Industrial Data
ISSN: 1560-9146
iifi@unmsm.edu.pe
Universidad Nacional Mayor de San Marcos
Perú

Huari Evangelista, Felix; Novara, Pablo José
Intérprete para probar un programa escrito en pseudocódigo
Industrial Data, vol. 17, núm. 1, enero-junio, 2014, pp. 101-109
Universidad Nacional Mayor de San Marcos
Lima, Perú

Revista de la Facultad de Ingeniería Industrial
17(1): 101-109 (2014) UNMSM
ISSN: 1560-9146 (Impreso) / ISSN: 1810-9993 (Electrónico)

Intérprete para probar un programa escrito en pseudocódigo

FELIX HUARI EVANGELISTA*
PABLO JOSÉ NOVARA**

RECIBIDO: 10/05/14 ACEPTADO: 25/05/14

RESUMEN

El objetivo del presente artículo es dar a conocer a la comunidad estudiantil que se inicia en la solución de problemas mediante programación el uso del programa llamado PSeInt. Este es un programa que permite escribir algoritmos mediante carta N-S, pseudocódigo, diagrama de flujo [1], y posteriormente hacer la prueba respectiva. Lo saltante es que el estudiante no necesita conocer ningún lenguaje de programación para usar esta herramienta.

Palabras clave: algoritmo, diagrama de flujo, carta N-S, programa, PSeInt, pseudocódigo

INTERPRETER TO TEST A PROGRAM WRITTEN IN PSEUDOCODE

ABSTRACT

The purpose of this article is to acquaint the student community that begins in troubleshooting programmatically using the program called PSeInt. This is a program that allows to represent algorithms using pseudocode, chart N-S, flowchart, and then make the respective test. The obvious trend is that the student does not need to know any programming language to use this tool.

Keywords: algorithm, chart N-S, flowchart, program, PSeInt, pseudocode

1. INTRODUCCIÓN

Cuando un estudiante inicia por vez primera la tarea de escribir programas de computadora para resolver un determinado problema, generalmente tiene que plantear el *algoritmo*, y este puede realizar usando un diagrama de flujo, cartas de Nassi-Shneiderman (carta N-S) o pseudocódigo.

Una vez resuelto el problema, debe simular la corrida del algoritmo con datos, a esta prueba se denomina prueba de escritorio, si al validar los datos en el algoritmo se obtiene los resultados correctos, podemos concluir que el algoritmo esta correcto y está preparado para ser *codificado* en cualquier lenguaje de programación [2].

El presente artículo tiene como objetivo presentar las características y bondades del programa PSeInt

2. OBJETIVO

Es escribir el algoritmo en diagrama de flujo, cartas de Nassi-Shneiderman o pseudocódigo sin necesidad de codificar en un determinado lenguaje de programación. Para lo cual se usará un intérprete denominado PSeInt y con ella comprobar los resultados y si existe error hacer la corrección en el mismo programa.

3. QUÉ ES PSeInt

El PSeInt viene de PSEudoINTérprete, donde PSE hacer referencia a PSEudocódigo e INT de INTérprete. Por lo tanto el PSeInt es un programa que interpreta un algoritmo escrito en pseudocódigo, carta N-S o diagrama de flujo. Este programa es libre y gratuito, se distribuye bajo licencia GPL (General Public License) [4].

PSeInt ofrece un editor con sintaxis coloreado, autocompletado, ayuda rápida en pantalla, la posibilidad de visualizar el algoritmo como diagrama de flujo, carta N-S, y luego convertir a código C++, y lo más importante la posibilidad de que la computadora interprete y ejecute el algoritmo en caso de ser válido, o señale los errores en caso contrario.

* Ing. Industrial. Docente de la FII-UNMSM. E-mail:fhuarie@unmsm.edu.pe
** Ing. Informático. Docente Facultad de Ingenierías y Ciencias Hídricas de la Universidad Nacional del Litoral-Argentina. E-mail: zaskar_84@yahoo.com.ar

PSeInt está pensado para asistir al estudiante que se inicia en la elaboración de programas o algoritmos computacionales. El pseudocódigo se suele utilizar como primer contacto para introducir conceptos básicos como el uso de estructuras de control, expresiones, variables, etc, sin tener que lidiar con las particularidades de la sintaxis de un lenguaje de programación real.

