si es posible, deberá dividir dicha línea recta en tantas partes como se pide (el separador será un símbolo '|'). Si no es posible hacerlo directamente, extienda la línea para poder hacer finalmente esa división. Al extender una línea, se pide hacerlo hasta el Mínimo Común Múltiplo entre la longitud de la línea y el numero de divisiones solicitado.

Para explicarlo, lo mejor es poner unos ejemplos:

**Ejemplo 1**: En este ejemplo se proporciona una línea de recta de 18 guiones y el número tres. Se pide dividir la línea de 18 guiones en 3 partes iguales. El separador es el símbolo "|"

```
------------------

3

------|------|------
```

Ejemplo 2: En este caso la línea original es de 11 guiones y se pide dividirla en tres partes. Como no es directamente posible hacerlo, la solución es extender la línea. Por tanto, lo extenderemos hasta una línea de 33 elementos con 3 partes de 11.

```
-----------

3

-----------|-----------|-----------
```

Ejemplo 3: En este caso la línea original es de 10 elementos y se pide dividirla en 8 partes iguales. Hay que extenderla, pero se debe extender a la mínima longitud necesaria. En este caso a longitud 40:

---

8

-----|-----|-----|-----|-----|-----|-----|-----

Nótese que la longitud de la línea inicial puede ser más corta que las partes que se quiere dividir.

En este ejercicio vamos a suponer que los datos de entrada son siempre correctos. Siempre se introducen guiones en la primera línea y un número entero mayor que cero en la segunda.

**ENTRADA**

- ----------
  3

**SALIDA**

- ------|------|------

**SOLUCION**

Escrita en Java:

```
import java.io.*;
import java.util.*;

public class test{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);

        String linea=sc.next();
        int longitud_linea=linea.length();

        int partes=sc.nextInt();
        if (longitud_linea%partes ==0)
        {
            //Se puede dividir
            int cada_parte=longitud_linea/partes;
            for (int i=0; i<partes; i++)
            {
```

```
                for (int j=0; j<cada_parte; j++)
                {
                    System.out.print("-");
                }
                System.out.print("|");
            }
        }
        else
        {
            int k=0;
            for (k=longitud_linea; k<=longitud_linea*partes;
k+=longitud_linea)
            {
                    if (k%partes==0) break;
            }

            int cada_parte=k/partes;

            for (int i=0; i<partes; i++)
            {
                for (int j=0; j<cada_parte; j++)
                {
                    System.out.print("-");
                }
                if (i<partes-1) System.out.print("|");
            }
        }
    }
}
```

## [14] Apollo 11 y las importancia de las prioridades – 18 Puntos

El próximo 20 de Julio, se cumplen 50 años del primer alunizaje tripulado, el de la misión Apollo 11.

Lo que no todo el mundo sabe, es que Margaret Hamilton, ingeniera de la NASA, fue la responsable de crear el software de navegación "on-board" del programa Apollo, el cual permitió que la humanidad pusiera un pie en la Luna en 1969.

Durante el descenso propulsado a la superficie lunar del módulo lunar Eagle, se sucedieron, entre otros imprevistos, las alarmas 1201 y 1202 en la computadora de a bordo que complicaron especialmente una fase de vuelo que se acometía por primera vez en la historia, y que ya era, de por sí, la más crítica y difícil de la misión.

Debido a un descuido en las especificaciones de diseño del sistema de alimentación eléctrica y una serie de desafortunadas decisiones tomadas durante la fase de descenso, el sistema eléctrico del radar enviaba interrupciones constantes a la computadora de guiado, lo que hacía que la computadora se saturara cada vez que su límite de procesamiento era rebasado.

Afortunadamente, la computadora de guiado poseía un diseño muy robusto. Su sistema ejecutivo lanzaba tareas siguiendo un sistema de prioridades preestablecido por el que, en caso de saturación, la computadora descartaba tareas no esenciales de modo que pudiera seguir manteniendo el vuelo de la nave de forma segura.

Gracias a que Hamilton con su equipo tuvieron en cuenta cómo resolver un problema de este tipo, la misión terminó siendo un éxito.

