# Introducción al uso de bibliotecas de álgebra para estudiantes de ingeniería

Adolfo Di Mare
Escuela de Ciencias de la Computación e Informática
Universidad de Costa Rica
**adolfo.dimare@ecci.ucr.ac.cr**

**RESUMEN**

Presentamos una método de enseñanza sencillo para introducir el uso de bibliotecas Java para la manipulación de matrices en el contexto del único curso de programación que reciben los estudiantes de ingeniería, quienes aprenden programación y construcción de algoritmos porque necesitan resolver problemas específicos usando computadores, dejando de lado el conocimiento detallado de las tecnologías de programación o de computación.

**Palabras clave**: álgebra lineal, matrices, uso de bibliotecas de programas, introducción de técnicas de programación, computación explícita, software.

**ABSTRACT**

We present a simple teaching method to introduce the use of Java libraries for matrix manipulation in the context of the only programming course for engineering students, who learn programming and algorithm construction because they need to solve specific problems using computers, leaving aside the detailed knowledge of computer technology or programming.

**Keywords**: linear algebra, matrices, usage of program libraries, introduction to programming techniques, explicit computation, software.

## 1. INTRODUCCIÓN

Es difícil encontrar un plan de estudios para formar ingenieros que no incluya un curso de programación de computadores, lo que muestra que sí hay consenso en que la programación es una componente fundamental para el uso de herramientas computacionales. Algunos autores disputan si el lenguaje adecuado es Java, C/C++ o una herramienta específica de solución de problemas como MATLAB (BA-1996); talvez sea posible obtener resultados igualmente útiles si los estudiantes aprenden a usar ambientes integrados de producción de documentos, como el "*Office*" de Microsoft o el "*OpenOffice*" de Sun, pero la mayor parte de los docentes quieren que sus alumnos de ingeniería puedan llegar un poco más allá y prefieren impartir un curso de programación de computadores suficientemente completo.

Debido al constante avance tecnológico contemporáneo, el futuro ingeniero necesita conocer más sobre una mayor cantidad de temas. Esto dificulta aumentar la cantidad de cursos de computación del plan de estudios de la carrera, aunque hay excepciones a esta regla.

Los profesores de computación generalmente prefieren profundizar en las técnicas de programación y con frecuencia se concentran en enseñar tópicos avanzados como programación por objetos o el uso de interfaces

gráficas y graficación (SB-2006). Si un ingeniero necesitará usar su computadora para realizar cálculos, en realidad lo que necesita es conocer sobre construcción de algoritmos: este hecho debe ser aceptado por el profesor de programación para evitar que el estudiante de ingeniería termine sabiendo mucho sobre lo que no necesita. La premisa de la que se parte en este trabajo es que el estudiante ya sabe cómo programar algoritmos y lo que necesita más bien es utilizar bibliotecas de programas para efectuar algunos cálculos avanzados, los que no se pueden realizar cómodamente en la calculadora programable que acompaña a todo ingeniero.

La decisión de cuál lenguaje utilizar difícil de tomar, porque cada lenguaje tiene cualidades que es fácil usar para opacar las ventajas y realzar las deficiencias de los otros. Una forma sencilla de resolver el problema es usar un lenguaje que sea suficientemente simple de manera que muchos profesores lo conozcan, lo que facilita conseguir profesores para el curso, pero que también tenga un entorno que facilite su enseñanza. Java tiene estas dos cualidades y por eso se usa en muchas universidades. Además, Java es un lenguaje muy completo que sirve para avanzar en la tecnología de programación y es usado como primer lenguaje en muchos programas de estudios de computación (King-1997). También es un hecho que algunas herramientas de aplicación usan lenguajes similares a Java para automatizar procesos, como ocurre en el "*Visual Basic for Applications*" que complementa el "*Office*" de Microsoft, o el "*OpenOffice Basic*" de Sun. Estos lenguajes aumentan la potencia de las bases de datos y las hojas de cálculo que los ingenieros usan.

El lenguaje C tiene restricciones sintácticas que dificultan su enseñanza, lo que impide que el estudiante pueda resolver problemas interesantes pronto. Herramientas como MATLAB son poderosas pero están alejadas de la formulación usual de algoritmos que se usa en muchas aplicaciones de ingeniería; es deseable tener una buena formación en programación para luego aprovechar la potencia de herramientas como MATLAB.