4. DESCRIPCION DEL PROGRAMA PSeInt

PSeInt es un programa compuesto por muchos programas (módulos). Es decir, consta en realidad de varios ejecutables que se invocan y comunican entre ellos, de forma tal que para el usuario final se observa como un único programa. El PSeInt está compuesto de los siguientes módulos [3]:

Pseint: Es el principal componente, se encarga de analizar un algoritmo en pseudocódigo, e indicar los errores si los hay, o interpretarlo en caso contrario. El análisis del algoritmo produce como resultado parcial un pseudocódigo normalizado que se utiliza como entrada en otros módulos. Es una aplicación de consola que toma el algoritmo del usuario desde un archivo de texto.

WxPSeInt: Es el editor de pseudocódigo (como texto), es la interfaz visual del sistema, desarrollado con wxWidgets. Presenta el editor de texto con todas sus ayudas, y se encarga de lanzar y gestionar el comportamiento de los demás módulos cuando es necesario.

Psterm: Es la terminal donde se ejecuta Pseint, y posee además la habilidad de registrar las entradas que el usuario hace por teclado para reproducir toda la ejecución desde cero cuando el algoritmo cambia, o se quiere volver en el tiempo para alterar una entrada.

Psdraw2: Se encarga de generar, mostrar y editar el diagrama de flujo. Toma por entrada un pseudocódigo normalizado, calcula los tamaños y posiciones de las entidades del diagrama y los visualiza y edita interactivamente con una interfaz basada en OpenGL (Open Graphics Library) y GLUT (del inglés OpenGL Utility Toolkit).

Psexport: Se encarga de traducir a código C++ un pseudocódigo normalizado. Gran parte de las tareas de traducción son independientes del lenguaje final.

Updatem: Solo se encarga de ver si hay actualizaciones una vez al día. Está separado de wxPSeInt solo para evitar que la interfaz se bloque o muestre errores cuando hay problemas de red.

5. CARACTERÍSTICAS Y FUNCIONALIDADES DE PSeInt

- Presenta herramientas de edición para escribir algoritmos en pseudocódigo en español
- Autocompletado
- Ayudas emergentes
- Plantillas de comandos
- Coloreado de sintaxis
- Indentado inteligente
- Permite generar y editar el diagrama de flujo del algoritmo
- Permite la edición simultánea de múltiple algoritmos
- El lenguaje pseudocódigo utilizado es configurable
- Ofrece perfiles de configuración predefinidos para numerosas instituciones.
- Puede interpretar (ejecutar) los algoritmos escritos
- Permite ejecutar el algoritmo paso a paso controlando la velocidad e inspeccionando variables y expresiones
- Puede elaborar automáticamente una tabla de prueba de escritorio
- Determina y marca claramente errores de sintaxis (mientras escribe) y en tiempo de ejecución
- Permite convertir el algoritmo de pseudocodigo a código C++
- Ofrece un sistema de ayuda integrado acerca del pseudocódigo y el uso del programa
- Incluye un conjunto de ejemplos de diferentes niveles de dificultad
- Es multiplataforma (probado en Microsoft Windows, GNU/Linux y Mac OS X)

6. ENTORNO DE DESARROLLO DEL PROGRAMA PSeInt

El entorno del programa PSeInt es el que se presenta a continuación y que es a través de ella se accede a las diversas opciones para la elaboración del pseudocodigo y su posterior puesta en marcha o prueba.

El programa presenta diversas opciones como todos los programas hechos para entorno Windows. El

menú principal principalestá dado por las opciones de Archivo, Editar, configurar, Ejecutar y Ayuda (figura 1). Cada una de estas a su vez tienes otras opciones que se describe como se indica.

Figura 1. Menú principal del PSeInt.

Fuente: Tomado del programa PSeInt.

Menú Archivo.- Está compuesto por los siguientes submenús (figura 2).:

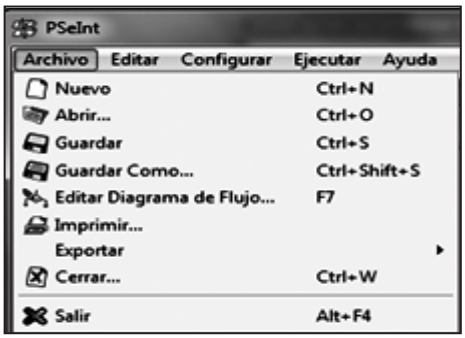
Nuevo.- Crea un nuevo archivo

Abrir.- Apertura un archivo existente

Guardar.- Permite guardar el archivo actual que se edita.