En este ejercicio vamos a emular la hazaña conseguida hace 50 años, diseñando un programa que ordene tareas por prioridades.

El programa recibirá como entrada una secuencia de números entre 1 y 64 separados por comas, donde cada número representa a una tarea. Por ejemplo:

1, 48, 31, 50, 33, 16, 63, 18

El programa deberá generar una salida ordenando las tareas, de mayor a menor prioridad, teniendo en cuenta los siguientes criterios:

- Las tareas de mayor prioridad serán aquellas cuyo número sea potencia de dos ($2^n$, es decir 2,4,8...).
- Las tareas con nivel de prioridad media serán aquellas con número par, que no sean potencia de dos.
- Las tareas con nivel de prioridad bajo serán aquellas cuyo número sea primo. Se puede utilizar el algoritmo de Sieve, explicado en otro ejercicio, para obtener si un numero es o no primo.
- Por último, todas las restantes tareas tendrán un nivel de prioridad muy bajo.

- En caso de que varias tareas tengan la misma prioridad, los números más bajos representan una mayor prioridad.

Los números de salida siempre deben mostrarse con dos dígitos (Es decir el 3 se mostrará como 03).

El ejemplo anterior debería generar la siguiente salida:

01 16 18 48 50 31 33 63

## ENTRADA

*Ejemplo 1:*

*1, 48, 31, 50, 33, 16, 63, 18*

*Ejemplo 2:*

*5, 15, 4, 23, 48, 31, 11, 2*

## SALIDA

*Ejemplo 1:*

*01 16 18 48 50 31 33 63*

*Ejemplo 2:*

*02 04 48 05 11 23 31 15*

## SOLUCION

Escrita en C++:

```
#include <functional>
#include <queue>
#include <vector>
#include <iostream>
#include <map>
#include <iomanip>
#include <cstdlib>
#include <sstream>
```

```
#include "t_t.h"

// Priority values (zero is higher)
enum Priority {
    HIGH,       // power of two
    MID,        // odd
    LOW,        // prime
    LOWEST      // remaining
};

// Parse input sequence string and return a vector with all numeric
values
std::vector<size_t> getSequence(const std::string &s, char delimiter)
{
    std::vector<size_t> values;
    std::string token;
    std::istringstream inputStream(s);
    while (std::getline(inputStream, token, delimiter)) {
        values.push_back(std::stoi(token));
    }
    return values;
}

// Print a sequence in format 00 01 02 03...
template<typename T>
void printSequence(T& s) {
    while(!s.empty()) {
        std::cout << std::setw(2) << std::setfill('0') << s.top() << " ";
        s.pop();
    }
    std::cout << '\n';
}

// Return true if number is power of two
bool isPowerOfTwo(size_t number) {
    return (number != 0) && ((number & (number - 1)) == 0);
}

// Return true if number is even
bool isEven(size_t number) {
    return (number % 2 == 0);
}

// Return true if number is prime
bool isPrime(size_t number) {
    if (number <= 1) return false;
    for (size_t i = 2; i < number; ++i)
        if (number % i == 0) return false;
```

```
    return true;
}

// Get the priority associated to a number
// Priority follows the following rules:
// 1st -> power of two
// 2nd -> even
// 3rd -> prime
// 4th -> remaining
size_t getPriority(size_t number) {
    return isPowerOfTwo(number) ? Priority::HIGH :
           isEven(number) ? Priority::MID :
           isPrime(number) ? Priority::LOW : Priority::LOWEST;
}

// Entry point
int main() {

    // Parse input line
    std::string line;

    std::getline(std::cin, line);

    if (line.empty()) {
        std::cerr << "Wrong input\n";
        return EXIT_FAILURE;
    }

    //
    std::vector<size_t> sequence = getSequence(line, ',');

    // Create the comparer
    // The comparer return the higher priority value of each pair of
numbers from
    // the sequence, based on the priorities returned by getPriority
function.
    // In addition, when two numbers have the same priority, smaller
values come first
    constexpr auto comparer = [](size_t lhs, size_t rhs) {
        size_t priorityLhs = getPriority(lhs);
        size_t priorityRhs = getPriority(rhs);

        return (priorityLhs > priorityRhs) ? true :
               (priorityLhs == priorityRhs) ? lhs > rhs : false;
    };

    // Create the priority queue
    // The priority queue uses the comparer bellow to sort the numbers
being inserted into it
```

```
    std::priority_queue<size_t, std::vector<size_t>, decltype(comparer) >
sortedSequence(comparer);

    // Insert values into the priority queue
    for (const auto& value : sequence) {
        sortedSequence.push(value);
    }

    // Print the sorted sequence
    printSequence(sortedSequence);

    return EXIT_SUCCESS;
}
```