Existen muchas bibliotecas para Java, y bastantes son de uso libre y gratuito (hasta parece que hay más bibliotecas Java funcionalmente completas que bibliotecas C++). Aquí se explica cómo lograr que un estudiante use la biblioteca JAMA de manipulación de matrices, pues después de que el alumno aprende a incorporar una biblioteca en su programa puede utilizar otras más según las necesite. La elección de la biblioteca JAMA, producida por el "*National Institute of Standards and Technology*", se fundamenta en estas cualidades:

- Es bastante completa, pues incluye la mayor parte de la operaciones en matrices que se necesita para resolver problemas prácticos.
- Como es simple sirve para introducir los conceptos inmediatamente.
- Incluye el código fuente lo que facilita entender cómo funciona la biblioteca.
- Está construida por una organización de buen prestigio en USA y en el mundo
  ➔ [ **`http://math.nist.gov/javanumerics/jama/`** ]

Parte de la contribución de este artículo es precisamente mencionar JAMA, pues hay tantas bibliotecas de difícil uso que muchas veces no se sabe por dónde comenzar.

Aunque JAMA es una biblioteca de objetos, pues Java usa fuertemente el paradigma de orientación a objetos, en este trabajo se destacan más bien cómo lograr llegar a una solución práctica para contar con un recetario de cómo usar esas rutinas.

## 2. MÉTODO DE ENSEÑANZA

Si se supone que ya el estudiante de ingeniería conoce los conceptos básicos de construcción de algoritmos entonces ya ha aprendido estos temas:

- Secuenciación
- Asignación y expresiones
- Decisiones **`if()`**
- Ciclos **`for(;;)`** y **`while()`**
- Uso de vectores o matrices
- Subrutinas y parámetros

No hace falta que el estudiante domine conceptos avanzados de programación por objetos como clases, referencias o clonación de objetos, pero sí debe conocer el uso un ambiente de programación, como DrJava (DrJava-2002) u otro similar. Muchos profesores de computación eligen DrJava porque incluye un depurador simbólico y es muy portable, hasta el punto de que es posible ejecutarlo desde una llave maya porque no necesita instalación.

Para lograr que el estudiante interiorice la forma de aprovechar una biblioteca, es muy útil darle las instrucciones como un proyecto o un examen. Aquí usamos el enunciado de este examen:
➔ [ **`http://www.di-mare.com/adolfo/cursos/2009-2/pi-ea-a.htm`** ]

Utilice la biblioteca JAMA para construir un programa que pueda resolver sistemas de ecuaciones de la forma AX=B. Construya su matriz A[70x70] llenando la primera fila con los primeros 70 números de Fibonacci. Para construir la segunda, corra todos los número de la primera fila hacia la izquierda una posición, y continúe así generando cada una de las 70 filas. Este es un ejemplo de una matriz de renglones Fibonacci:

```
0 1 1 2 3 5 8  Matriz A[7x7]
1 1 2 3 5 8 0
1 2 3 5 8 0 1
2 3 5 8 0 1 1
3 5 8 0 1 1 2
5 8 0 1 1 2 3
8 0 1 1 2 3 5
```

Las instrucciones específicas para usar la biblioteca JAMA son las siguientes:

Todos los archivos que forman un proyecto DrJava deben estar en una carpeta.

- Haga una carpeta nueva en la que copiará todos los archivos de su proyecto.
- Primero debe obtener el paquete "**`.jar`**" completo de JAMA:
  ➔ [ **`http://math.nist.gov/javanumerics/jama/`** ]
- Coloque el código Java en el archivo FiboMatrix.java.
- Para crear el proyecto DrJava, coloque el código XML en el archivo FiboMatrix.drjava.
- Copie los archivos "**`.jar`**" que forman la biblioteca JAMA en su carpeta.
- Verifique que el nombre de los archivos "**`.jar`**" que están en su carpeta son los nombres que se mencionan en el proyecto DrJava. Por ejemplo, si en su carpeta está el archivo "**`Jama-1.0.2.jar`**" es necesario que en el proyecto DrJava sea ese el nombre que se menciona en la etiqueta "**`classpath`**" del proyecto:

```
<classpath>
      <file absolute="false" name="Jama-1.0.2.jar"/>
<classpath/>
```

- También es posible crear el proyecto desde DrJava en el menú Project➔New, para luego agregar los archivos "**`.jar`**" en la sección "Extra Classpath" de Project➔Properties.
- La lista de archivos que usted debe tener en su carpeta es la siguiente:
  - **`FiboMatrix.drjava`**
  - **`FiboMatrix.java`**
  - **`Jama-1.0.2.jar`**
- Abra el proyecto FiboMatrix.drjava con DrJava, compílelo y ejecútelo.