Guardar como.- Usado para guardar el archivo con otro nombre.

Figura 2. Menú Archivo.



Fuente: Tomado del programa PSeInt.

Editar diagrama de flujo.- Usado para editar el diagrama de flujo del pseudocodigo actual y actualizando automáticamente.

Imprimir.- Permite imprimir el archivo actual

Exportar.-Permite exportar el pseudocódigo a un programa en C, C++, pascal otros.

Cerrar.- Usado para cerrar el archivo actual

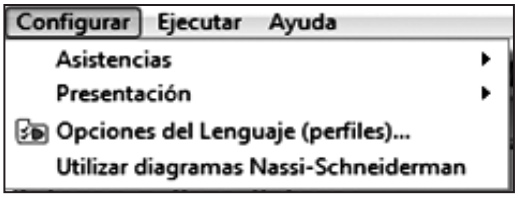
Salir.- Finaliza el programa.

Menú Editar.- Las opciones que se indican son algo similares que se presentan en la gran mayoría de editores de texto tales como copiar, cortar, pegar, buscar, reemplazar, etc.

Menú Configurar.-Presenta las siguientes opciones que se describan a continuación (figura 3):

Asistencia.- Permite activar la ayuda emergente, comprobación de sintaxis al escribir, autocompleato, otros.

Figura 3. Menú configuración.



Fuente: Tomado del programa PSeInt.

Opciones del lenguaje.- Configura el perfil que va usar en el pseudocodigo.

Utilizar diagramas Nassi-Schneiderman.- Convierte de pseudocódigo a diagrama N-S

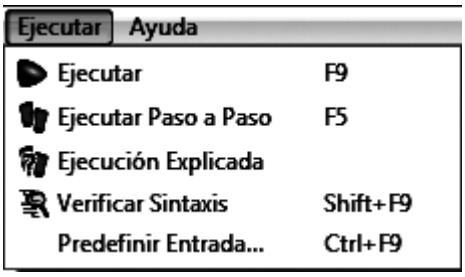
Menú Ejecutar.- Muestra las siguientes opciones (figura 4).

Ejecutar (F9).- Ejecuta el pseudocódigo mostrado actualmente.

Ejecutar paso a paso (F5).- Ejecuta paso a paso el pseudocódigo abierto actualmente.

Ejecución explicada.- Se ejecuta línea por línea y explicando lo que sucede.

Figura 4. Menú ejecutar.



Fuente: Tomado del programa PSeInt.

Verificar Sintaxis. Se encarga de verificar sintaxis del pseudocódigo.

Menu Ayuda. – Muestra la ayuda disponible para el programa.

Además de las opciones descritas arriba se puede encontrar las opciones mas importantes en la barra de herramientes como se indica a continuacion

Barra de herramientas

Es para accesar de forma rapida a las acciones que se desea realizar.

Figura 5. Barra de herramientas.

 Crea un nuevo archivo

 Abrir un archivo ya existente

 Guardar el archivo

 Guarda el archivo con otro nombre

 Deshace una acción

 Rehace una acción

 Cortar un texto o imagen

 Copiar un texto o imagen

 Pegar un texto o imagen

 Corregir indentado

 Buscar

 Buscar anterior

 Buscar siguiente

 Ejecutar

 Ejecutar paso a paso

 Dibujar diagrama de flujo

 Ayuda

Fuente: Tomado del programa PSeInt.

7. ELEMENTOS DE UN PROGRAMA EN PSEUDOCÓDIGO PARA PSeInt

Variables. Una variable es una posición de memoria donde se puede almacenar información.

Tipos de datos simples.- Existen tres tipos de datos básicos:

Numérico.-Indica todos los números, tanto enteros como reales.

Lógico. Solo puede tomar dos valores: VERDADERO o FALSO.

Carácter. Indica todos los caracteres o cadenas de caracteres encerrados entre comillas (pueden ser dobles o simples). Ejemplos “hola mundo”, ‘123’,etc

Operadores. Este pseudolenguaje dispone de un conjunto básico de operadores que pueden ser utilizados para la elaboración de expresiones más o menos complejas.