## [15] El corta-pega genético – 5 Puntos

Emmanuelle Charpentier y Jennifer Doudna construyeron la hoy famosa técnica del corta-pega genético, que se presentó a la comunidad científica en 2012.

En la naturaleza, el mecanismo CRISPR/Cas9 destruye a los invasores cortando su ADN con la enzima Cas9 que actúa de tijera molecular. En el laboratorio, el ADN vírico que en CRISPR sirve para reconocer al enemigo, es sustituido por otro fragmento guía, que lleva las tijeras a una región específica del genoma. Se obtiene así un método que corta el ADN con altísima precisión y además lo vuelve a pegar, introduciendo secuencias nuevas si se desea.

Se pide crear un programa que utilice un algoritmo similar al mecanismo CRISP/Cas9. ¿Podrías crear un código que recibiendo dos secuencias de ADN, uno completamente saludable y otro infectado por el virus de la gripe, detectara el fragmento de ADN infectado y copiara ese fragmento del ADN saludable al ADN infectado?.

Para simplificar, representaremos el ADN como 0's y 1's.

Ten en cuenta que en la entrada:

- El tamaño del ADN es 8. Por tanto, la entrada será una secuencia de ocho 0's y 1's separados por espacios.
- Se introducirán dos vectores de entrada, la primera línea se refiere al vector del ADN saludable y la segunda línea al vector del ADN infectado.

La salida mostrará:

- El ADN saludable con el mensaje: "El ADN saludable es:{ ... }". Los puntos suspensivos se sustituirán por el ADN saludable.
- El ADN infectado con el mensaje "El ADN infectado por el virus es: { ... }".
- El fragmento de ADN infectado (diferente al ADN saludable) y la posición en la que empieza y en la que termina este fragmento (Ver el ejemplo siguiente).
- El ADN infectado una vez hemos reemplazado el fragmento infectado por el correspondiente fragmento del vector saludable (Ver el ejemplo siguiente).

Lo mejor es ver un ejemplo. Para la siguiente posible entrada (por teclado):

0 1 1 0 1 0 1 0

0 1 1 0 0 1 0 0

Se desea la siguiente salida:

