

Algoritmos y Programación III

(75.07)

Cátedra Fontela

Primer Cuatrimestre de 2008

Trabajo Práctico

Datos Personales			
Apellido y Nombre:		CIAN, Nicolas	
Padrón		88056	
Email		nicolas.cian@gmail.com	
Turno	003	Grupo	
Fecha Aprobación			
Nota de Concepto			

```

/*
 * Nicolas Cian 88056
 *
 * TPO - E2
 * 26 de Marzo del 2008
 * Trabajo Practico Cero, Ejercicio Dos.
 *
 * 75.07 Algoritmos y Programacion 3 - Carlos Fontela
 * Universidad de Buenos Aires (UBA)
 * Facultad de Ingenieria (FIUBA)
 */

/**
 * Clase Documento, maneja datos de tipo texto,
 * se pueden crear documentos con un maximo
 * predeterminado de palabras.
 *
 *
 * @version 1.0 24 Mar 2008
 * @author Nicolas Cian
 */
public class Documento{
    /* Esta clase esta implementada con una estructura de Array estaticos */

    /** texto, es el nombre del vector de palabras del documento */
    private String[] texto;

    /**
     * cantidadPalabras, lleva la cuenta de cuantas palabras contiene
     * el vector texto.
     */
    private int cantidadPalabras;

    /**
     * Constructor sin parametros
     * Suponemos un maximo de 100 palabras
     */
    public Documento(){

        /* maximo: constante de maxima cantidad de palabras */
        final int maximo = 100;

        cantidadPalabras = 0;
        this.texto = new String[maximo];
    }

    /**
     * Constructor con argumentos
     * determinamos la cantidad maxima de palabras
     */
    public Documento(int maximo){
        cantidadPalabras = 0;
        this.texto = new String[maximo];
    }

    /**
     * Metodo agregar, se agrega al documento una cadena
     * de caracteres especifica.
     * utiliza la clase java.lang.String
     * utiliza los metodos split, replaceAll, length
     */
    @param cadena caracteres para agregar al archivo
    que forman un texto String
    */
    public void agregar(String cadena){

        //tomar la cadena y separar las palabras buscando los espacios
        //cada palabra va en una posicion del array Texto

        //me aseguro que no hay espacios repetidos, almenos 3
        cadena = cadena.replaceAll(" ", ":");
        cadena = cadena.replaceAll(" ", ":");
        cadena = cadena.replaceAll(" ", ":");

        //pongo cada palabra en un lugar del vector

```

```

String[] palabras = cadena.split(":");

//agrego estas palabras al atributo Texto
int tamañoPalabras = palabras.length;
for(int i=0; i<tamañoPalabras ;i++){
    texto[this.obtenerCantidadTotalPalabras() +i] = palabras[i];
}

//aumento la cantidad de palabras
this.cantidadPalabras = cantidadPalabras + tamañoPalabras;
}

/**
 * Metodo obtenerCantidadTotalPalabras, nos permite
 * conocer las palabras que contiene el documento
 * actualmente.
 *
 * @return cantidad de palabras del documento.
 */
public int obtenerCantidadTotalPalabras(){
    return this.cantidadPalabras;
}

/**
 * Metodo obtenerCantidadTotalPalabrasTerminadasEn, nos dice el
 * numero de palabras con una determinada terminacion.
 * utiliza la clase java.lang.String
 * utiliza los metodos toCharArray, charAt, length
 *
 * @param cadena texto para comparar la terminacion de
 * las palabras.
 * @return el numero de alabras termiandas en <code>cadena</code>
 */
public int obtenerCantidadPalabrasTerminadasEn(String cadena){

    int contador=0;

    for(int i=0; i<this.obtenerCantidadTotalPalabras() ;i++){
        // idea de LUCAS PANDOLFO
        char c = texto[i].charAt(texto[i].length() -1);
        if(c == (cadena.toCharArray())[0]){
            contador++;
        }
    }

    return contador;
}

/**
 * Metodo borrar, se encarga de eliminar las palabras que contiene
 * el documento al momento de la invocacion.
 */
public void borrar(){
    //setiamos la cantidad
    cantidadPalabras = 0;

    //ponemos vacios en todos lados
    for(int i=0; i<this.obtenerCantidadTotalPalabras(); i++){
        texto[i]="";
    }
}
}

```

Pasos a seguir para probar exitosamente el trabajo

en una consola, navegamos hasta la ubicación de los archivos .java (Documento.java y ProgramaDePrueba.java) para E2