La siguiente tablamuestran la totalidad de los operadores de este lenguaje reducido:

Figura 6. Operadores utilizados.

Operador	Significado	Ejemplo
Aritméticos		
+	Suma	s←a+b
-	Resta	st←st-q
*	Multiplicación	p←a*b
/	División	z←a/b
% o MOD	Resto de división entera	r←a MOD b
Relacionales		
>	Mayor que	10>5
<	Menor que	5<4
=	Igual que	10=10

<>	Diferente que	A<>B
>=	Mayor o igual que	13>=10
<=	Menor o igual que	6<=32
Lógicos		
&ó Y	Conjunción (y)	(10>5)& (3<>9)
| ó O	Disyunción (o)	(4=4) | (8>10)
~ ó NO	Negación (no)	~(45<50)

Fuente: Tomado del programa PSeInt - ayuda.

8. ESTRUCTURA GENERAL DE UN ALGORITMO EN PSEUDOCÓDIGO PARA PSeInt

Todo algoritmo en pseudocódigo en PSeInt tiene la siguiente estructura (figura 7):

Figura 7. Estructura de un proceso.

ProcesoSinTitulo
acción 1;
.
acción n;
FinProceso

Fuente: Tomado del programa PSeInt - ayuda.

Comienza con la palabra clave **Proceso** seguida del nombre del programa, luego le sigue una secuencia de instrucciones y finaliza con la palabra **FinProceso**. Una secuencia de instrucciones es una lista de una o más instrucciones, las acciones incluyen operaciones de entrada y salida, asignaciones de variables, condicionales si-entonces o de selección múltiple y lazos mientras, repetir o para.

Sentencias secuenciales

Asignación. - Al ejecutarse la asignación, primero se evalúa la expresión de la derecha y luego se asigna el resultado a la variable de la izquierda (figura 8).Existen dos operadores de asignación alternativos que pueden utilizarse indistintamente en cualquier caso (:=, =), pero la habilitación del segundo (=) depende del perfil de lenguaje seleccionado.

Figura 8. Sentencia de asignación.

<variable><- <expresión>
<variable> := <expresión>
<variable> = <expresión>

Fuente: Tomado del programa PSeInt - ayuda.

Lectura.- Esta instrucción toma N valores desde el teclado y los asigna a las N variables mencionadas (figura 9). Pueden incluirse una o más variables, por lo tanto la instrucción leerá uno o más valores.

Figura 9. Sentencia de entrada de datos.

Leer<variable1>,<variable2>,.. <variableN>

Fuente: Tomado del programa PSeInt - ayuda.

Escritura.-Esta instrucción escribe en la pantalla los valores obtenidos de evaluar N expresiones(figura 10). Si en algún punto de la línea se encuentran las palabras clave **"SIN SALTAR"** o **"SIN BAJAR"** los valores se muestran en la pantalla, pero no se avanza a la línea.

Figura 10. Sentencia de salida de datos.

Escribir<exprl> ,<expr2> , ... , <exprN> ;

Fuente: Tomado del programa PseInt - ayuda.

Puede utilizarse indistintamente las palabras **Imprimir** y **Mostrar** en lugar de Escribir si su perfil de lenguaje permite sintaxis flexible. Además, en este caso se permite opcionalmente separar las

expresiones a mostrar simplemente con espacios en lugar de comas. Esto se configura en el cuadro de Opciones del Pseudocódigo.

Estructuras de control condicionantes

Condicional Si-Entonces.- Al ejecutarse esta instrucción, se evalúa la condición y se ejecutan las instrucciones que le siguen al Entonces si la condición es verdadera, o las instrucciones que le siguen al Sino si la condición es falsa (figura 11). La condición debe ser una expresión lógica, que al ser evaluada retorna Verdadero o Falso.

Figura 11. Sentencia de condicionante.

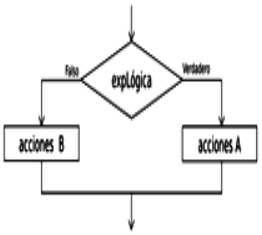
Si<condición>**Entonces**

<instrucciones>

Sino

<instrucciones>

FinSi



Fuente: Tomado del programa PseInt – ayuda.