**Figura 1**

Estas instrucciones permiten agregar una biblioteca ".jar" y usarla en un proyecto del ambiente DrJava (en otros ambientes de trabajo el efecto se logra usando el mecanismo de administración de proyectos). Este instructivo es muy simple y es muy corto; muestra que efectivamente Java es una plataforma computacional para la resolución de problemas funcional y completa.

Si ya el alumno sabe usar proyectos, basta darle esta indicación: "Agregue el archivo "**`.jar`**" en el menú Edit➔Preferences➔ResourceLocations de DrJava".

## 3. EJEMPLO PEDAGÓGICO

Al administrar el examen anterior los alumnos cometieron varios errores que conviene revisar. Al programador novato le cuesta separar las diferencias que hay entre clases, archivos y módulos Java. Por eso, el primer error que cometen es no incluir la biblioteca "**`JAMA.rar`**" al compilar y ejecutar el programa. Pese a que el profesor repite con insistencia que la computadora no adivina, en la mente de los muchachos la computación se les parece a una consulta Internet en donde todo lo relevante aparece junto y organizado, independientemente de su localización. Es extraño, pero los seres humanos no estamos acostumbrados a seguir instrucciones exactas y, pese a que el enunciado es claro y explícito, la mayor parte de las personas necesitan que les falle la compilación para interiorizar el hecho de que usar una biblioteca solo es posible si se le indica al compilador explícitamente que lo haga.

```
// Este es el programa solución completo, en pequeño, dimensión 7x7

public class FiboMatrix {
    public static void main( String args[] ) {
        double mat_double[][] = { // valores iniciales para M[][]
            { 0., 1., 1., 2., 3., 5., 8. },
            { 1., 1., 2., 3., 5., 8., 0. }, // F(0)==0
            { 1., 2., 3., 5., 8., 0., 1. }, // F(1)==1
            { 2., 3., 5., 8., 0., 1., 1. },
            { 3., 5., 8., 0., 1., 1., 2. }, // F(n)==F(n-1)+F(n-2)
            { 5., 8., 0., 1., 1., 2., 3. },
            { 8., 0., 1., 1., 2., 3., 5. },
        };

        Matrix M = new Matrix(mat_double);
        Matrix B = Matrix.random(M.getColumnDimension(),1);
        Matrix X = M.solve(B);
        B = B.transpose(); // para que salga impreso su valor en un renglón
        X = X.transpose();

        System.out.print("Matrix original M[][]"); {
            java.io.PrintWriter pwOut = new java.io.PrintWriter(System.out);
            M.print( pwOut, 6,2 );
            pwOut.flush(); // sin "flush()" graba retardado
        }
        System.out.print("Vector incógnita B[]"); {
            final boolean autoFlush = true;
            B.print( new java.io.PrintWriter(System.out,autoFlush) , 6,2 );
        }
        System.out.print("Vector solución X[]"); {
           B.print(  6,2 );
       }
    }
}
```

**Figura 2**

El siguiente error común es tratar de copiar el código ya conocido para obtener la solución, pero sin hacerle ningún cambio. La tendencia del alumno es tratar de programar de la misma forma con que se responde a los mensajes de correo electrónico, en que la costumbre es incluir el mensaje original como parte del nuevo sin hacerle cambios. A los estudiantes les costó mucho encontrar este error porque el cálculo recursivo de los números de Fibonacci se hace muy lento cuando los valores crecen por encima de 50 (¡algunos creyeron que la computadora se les había descompuesto o que había sido afectada por un virus!).

El no utilizar la indentación adecuada al escribir el código también causó problemas, pero afortunadamente ya el ambiente DrJava permite ver adónde comienza y adónde termina un bloque de código. Definitivamente es importante que, como parte de la evaluación, el profesor siempre exija que el código esté espaciado e indentado correctamente (Oppen-1980); el uso de herramientas automatizadas para darle formato al código es muy saludable (Astyle-2009).

