



PRACTICO N° 8

En todas las soluciones considere la importancia de diseñar algoritmos correctos, eficientes y legibles seleccionando estructuras de control adecuadas. En todas las implementaciones puede agregar métodos para favorecer la modularización.

EJERCICIO 1. Un profesor desea automatizar el mantenimiento de la planilla de notas de sus alumnos.

Durante el trimestre el profesor considera tres notas para cada alumno, la primera corresponde a una prueba escrita, la segunda a la exposición oral y la tercera a los trabajos prácticos entregados. Todas las notas están dentro del rango 0 a 10. A partir de la planilla el profesor desea saber:

- El promedio de cada alumno.
- La nota máxima por evaluación, escrita, oral y tp.
- El promedio general, que se obtiene de promediar el promedio de todos los alumnos.
- El promedio general por evaluación, escrita, oral y tp.
- El promedio de un alumno en particular.

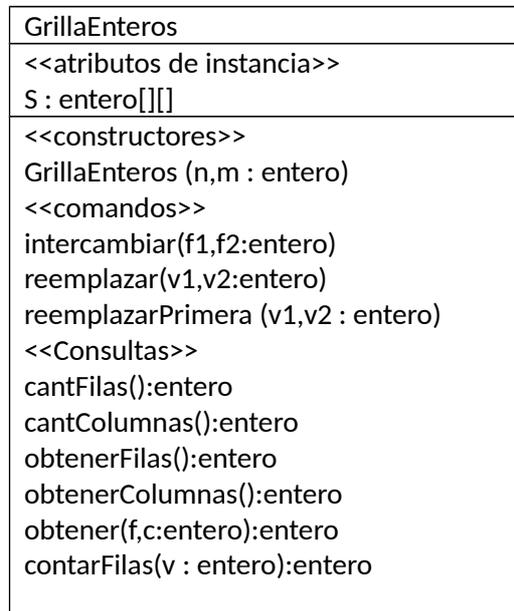
El profesor ha codificado la nómina de alumnos de modo que a cada uno de ellos le asignó un número comenzando con 1.

NotasAlumnos
<<atributos de instancia>> examen: entero [][]
<<constructor>> NotasAlumnos(cantAlu : entero)
<<comandos>> establecerNotaEscrita(p, n : entero) establecerNotaOral(p, n : entero) establecerNotaTP(p, n : entero)
<<consultas>> cantAlumnos(): entero obtenerNotaEscrita(p : entero) : entero obtenerNotaOral(p : entero) : entero obtenerNotaTP(p : entero) : entero promedioAlumno(p : entero) : real promedioEscritas() : real promedioOrales() : real promedioTP() : real promedioGeneral() : real maximaNotaEscrita() : entero maximaNotaOral() : entero maximaNotaTP() : entero

- Implemente una solución para este problema considerando el diagrama de clase y agregando todos los métodos que considere adecuados.
- Defina una clase tester para la clase NotasAlumnos, que lea las notas de un archivo secuencial y luego verifique los servicios provistos por la clase.
- Agregue a la clase NotasAlumnos una consulta **frecuencia()** que retorne objeto de la clase *NumerosEnteros* (del práctico 7) de 11 números enteros. Cada componente i del arreglo indica cuántos alumnos obtuvieron la nota i ($0 \leq i \leq 10$) en la exposición oral.



EJERCICIO 2. Implemente la clase GrillaEnteros modelada en el siguiente diagrama



El constructor **GrillaEnteros (n,m : entero)** crea un arreglo de n por m componentes y lo inicializa con números generados al azar.

intercambiar(f1,f2:entero) intercambia los elementos de las filas $f1$ y $f2$.

reemplazar(v1,v2:entero) reemplazar todas las apariciones del elemento $v1$ por el elemento $v2$.

reemplazarPrimera (v1,v2 : entero) reemplaza en cada fila la primera aparición del elemento $v1$ por el elemento $v2$.

contarFilas(v : entero):entero Cuenta en cuántas filas aparece el elemento v .

- Implemente la clase GrillaEnteros
- Establezca casos de prueba adecuados para verificar cada servicio.
- Escriba una clase tester que permita verificar los servicios con los casos de prueba almacenados en un archivo secuencial. La clase debe incluir un método interno mostrarGrilla()

Ejercicio 3. Analice la legibilidad, eficiencia y correctitud de la siguiente implementación del método tieneUnaRaya. El método debe retornar verdadero si y solo si, la matriz m , cuyos valores han sido establecidos previamente, contiene una fila i , $0 \leq i < nf$, tal que $m_{ij} = m_{i,j-1}$ para $1 \leq j < nc$

```
class Matriz {
    //Atributos de instancia
    int [][] m;
    int nf,nc;

    //Constructor
    public Matriz (int f,int c){
        m = new int [f][c];
        nf = f ; nc = c;
    }

    //Comando
    public void establecer( int e,int f,int c){
        //Requiere  $0 \leq f < nf$  y  $0 \leq c < nc$ 
        m[f][c] = e;
    }
}
```