Selección múltiple.- Esta instrucción permite ejecutar opcionalmente varias acciones posibles, dependiendo del valor almacenado en una variable de tipo numérico. Al ejecutarse, se evalúa el contenido de la variable y se ejecuta la secuencia de instrucciones asociada con dicho valor (figura 12).

Cada opción está formada por uno o más números separados por comas, dos puntos y una secuencia de instrucciones. Si una opción incluye varios números, la secuencia de instrucciones asociada se debe ejecutar cuando el valor de la variable es uno de esos números.

Opcionalmente, se puede agregar una opción final, denominada **De Otro Modo**, cuya secuencia de instrucciones asociada se ejecutará sólo si el valor almacenado en la variable no coincide con ninguna de las opciones anteriores.

Figura 12. Sentencia de selección múltiple.

Segun<variable>**Hacer**

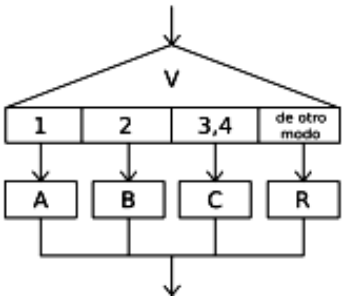
<número1>: <instrucciones>

<número2>,<número3>**:**<instrucciones>

<...>

De Otro Modo:<instrucciones>

FinSegun



Fuente: Tomado del programa PSeInt – ayuda.

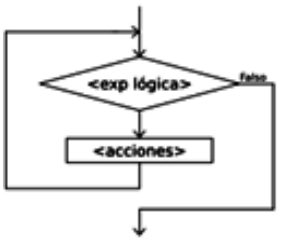
Lazos Mientras–Mientras la condición resulta verdadera, se ejecuta la secuencia de instrucciones que forman el cuerpo del ciclo, si la condición es falsa finaliza el ciclo repetitivo (figura 13).

Figura 13. Bucle mientras sea verdad.

Mientras<condición>**Hacer**

<instrucciones>

FinMientras



Fuente: Tomado del programa PSeInt – ayuda.

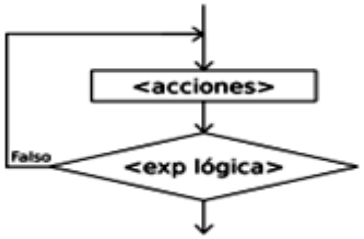
Lazos Repetir.- La instrucción *Repetir-Hasta Que* (figura 14) ejecuta una secuencia de instrucciones hasta que la condición sea verdadera, si la condición es falsa, sigue repitiéndose el ciclo.

Figura 14. Sentencia repetir hasta que.

Repetir

<instrucciones>

Hasta Que<condición>



Fuente: Tomado del programa PSeInt - ayuda

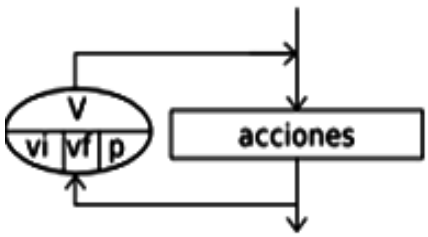
Lazos Para.- La instrucción **Para** ejecuta una secuencia de instrucciones un número determinado de veces (figura 15). Al ingresar al bloque, la variable <variable> recibe el valor <inicial> y se ejecuta la secuencia de instrucciones que forma el cuerpo del ciclo. Luego se incrementa la variable <variable> en <paso> unidades y se evalúa si el valor almacenado en <variable> superó al valor <final>. Si esto es falso se repite hasta que <variable> supere a <final>. Si se omite la cláusula Con Paso <paso>, la variable <variable> se incrementará en 1

Figura 15. Sentencia para.

Para<variable><-<inicial>**Hasta**<final> Con **Paso**<paso>**Hacer**

<instrucciones>

FinPara



Fuente: Tomado del programa PSeInt – ayuda.

Funciones/Subprocesos en PSeInt

Si el perfil de lenguaje seleccionado lo permite (ver Opciones del Pseudocódigo), se pueden declarar nuevas funciones o subprocesos (figura 16) en un algoritmo en Pseudocódigo. La sintaxis para ello es la siguiente:

Figura 16. Funciones o subprocesos.

SubProcesovariable_de_retorno<- nombre_de_la_funcion (argumento_1, argumento_2, .)

acción 1;

acción 1;

.

acción n;

FinSubproceso

Fuente: Tomado del programa PSeInt – ayuda.