Luego los estudiantes tuvieron que enfrentar el problema de la representación de la información en la computadora. Pese a que en Java todo es un objeto, no es lo mismo el objeto matriz que es un objeto vector que contiene referencias a otros objetos vectores, que el objeto "Matriz" de la biblioteca JAMA cuya representación interna está oculta, pues sus campos son privados (DiMare-2007). Este error se puede mostrar con este extracto de código:

```
double M[70][70]; // matriz de reglones Fibonacci
double B[70];     // vector de resultados
double X[70];     // vector de incógnitas
// ...
X = M.solve(B);   // solución del sistema en X.
```

El error es muy simple de visualizar pero para el novato no es fácil aceptar que la matriz "`M[][]`" se debe usar para construir el objeto JAMA que sí tiene el método "`solve()`":

```
double mat_double[70][70]; // matriz de reglones Fibonacci
Matrix M = new Matrix(mat_double);
Matrix B = Matrix.random(M.getColumnDimension(),1);
Matrix X = M.solve(B);
```

Otro error es confundir parámetros y objetos, utilizando "`B.solve(M)`" en lugar de "`M.solve(B)`". Como el compilador no emite mensajes de error en estos casos, para muchos estudiantes se tornó incomprensible el comportamiento de la máquina. Ellos saben que "la computadora no comete errores, es el programador quien no la ha programado correctamente", pero es más fácil echarle la culpa a otro cuando las cosas no funcionan.

Los alumnos luego quisieron mejorar su solución incorporando otras herramientas computacionales. El primer paso que dieron fue grabar los valores calculados en el formato CSV ("*Comma Separated Values*") (RFC-4180), que sirve para exportar datos para usarlos en una hoja de cálculo. Otros usaron la biblioteca "Java CSV" (JavaCSV-2009) pues entendieron que para incorporar esa biblioteca en sus programas basta seguir las mismas instrucciones del enunciado del examen (también comentaron, con orgullo, que sí vale más la pena hacer las cosas con Java en lugar de trabajarlas en la calculadora científica).

Lo interesante de esta experiencia es que muestra que si una persona aprende la parte básica de la programación, luego puede utilizar adecuadamente bibliotecas para obtener soluciones a problemas específicos aún si no tiene conocimientos profundos de construcción de sistemas pues el conocer un poquito de programación le permite a los estudiantes encontrar sus soluciones a sus problemas. Así como más vale enseñar a pescar que regalar pescado, también conviene más saber programar que solo saber usar un herramienta especializada para resolver un problema especializado. Por eso vale la pena aprender Java.

## 4. Precisión de la biblioteca JAMA

Es interesante determinar si la biblioteca JAMA tiene una precisión numérica adecuada para resolver problemas

medianos o grandes. Una forma de medir esta precisión es calcular la inversa de una secuencia de matrices cuyos renglones contienen los números de Fibonacci rotados de la manera definida en la sección anterior. Usando números enteros Java de tipo **`(long)`**, el valor "FiboMax"de la serie Fibonacci más grande que se puede calcular es el siguiente:

➔ **`Fibonacci(1,774) = 5,,181,326,,260,204,,576,479`**

Después de este número la precisión del resultado ya no cabe en una variable **`(long)`**, lo que se nota porque el valor calculado es un entero negativo.

La matriz de renglones Fibonacci de tamaño 1,774 x 1,774 tiene valores en un rango muy amplio [**`0..10`**$^{18}$], lo que permite ver si los errores por redondeo afectan mucho la precisión de los cálculos realizados con la biblioteca JAMA. Una forma de estimar este error es calcular la matriz inversa, multiplicarla luego por la matriz de tamaño "**`n`**" original para restarle la matriz identidad: **`norm(MxM'-I)`**, para cada uno de los valores de "**`n`**" en el rango [**`2..1,774`**]. Al tomar la norma se puede observar qué tan distante de la matriz identidad está el resultado de multiplicar la matriz por su inversa. Aquí el cálculo se hizo utilizando 3 normas diferentes: la mayor suma de las columnas, la mayor suma de las filas y la norma euclídea.

```
public static void main( String args[] ) {
    System.out.println( "N, MaxColSum, MaxRowSum, SqrtSumSquare, Millisec" );
    System.out.println( "0,0,0,0,1" ); // graba en formato CSV
    System.out.println( "1,0,0,0,1" );
    long ahora, ml;
    for ( int n=2; n<=FiboMax; ++n ) {
        Matrix Diff; // Diff = M.times(INV).minus(ID);
        {   Matrix INV, M = llenaFibonacci( n );
            //  if ( n<12 ) { M.print(4,0); }
            {   // ID Este bloque le permite a Java eliminar ID pronto
                Matrix ID = Matrix.identity(n,n);
                ahora = System.currentTimeMillis();
                // INV es la matriz inversa de M[][]
                INV = M.solve(ID);
                ml = System.currentTimeMillis()-ahora+1;
            }   // aqui el Recolector de Basura ya puede recolectar ID
            Diff = M.times(INV);
        }
        for ( int i=0; i<n; ++i ) { // Calcula Diff = Diff.minus(ID);
           Diff.set(i,i, Diff.get(i,i)-1.0);
        }
        double n1 = Diff.norm1();   // maximum column sum.
        double nI = Diff.normInf(); // maximum row sum.
        double nF = Diff.normF();   // sqrt of sum of squares of all elements.
        System.out.println( n + "," +  n1 + "," + nI + "," + nF + "," + ml );
    }
}
```