El ADN saludable es: { 0 1 1 0 1 0 1 0 }

El ADN infectado por el virus es: { 0 1 1 0 0 1 0 0 }

El fragmento ADN infectado va desde la base 4 a la base 7: { 0 1 0 }

Ahora, el ADN infectado por la gripe está curado: { 0 1 1 0 1 0 1 0 }

## ENTRADA

*Ejemplo 1:*

*0 1 1 0 1 0 1 0*

*0 1 1 0 0 1 0 0*

## SALIDA

*Ejemplo 1:*

*El ADN saludable es: { 0 1 1 0 1 0 1 0 }*

*El ADN infectado por el virus es: { 0 1 1 0 0 1 0 0 }*

*El fragmento ADN infectado va desde la base 4 a la base 7: { 0 1 0 }*

*Ahora, el ADN infectado por la gripe está curado: { 0 1 1 0 1 0 1 0 }*

## SOLUCION

Escrita en C++:

```
#include <iostream>
#include <vector>
#include <fstream>

int main()
{
	int ADNSize = 8;
	std::vector<int> healthyADN;
	std::vector<int> fluVirusADN;
```

```
std::ifstream input_file("input.txt");

if(!input_file.is_open())
      std::cerr << "error abriendo fichero";

int gen;
for (int i = 0; i < ADNSize; i++)
{
      input_file >> gen;
      healthyADN.push_back(gen);
}
for (int i = 0; i < ADNSize; i++)
{
      input_file >> gen;
      fluVirusADN.push_back(gen);
}

input_file.close();

std::cout << "El ADN saludable es: { ";
for( int i = 0; i < healthyADN.size(); i++)
{
      std::cout << healthyADN[i] << " ";
}
std::cout << "}" << std::endl;

std::cout << "El ADN infectado por el virus es: { ";
for( int i = 0; i < fluVirusADN.size(); i++)
{
      std::cout << fluVirusADN[i] << " ";
}
std::cout << "}" << std::endl;
```

```
	int initPositionWrongADN = 0;
	int endPositionWrongADN = ADNSize;

	bool initPosition = true;
	bool endPosition = false;

	for( int i = 0; i < ADNSize; i++)
	{
		if( fluVirusADN[i] != healthyADN[i])
		{
			if(initPosition)
			{
				initPositionWrongADN = i;
				initPosition = false;
				endPosition = true;
			}
		}
		else
		{
			if(endPosition)
			{
				endPositionWrongADN = i;
			}
		}
	}

	std::cout << "El fragmento ADN infectado va desde el gen " <<
initPositionWrongADN << " al gen " << endPositionWrongADN << " : { ";
	for ( int i = initPositionWrongADN; i < endPositionWrongADN; i++)
	{
		std::cout << fluVirusADN[i] << " ";
	}
	std::cout << "} " << std::endl;
```

```
	for ( int i = initPositionWrongADN; i < endPositionWrongADN; i++)
	{
		fluVirusADN[i] = healthyADN[i];
	}

	std::cout << "Ahora, el ADN infectado por la gripe está curado: { ";
	for( int i = 0; i < fluVirusADN.size(); i++)
	{
		std::cout << fluVirusADN[i] << " ";
	}
	std::cout << "}" << std::endl;

	return 0;
}
```

## [16] Teorema en el espejo – 3 Puntos

Emmy Noether, conocida como la madre del álgebra moderna, desarrolló un teorema calificado como el teorema "más bello del mundo" y hoy en día es completamente fundamental para comprender toda la física más sofisticada.

Como el teorema de Noether, en su esencia, trata de la simetría, vamos a homenajear su trabajo con un pequeño y sencillo programa. ¡A crear simetría!.

Escribe un programa que:

- Te pida por pantalla que se escriba una frase.

- Al introducir la frase y dar a enter, el programa mostrará la frase escrita seguida de la misma, pero del revés. La separación entre la frase escrita del derecho y del revés será un espacio en blanco.

### ENTRADA

*Ejemplo 1:*

*Emmy Noether fue el genio matemático creativo más importante que haya existido*

### SALIDA

*Ejemplo 1:*

*Emmy Noether fue el genio matemático creativo más importante que haya existido oditsixe ayah euq etnatropmi sám ovitaerc ocitámetam oineg le euf rehteoN ymmE*

### SOLUCION

Escrita en C++:

```
#include <stdio.h>
#include <stdlib.h>

// Function to reverse a string
char* reverse(char* string) {
      char *reversed = (char*)malloc(1000*sizeof(char));
      // We get the size of the string using the string termination '\0'
```

```
	int size = 0;
	while (string[size] != '\0'){
		size++;
	}
	// Doing the inversion
	for (int i = 0; i < size; i++) {
		reversed[i] = string[size - (i + 1)];
	}
	// Setting the string termination
	reversed[size] = string[size];

	return reversed;
}

int main(int argc, char** argv) {

	char *original = (char*)malloc(1000*sizeof(char));

	// We get a string from the console
	printf("Enter a phrase to reverse it: ");
	// We use [^\n] to scan until we press enter
	scanf("%[^\n]s", original);

	printf("You create simmetry!! \n");
	printf("%s %s\n", original, reverse(original));

	return 0;
}
```

## [17] Show me the money (o toma el dinero y corre) – 7 Puntos

Maggie Lena Walker fue la primera mujer directora de un banco en 1902. En concreto se trató del St. Luke Penny Savings Bank de Virginia en EEUU. Posteriormente continuó en el consejo de administración del banco hasta 1915.

