```java
package edu.upvehu.gbg.ejemplos;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.URL;
import java.util.ArrayList;
import java.util.List;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class MummecPV {
    private static final String URL="http://gtts.ehu.es/German/Docencia/MUMMEC/ejercicios/20161215.txt";

    //Ejecutamos la prueba de ambos métodos
    //Los dos arrojan IOException que engloba también la MalformedURLException, de modo que se tratan
    public static void main(String[] args){
        try {
            Long t=System.currentTimeMillis();
            metodo1();
            System.out.println("Tiempo de ejecución: "+(System.currentTimeMillis()-t)+" ms.");
        } catch (IOException ex) {ex.printStackTrace();} //Mostramos el stack porque es un programa de consumo propio.

        try {
            Long t=System.currentTimeMillis();
            metodo2();
            System.out.println("Tiempo de ejecución: "+(System.currentTimeMillis()-t)+" ms.");
        } catch (IOException ex) {ex.printStackTrace();} //Mostramos el stack porque es un programa de consumo propio.
    }

    static void metodo1() throws IOException{
        List<Double> lista=new ArrayList<>();

        BufferedReader br = new BufferedReader(new InputStreamReader(new URL(URL).openStream()));
        String linea;
        while ((linea=br.readLine())!=null) {
            //Las siguientes dos lineas son típicas, pero no las necesitamos si descartamos todos los errores
            //linea=linea.trim();
            //if (linea.length()==0 || linea.startsWith("#")) continue;
            try {linea=linea.split("\\|")[2];
                String[] campos=linea.split(" ");
                lista.add(Double.parseDouble(campos[1])-Double.parseDouble(campos[0]));
            } catch(Exception ignore){/*si se quiere comprobar lo que se deshecha --> System.out.println(linea);*/}
        }
        System.out.printf("Número de muestras: %d\nMedia: %.2f ms.\nDesviación Tipica: %.2f ms.\n",lista.size(),media(lista)*1000,desviacionTipica(lista)*1000);
    }

    static void metodo2() throws IOException{
        //Llevamos el fichero remoto a un buffer (los pasos de linea se pierden, no los necesitamos)
        StringBuffer sb=new StringBuffer();
        BufferedReader br = new BufferedReader(new InputStreamReader(new URL(URL).openStream()));
        String linea; while ((linea=br.readLine())!=null) sb.append(linea);
        br.close();

        //Establecemos el "matcher" con el patrón que buscamos y llevamos los valores que nos interesan a una lista
        Matcher matcher=Pattern.compile("(\\d+\\.\\d+) (\\d+\\.\\d+)").matcher(sb);
        List<Double> lista=new ArrayList<>();
        while (matcher.find()) lista.add(Double.parseDouble(matcher.group(2))-Double.parseDouble(matcher.group(1)));

        System.out.printf("Número de muestras: %d\nMedia: %.2f ms.\nDesviación Tipica: %.2f ms.\n",lista.size(),media(lista)*1000,desviacionTipica(lista)*1000);
     }


    static double media(List<Double> l){
        double sum=0;
        for (Double d:l) sum+=d;
        return sum/l.size();
    }

    static double desviacionTipica(List<Double> l){
        return Math.sqrt(varianza(l));
    }

    static double varianza(List<Double> l){
        double media=media(l), sum=0;
        for (Double d:l) sum+=(d-media)*(d-media);
        return sum/l.size();
    }

}
```