**Figura 3**

En la Figura 3 está el programa usado para hacer los cálculos. Para utilizar menos memoria no se usa la matriz identidad "**`ID`**" para restar el valor de la matriz "**`Diff`**", y se incluyen corchetes como sugerencia para que el recolector de basura Java recupere pronto la memoria usada por "**`ID`**". El método estático "**`llenaFibonacci(N)`**" produce una matriz JAMA de dimensión **`NxN`**.

Al examinar los cálculos se nota varios resultados curiosos, pues si la matriz de renglones Fibonacci tiene dimensión superior a **`NaNmax==1,477`** el valor que la biblioteca JAMA retorna para la norma de la matriz es "**`NaN`**" ("*Not A Number*"), lo que indica que la precisión numérica se ha agotado y en consecuencia el cálculo de la norma es incorrecto. Sin embargo, hasta ese punto el error máximo reportado por las 3 normas es **`5.5 x 10`**$^{-15}$,

que es un valor fenomenalmente bajo. Al correr el programa fue necesario cambiar el tamaño de la memoria dinámica Java disponible, ajustando el valor "*Maximun Heap Size for Main JVM in MB*" a 1024 megabytes, pues el valor por defecto resulta en falta de memoria para dimensiones superiores a 1,000. El computador usado para hacer los cálculos tiene 4 gigabytes de memoria administrada por Windows. No hubo problema en invertir una matriz de 1,774 renglones Fibonacci, y el tiempo utilizado fue relativamente corto: menos de 10 minutos.

Si la condición de la matriz es mala, en general ninguna biblioteca de matrices puede encontrar fácilmente la matriz inversa. Por ejemplo, al usar matrices de Hilbert, definidas por la fórmula `A[i,j]=1/(i+j+1)` con `A[0,0]=1`, JAMA calcula con error de 10-3 la inversa correcta para `N<=10`, pero a partir de `N=50` JAMA reporta que la matriz es singular, lo que es un error (wiki-2010).

Hay aplicaciones para las que es necesario manejar matrices de dimensiones más grandes, pero si decimos que una matriz de tamaño 1,000x1,000 es una matriz de mediano tamaño, podemos concluir que Java es una herramienta que sirve para obtener soluciones adecuadas para estos problemas. También podemos concluir que JAMA sí es una solución viable para matrices de tamaño mediano. Para matrices más grandes, ya es necesario usar otras bibliotecas más potentes, las que pueden estar implementadas en otros lenguajes como Fortran o C/C++. Es conocido el hecho de que la precisión de Java es adecuada para la solución de problemas numéricos (Bailey-2005), pero también es claro que la lentitud de una aplicación Java puede ser de uno o dos o órdenes de magnitud si se le compara con bibliotecas optimizadas Fortran o C/C++ (MMG-1998).

Como es de esperar, el tiempo que toma hacer los cálculos es una función que crece cúbicamente con el dimensión de la matriz. Para dimensiones inferiores a 500 el tiempo se mide en segundos, y luego se requieren minutos de espera para valores mayores. Para matrices de dimensión 2,000x2,000 no hace falta esperar más de una hora si se usa JAMA.

En Java está definido el tamaño de las variable escalares: tanto el tipo **`(long)`** como el tipo **`(double)`** se representan en el formato IEEE usando 64 bits o el equivalente de 8 bytes (LF-1999). Aún si el computador tiene una arquitectura de 128 bits, el tamaño de los escalares seguirá siendo el mismo; por eso, la precisión de los cálculos Java aquí establecidos es independiente del computador en que se ejecute el programa. Esta precisión permite manejar matrices muy grandes, que podrían llegar a dimensiones de unos pocos miles de filas y columnas.