Comienza con la palabra clave SubProceso (o Función, son equivalentes) seguida de la variable de retorno, el signo de asignación, el nombre del subproceso, y finalmente, la lista de argumentos entre paréntesis. Existen variantes para esta estructura. Si la función no retorna ningún valor, pueden omitirse el identificador variable_de_ retorno y el signo de asignación, es decir, colocar directamente el nombre y los argumentos a continuación de la palabra clave SubProceso. Si el subproceso no recibe ningún valor pueden colocarse los paréntesis vacíos u omitirse, finalizando la primer línea con el nombre del subproceso. Las reglas para los nombres de subprocesos, variables de retorno y argumentos son las mismas que para cualquier identificador en pseudocódigo.

Además, opcionalmente pueden agregarse las palabras claves Por Valor o Por Referencia para indicar el tipo de paso en cada argumento. Si no se indica, los arreglos se pasan por referencia, las demás expresiones por valor. El paso por referencia implica que si la función modifica el argumento, se modificará en realidad la variable que se utilizó en la llamada, mientras que el paso por valor implica que la función opera con una copia de la variable (o el resultado de la expresión) que se utilizó en la llamada, por lo que las modificaciones que aplique la función no se verán reflejadas fuera de la misma.

Observaciones Importantes para el programa PSeInt

Se pueden introducir comentarios luego de una instrucción, o en líneas separadas, mediante el uso de la doble barra (//). Todo lo que precede a //, hasta el fin de la línea, no será tomado en cuenta al interpretar el algoritmo.

Los identificadores, o nombres de variables, deben constar sólo de letras, números y/o guión_bajo (_), comenzando siempre con una letra.

Las constantes de tipo carácter se escriben entre comillas (").

En las constantes numéricas, el punto (.) es el separador decimal.

Las constantes lógicas son Verdadero y Falso.

Definir. Define una variable en un proceso estricto,ejemplo: definir x como real. Los tipos posibles son entero, real, logico, caracter/texto.

PSeInt acepta las palabras clave Y, O, NO, y MOD como sinónimos de los operadores &, |, ~ y % respectivamente si la opción esta activada.

Personalización del Lenguaje

Hay ciertas características del pseudocódigo que pueden personalizarse para adaptarse a las preferencias de cada docente. Por defecto, PSeInt utiliza la configuración más flexible y natural posible, pero ofrece una lista de configuraciones predefinidas denominadas "perfiles". Se puede acceder a la lista de perfiles desde el ítem "Opciones del Lenguaje..." del menú "Configurar".

Si ninguno de los perfiles resulta adecuado se puede utilizar el botón "Personalizar" ubicado debajo de la lista de perfiles para definir una por una las siguientes posibles personalizaciones:

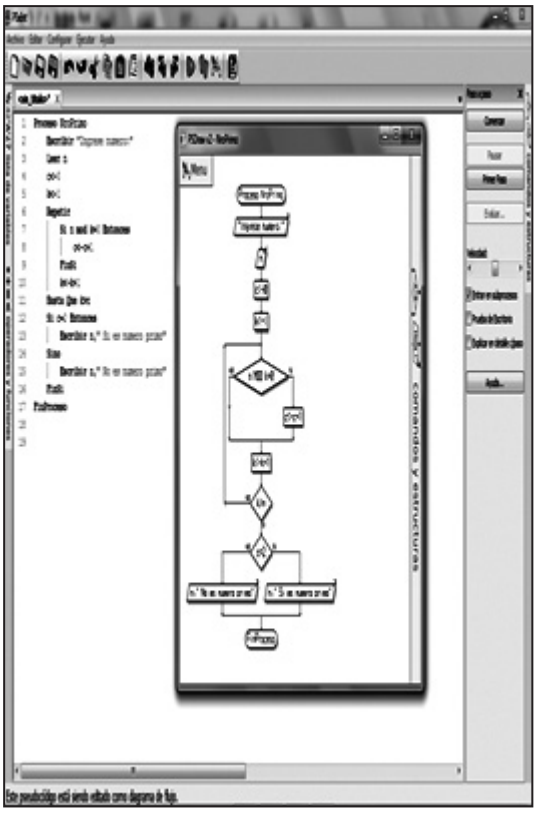
Usar diagramas de Nassi-Schneiderman: Si esta opción está deshabilitada el editor de diagramas flujo graficará este tipo de diagramas en lugar de los diagramas de flujo clásicos utilizados originalmente por PSeInt. Esta opción no modifica el lenguaje, por lo que también se puede activar/desactivar directamente desde el menú configurar mediante el ítem Usar diagramas de Nassi-Schneiderman, de forma que el usuario pueda alternar fácilmente entre uno y otro tipo de diagrama si lo desea.