Al banco a menudo venían empresarios pidiendo préstamos. Para ello tenían que rellenar una solicitud.
En esas solicitudes, los empresarios detallaban hasta el céntimo, pero entonces no había tarjetas de crédito y el dinero se manejaba en efectivo. Por lo que había que dárselo a cada uno en efectivo, y se necesita saber cuántas monedas y billetes de cada tipo eran necesarias para poder pagarles y que puedan seguir invirtiendo.

Y aquí es donde entramos nosotros, vamos a ayudarle a Maggie un poquito escribiendo un programa que dada una cantidad de dinero (con dos decimales, que los céntimos importan) vamos a devolver como salida la mejor combinación de billetes y monedas con las que se puede pagar, así que:

Para facilitar las cosas vamos a suponer que estamos hablando de Euros.

El programa pedirá un número con hasta dos dígitos decimales. Por ejemplo:

123.56

Y mostrará el número de billetes de 500€, de 200€, de 100€, etc. así como las monedas de 2€, de 1€, etc. Para poder pagar la cantidad solicitada.

**ENTRADA**
*Ejemplo 1:*
*123.23*

*Ejemplo 2:*
*4532178.56*

**SALIDA**
*Ejemplo 1:*
*El cambio que necesitas para pagar 123.23 es el siguiente:*
*Billetes de 500 euros: 0*
*Billetes de 200 euros: 0*
*Billetes de 100 euros: 1*
*Billetes de 50 euros: 0*
*Billetes de 20 euros: 1*
*Billetes de 10 euros: 0*
*Billetes de 5 euros: 0*
*Monedas de 2 euros: 1*
*Monedas de 1 euro: 1*
*Monedas de 50 céntimos: 0*
*Monedas de 20 céntimos: 1*
*Monedas de 10 céntimos: 0*
*Monedas de 5 céntimos: 0*
*Monedas de 2 céntimos: 1*

*Monedas de 1 céntimo: 1*

*Ejemplo 2: El cambio que necesitas para pagar 4532178.56 es el siguiente:*
*Billetes de 500 euros: 9064*
*Billetes de 200 euros: 0*
*Billetes de 100 euros: 1*
*Billetes de 50 euros: 1*
*Billetes de 20 euros: 1*
*Billetes de 10 euros: 0*
*Billetes de 5 euros: 1*
*Monedas de 2 euros: 1*
*Monedas de 1 euro: 1*
*Monedas de 50 céntimos: 1*
*Monedas de 20 céntimos: 0*
*Monedas de 10 céntimos: 0*
*Monedas de 5 céntimos: 1*
*Monedas de 2 céntimos: 0*
*Monedas de 1 céntimo: 1*

## SOLUCION

<NO DISPONIBLE>

## [18] Mary Anderson y los limpiaparabrisas– 9 Puntos

Imaginaos que estamos en 1903, de viaje en Nueva York y tomamos un tranvía. Es invierno y está lloviendo aguanieve.

El viaje se hace interminable, el conductor tiene que bajarse todo el rato para limpiar el cristal delantero porque se ensucia mucho.

Pero a nadie le importa.

Bueno, a alguien sí. Justo al lado nuestro está Mary Anderson. Y se ve que tiene cara de estar pensando en algo……..

Dos años más tarde en la oficina de patentes patenta el primer limpiaparabrisas.

Claro, era manual, pero se accionaba desde dentro. No había que salir fuera del coche a limpiar el cristal.

Han pasado más de 100 años y el mecanismo no ha cambiado tanto. Pero ahora lo podemos controlar con un ordenador.

Así que nuestro próximo problema va de limpiaparabrisas. Como no tenemos uno a mano para controlarlo haremos una simulación del movimiento del motor de un limpiaparabrisas.

Vamos a hacer un programa que nos dirá en qué posición están las escobillas en cada momento. Cada vez que introduzcamos una línea (con cualquier letra) será como si hubiera pasado un segundo y el programa escribirá la posición del parabrisas en ese momento.

Las escobillas de los limpiaparabrisas van desde una posición inicial (será la posición 0) hasta una posición final (como hace un giro, supongamos que se mueve 100 grados, la posición final será 100) y después vuelve hasta la posición inicial y así continuamente.