## 5. CONCLUSIONES

Es buena idea que los estudiantes de ingeniería aprendan a programar y también es una buena elección usar el lenguaje Java en el único curso de programación que ellos reciben, pues así adquieren tanto los fundamentos de programación para resolver problemas adecuadamente como las bases para utilizar después otras herramientas más especializadas.

Al estudiante de ingeniería le basta un solo curso de programación y, si necesita resolver problemas grandes que sí requieren de un conocimiento más profundo de computación, como ingeniero puede decidir adquirir por sí mismo el conocimiento adicional o puede recurrir a un profesional de la computación para que le ayude.

No es necesario que los estudiantes de ingeniería profundicen en técnicas de programación, pero sí es valioso que sus profesores les muestren el uso de bibliotecas cuya funcionalidad puedan aprovechar fácilmente. En especial puede ser útil utilizar estás 3 bibliotecas que tienen en común su sencillez y facilidad de uso:

Matrices Java
➔ [ **`http://math.nist.gov/javanumerics/jama/`** ]
Graficación Java
➔ [ **`http://jchart2d.sourceforge.net/`** ]

Trasiego de datos en formato CSV

➔ [ **http://sourceforge.net/projects/javacsv/** ]

## 6. CÓDIGO FUENTE

Todos los programas usados en este trabajo están disponibles aquí:

➔ **[ http://www.di-mare.com/adolfo/p/ingbib ]**

➔ **[ http://www.di-mare.com/adolfo/p/ingbib/ingbib.zip ]**

## BIBLIOGRAFÍA


(Astyle-2009) "*Artistic Style 1.23 A Free, Fast and Small Automatic Formatter for C, C++, C#, and Java Source Code*", 2009. http://astyle.sourceforge.net/

(BA-1996) Bjedov, G. & Andersen, P. K.: "*Should Freshman Engineering Students Be Taught a Programming Language?*", Proceedings of the 26th Annual Conference on Frontiers in Education (FIE '96), vol.1, pp:90-92, Nov-1996.

(Bailey-2005) Bailey, D.H.: "*High-precision floating-point arithmetic in scientific computation*", Computing in Science & Engineering, Vol.7, Num.3, pp 54-61, May-2005.

(DiMare-2007) Di Mare, Adolfo: "*¡No se le meta al Rep!*", Reporte Técnico ECCI-2007-01, Escuela de Ciencias de la Computación e Informática, Universidad de Costa Rica, 2007. http://www.di-mare.com/adolfo/p/Rep.htm

(DrJava-2002) Allen, Eric & Cartwright, Robert & Stoler, Brian: "*Dr Java: A lightweight pedagogic environment for Java*", Rice University, ACM SIGCSE'02, Covington, Kentucky, USA, Feb-2002. http://www.DrJava.org

(JavaCSV-2009) *Java CSV Library*, 2009. http://sourceforge.net/projects/javacsv/

(King-1997) King, K. N.: "*The Case for Java as a First Language*", Proceedings of the 35th Annual ACM Southeast Conference, pp. 124–131, Apr-1997.

(LF-1999) Lindholm, Tim & Yellin, Frank: "*3.3 Primitive Types and Values*" in "*The JavaTM Virtual Machine Specification, 2nd Ed*"; Sun Microsystems, Inc; 1999. http://java.sun.com/docs/books/jvms/second_edition/html/VMSpecTOC.doc.html

(MMG-1998) Moreira, J.E. & Midkiff, S.P. & Gupta, M.: "*A comparison of Java, C C++, and FORTRAN for numerical computing*", IEEE Antennas and Propagation Magazine, Vol. 40, No. 5, Oct-1998.

(Oppen-1980) Oppen, Dereck C.: "*Prettyprinting*", Transactions on Programming Languages and Systems (TOPLAS), Vol.2 No.4, Oct-1980.

(RFC-4180) Shafranovich, Y.: "*Common Format and MIME Type for Comma-Separated Values (CSV) Files*", Request for Comments: 4180, IETF The Internet Engineering Task Force, Oct-2005. http://tools.ietf.org/html/rfc4180

(SB-2006) Schulte, Carsten & Bennedsen, Jens: "*What do Teachers Teach in Introductory Programming?*" ACM ICER'06, Canterbury, United Kingdom, Sep-2006.

(wiki-2010) Wikipedia: "*Hilbert matrix*", http://en.wikipedia.org/wiki/Hilbert_matrix


### *Authorization and Disclaimer*

*Authors authorize LACCEI to publish the paper in the conference proceedings. Neither LACCEI nor the editors are responsible either for the content or for the implications of what is expressed in the paper.*