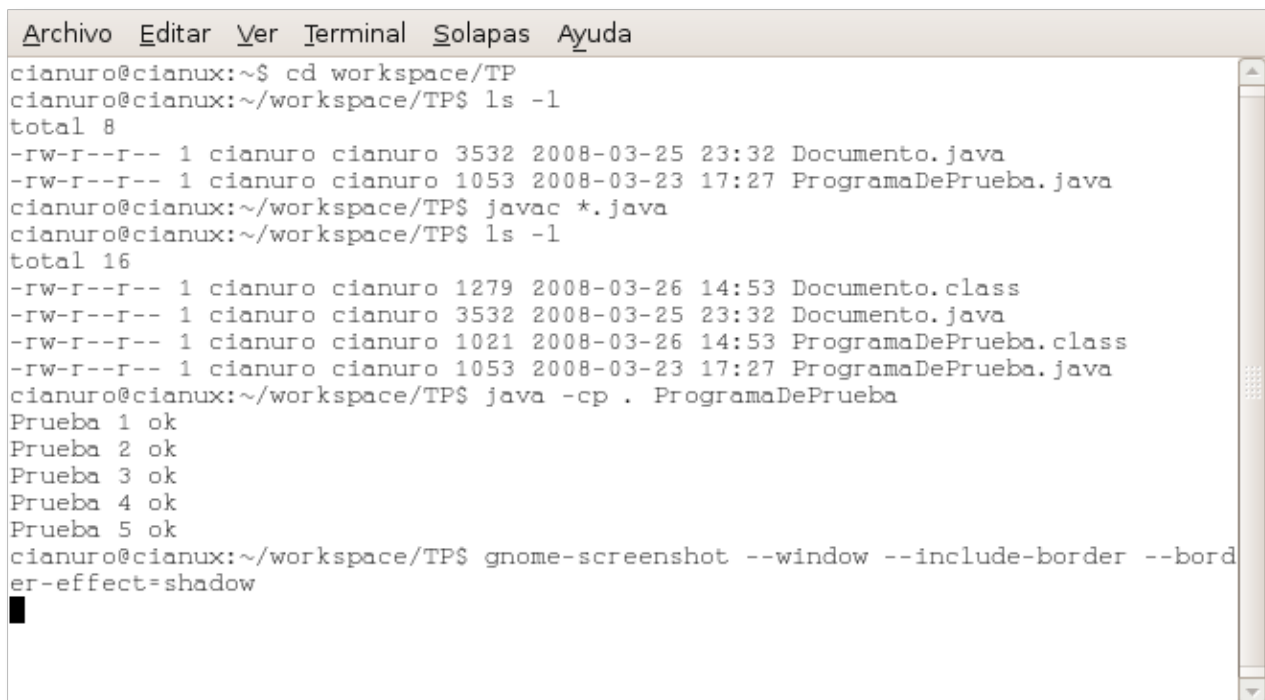
```
cianuro@cianux:~/workspace/TP$
```

ejecutamos el compilador

```
cianuro@cianux:~/workspace/TP$ javac *.java
```

ejecutamos la aplicación

```
cianuro@cianux:~/workspace/TP$ java -cp . ProgramaDePrueba
```



```
Archivo  Editar  Ver  Terminal  Solapas  Ayuda
cianuro@cianux:~$ cd workspace/TP
cianuro@cianux:~/workspace/TP$ ls -l
total 8
-rw-r--r-- 1 cianuro cianuro 3532 2008-03-25 23:32 Documento.java
-rw-r--r-- 1 cianuro cianuro 1053 2008-03-23 17:27 ProgramaDePrueba.java
cianuro@cianux:~/workspace/TP$ javac *.java
cianuro@cianux:~/workspace/TP$ ls -l
total 16
-rw-r--r-- 1 cianuro cianuro 1279 2008-03-26 14:53 Documento.class
-rw-r--r-- 1 cianuro cianuro 3532 2008-03-25 23:32 Documento.java
-rw-r--r-- 1 cianuro cianuro 1021 2008-03-26 14:53 ProgramaDePrueba.class
-rw-r--r-- 1 cianuro cianuro 1053 2008-03-23 17:27 ProgramaDePrueba.java
cianuro@cianux:~/workspace/TP$ java -cp . ProgramaDePrueba
Prueba 1 ok
Prueba 2 ok
Prueba 3 ok
Prueba 4 ok
Prueba 5 ok
cianuro@cianux:~/workspace/TP$ gnome-screenshot --window --include-border --border-effect=shadow
█
```

(ejemplificado en UBUNTU)