La velocidad del limpiaparabrisas será de 15 grados por segundo. Inicialmente estará parado en su posición inicial.

Cuando la línea introducida tenga la letra ‘a’ empezará a moverse. Y se mantendrá en marcha hasta que se introduzca una línea con la letra ‘p’.

Cuando queramos terminar pondremos una línea con la letra ‘s’.

Cuidado, no nos podemos pasar por encima de la posición final ni por debajo de la posición inicial.

¿Ya lo tenéis?

¡¡Enhorabuena!! Vamos a hacerlo un poco mejor.

Cuando le decimos que pare, tendrá que seguir moviéndose hasta que llegue a la posición inicial. Si no, queda muy feo que se queden parados en mitad del cristal

Ahora vamos a ponerle velocidad variable... que a veces llueve mucho y otras muy poco.

Cuando la línea sea '+' incrementaremos la velocidad en 5 grados por segundo y cuando sea '-' se bajará la velocidad en 5 grados por segundo.

Pero tiene truco, la velocidad sólo se cambia cuando se pasa por la posición inicial.

Ah, y la velocidad máxima es 30 grados por segundo y la mínima es 5 grados por segundo.

## ENTRADA

*Ejemplo 1:*

*t*
*t*
*t*
*s*

*Ejemplo 2:*
*t*
*t*
*a*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*s*

*Ejemplo 3:*
*t*
*a*
*p*
*t*

*t*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*t*
*s*

**SALIDA**

*Ejemplo 1:*

*0*
*0*
*0*
*0*

*Ejemplo 2:*
*0*
*0*
*15*
*30*
*45*
*60*
*75*
*90*
*100*
*85*
*70*
*55*
*40*
*25*
*10*
*0*
*15*
*30*

*Ejemplo 3:*
*0*
*15*
*30*
*45*

*60*
*75*
*90*
*100*
*85*
*70*
*55*
*40*
*25*
*10*
*0*
*0*

## SOLUCION

<NO DISPONIBLE>

## [19] Donde está mi planeta? – 3 Puntos

Ellen Harding Baker fue una profesora que, en 1876, para impartir sus clases y conferencias sobre astronomía, tuvo una brillante idea: bordar un quilt (textil multicapa) representando el Sistema Solar.

Para ello en su quilt bordó los 8 planetas en su orden: Mercurio, Venus, Tierra, Marte, Júpiter, Saturno, Urano y Neptuno.

Ellen Harding Baker tardó 7 años en elaborar el quilt. En honor al mismo, vamos a crear un sencillo programa que nos indique qué posición corresponde a cada planeta.

Implementa un programa que:

- Te pida por consola que escribas un número entre 1 y 8 (inclusive).

- Al escribir el número y dar a enter, el programa mostrará una frase.

Por ejemplo, si introducimos el número 3 mostrará la siguiente frase:

La posición 3 corresponde al planeta Tierra.

### ENTRADA

*Ejemplo 1:*

*1*

*Ejemplo 2:*

*3*

### SALIDA

*Ejemplo 1:*

*La posición 1 corresponde al planeta Mercurio.*

*Ejemplo 2:*

*La posición 3 corresponde al planeta Tierra.*

### SOLUCION

Escrita en C++:

```
#include <iostream>
#include <vector>

using namespace std;

void planetPosition(int num);
string planetName(int num);

int main()
{
    cout << "Introduce un numero entre el 1 y el 8 (inclusive): ";

    int num;
    cin >> num;

    planetPosition(num);

    return 0;
}

void planetPosition(int num)
{
    if(num > 0 && num < 9)
    {
        cout << "La posicion " << num << " corresponde al planeta "<<
planetName(num) <<"." << endl;
    }
}

string planetName(int num)
{
    string vectorPlanet[] = {"Mercurio", "Venus", "Tierra", "Marte",
"Jupiter", "Saturno", "Urano", "Neptuno"};
    string planet = "";
    int vectorSize = sizeof(vectorPlanet);

    for (int i = 0; i < vectorSize; ++i)
    {
        if(i == (num-1))
        {
            planet = vectorPlanet[i];
            break;
        }
    }
    return planet;
}
```