9. APLICACIÓN EN PSeInt DE UN PSEUDOCÓDIGO

A continuación se muestra el pseudocodigo para verificar si un número entero positivo es un número primo. Se dice que un número entero es primo si es divisible entre la unidad y el mismo número.

La siguiente captura de pantalla (figura 17) muestra la edición del pseudocodigo, al seleccionar la opción muestra automáticamente el diagrama de flujo.

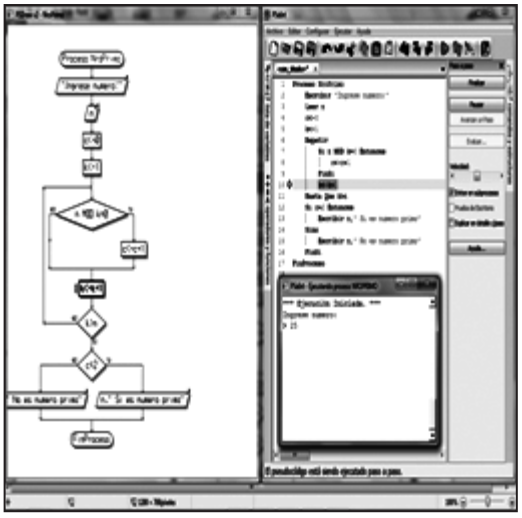
Figura 17. Pseudocódigo versus diagrama de flujo.



Fuente: Elaboración propia con el programa PSeInt.

En esta pantalla se muestra la prueba del algoritmo paso a paso (figura 18).

Figura 18. Prueba del algoritmo paso a paso



Fuente: Elaboración propia con el programa PSeInt

10. CONCLUSIONES

La mayoría de los estudiantes que se aventura a escribir programas de computadora, siempre tienen la dificultad de entender cómo trabaja un programa, pero a través del presente artículo esperamos brindar un aporte y que el estudiante podrá hacer lo siguiente:

1. El uso del programa PSeInt facilita su entendimiento al resolver un problema cualquiera usando pseudocodigo, carta N-S o diagrama de flujo.
2. Mostrar el problema en cualquier momento en las formas indicadas (pseudocodigo, carta N-S o diagrama de flujo).
3. Hacer la prueba con datos reales a través del programa en estudio.
4. No necesita utilizar un lenguaje de programación para probar el programa.
5. La rigidez o la flexibilidad de las instrucciones son seleccionables o pueden ser personalizados.
6. Las instrucciones son sencillas y fáciles de entender dado que están escrito en español.
7. El pseudocodigo, carta N-S o diagrama de flujo puede ser exportado a lenguajes como C++, pascal, C, etc.
8. Los docentes podrán utilizar como herramienta instructivo para hacer la demostración a los estudiantes.
9. Se recomienda hacer todas las pruebas necesarias con el programa y si por allí saltan errores, comunicar para informar a los desarrolladores para que puedan hacer la actualización respectiva.

11. AGRADECIMIENTO

La publicación de este artículo ha sido posible gracias a la colaboración del Ing. Pablo José Novara, docente de la Facultad de Ingenierías y Ciencias Hídricas de la Universidad Nacional del Litoral, república Argentina.

12. REFERENCIAS BIBLIOGRÁFICAS

[1] Joyanes, L., Zahonero I. (2005). Programación en C Metodología, algoritmos y estructura de datos, 2da. edición. Editora: Concepción Fernández Madrid – España.

[2] Ceballos, F. (2002). C/C++ Curso de programación Ra-Ma, 2da. edición. Alfaomega Mexico.

[3] Novara P. (2003-2014). PSeInt http://pseint.sourceforge.net/index.php?page=perfiles.php (visitado el 23/03/2014)

[4] Free software fundation (2007). General Public License http://es.wikipedia.org/wiki/GNU_General_Public_License (visitado el 23/03/2014)