La notación polaca como estrategia para aproximarse al pensamiento funcional y al pensamiento matemático*[1]

[Polish notation as strategy to approximate the Functional thinking and the mathematical thinking]

Omar Iván TREJOS BURITICÁ[2], Álvaro Ignacio MORALES GONZÁLEZ[3]

Recibo: 06.02.2013 – Aprobación: 25.11.2013

Resumen

El presente artículo expone una propuesta para aproximarse a la esencia del pensamiento funcional, base para la apropiación y aprovechamiento del paradigma funcional en programación de computadores, y la simplificación del pensamiento matemático. Dicho paradigma ha sido utilizado en la asignatura Programación I, del programa Ingeniería de Sistemas ofrecido por la Universidad Tecnológica de Pereira. Este es el primer curso de programación de computadores con que los estudiantes se encuentran, por lo que se ha querido proponer un camino para que ellos apropien, asimilen y comprendan la esencia del pensamiento funcional y, de paso con ello, simplifiquen el pensamiento lógico matemático.

Palabras clave*: Aprendizaje, Ingeniería de Sistemas, Paradigma Funcional, Pensamiento Matemático, Programación*

---

* Modelo para la citación de este artículo:
TREJOS BURITICÁ, Omar Iván & MORALES GONZÁLEZ, Álvaro Ignacio (2013). La notación polaca como estrategia para aproximarse al pensamiento funcional y al pensamiento matemático. En: Ventana Informática No. 29 (jul-dic). Manizales (Colombia): Facultad de Ciencias e Ingeniería, Universidad de Manizales. p. 145-160. ISSN: 0123-9678

1 Artículo de investigación científica y tecnológica proveniente del proyecto *Formulación de un modelo para el mejoramiento del aprendizaje basado en técnicas socio afectivas y estrategias BBL desde una óptica humanizante,* aprobado por la *Vicerrectoría de Investigaciones, Extensión e Innovación de la Universidad Tecnológica de Pereira*, ejecutado en el periodo *01.2010-12.2012*, en el *Grupo de Investigación en Informática.*

2 Especialista en Instrumentación Física, Maestría en Comunicación Educativa, Doctorado en Ciencias de la Educación. Docente de Planta, Universidad Tecnológica de Pereira (Pereira, Risaralda, Colombia). Correo electrónico: omartrejos@utp.edu.co

3 Ingeniero de Sistemas, Especialista en Instrumentación Física, Magister en Instrumentación Física. Docente Asociado I, Universidad Católica de Pereira (Pereira, Risaralda, Colombia). Correo electrónico: alvaro.morales@ucp.edu.co

Abstract

This paper presents a proposal to approach the essence of functional thinking, basis for use of the functional paradigm in computer programming, and to simplify the mathematical thinking that has been used in the Programming I Course in Systems Engineering Program from Universidad Tecnológica de Pereira. This is the first course of computer programming with which students are and what we have tried to propose away for ownership, assimilate and understand the essence of functional thinking and incidentally there by simplify the mathematical logical thinking.

Keywords: *Learning, Systems Engineering, Functional Paradigm, Mathematical Thinking, Programming*

Introducción

El presente artículo se inspira en la búsqueda permanente de nuevas estrategias metodológicas que posibiliten aproximar al estudiante a la lógica de programación en el primer curso que reciben en su proceso de formación como ingenieros de sistemas y computación. Lo que se explica en este artículo ha sido probado, revisado y evaluado en la asignatura Programación I del programa Ingeniería de Sistemas de la Universidad Tecnológica.

A lo largo de más de 20 años en el ejercicio, tanto de la profesión de Ingenieros de Sistemas como de la labor como docentes, los autores han encontrado una gran preocupación de los docentes del área de programación de computadores en buscar mecanismos, estrategias y actividades que posibiliten un entendimiento más expedito de la lógica de programación de manera que se puedan apropiar los conceptos fácilmente y que aquello que en programación de computadores se llama paradigma pueda ser asimilado de manera simple y sencilla en sus más elementales conceptos, de acuerdo con Bransford (2003, 52). En este artículo se acude al paradigma funcional de programación dado que este es el paradigma del cual se ocupa la asignatura mencionada.

Debido a la avanzada de la tecnología y a la manera como ésta ha penetrado cada uno de los escenarios de la vida, cuando los estudiantes de Ingeniería de Sistemas se aproximan a un lenguaje de programación, sin quererlo, terminan acercándose mucho más al enfoque instrumental que al enfoque conceptual, según Aamodt & Wang (2009, 66), y es por eso que muchas veces se encuentran unos programadores bastante hábiles en el manejo del código y, al tiempo, muy distantes de los conceptos que subyacen a las tecnologías que posibilitan la programación de computadores.

Por esta razón, y a manera de hipótesis, coincidiendo con Verlee (1986, 72), podría preguntarse si ¿es posible adoptar una estrategia que permita simplificar, al tiempo, el pensamiento funcional y el pensamiento matemático alrededor del concepto de la programación de computadores?. La respuesta gira alrededor de la propuesta que se hace en este artículo y que, en palabras sencillas, consiste en presentarle a los estudiantes la notación polaca como mecanismos para lograr los dos objetivos que se formulan en la hipótesis.

En el caso del paradigma funcional, todos los esfuerzos didácticos se orientan a que los estudiantes de programación asimilen la filosofía de saber primero qué es lo que se quiere hacer, segundo con qué lo va a hacer y tercero cómo lo va a hacer. Dado que se vive en un mundo de tanta inmediatez producto de la influencia de la tecnología en la vida cotidiana de la sociedad, considera Bateson (1972, 19), el estudiante de Ingeniería de estos tiempos, teniendo en cuenta que es un nativo digital, de acuerdo con Small & Vorgan (2009), comienza en orden contrario, es decir, primero empieza a hacer lo que creía que quería hacer, luego piensa en lo que necesitaba para hacerlo y finalmente se pregunta qué es lo que quería hacer.

El paradigma de programación funcional establece ese orden para todas sus operaciones y, si bien no es fácil entenderlo dado que desde la formación en la básica primaria se enseña el otro orden, la propuesta de este artículo es una estrategia metodológica para que dicha filosofía no solo se asimile sino que también se aplique tanto en contextos matemáticos como en cualquier otro contexto que la programación de computadores propicie. En lo fundamental este artículo se basa en la tesis de que si se simplifican los conceptos, se les concede significado según Cohen (2001, 45), se usa de apoyo el conjunto de conocimientos previos y se relaciona con nuevos conocimientos, dentro de un marco de actitud positiva del estudiante, como señala Enzensberger (2000, 111), se habrá encontrado un camino expedito para que el estudiante de Ingeniería de Sistemas pueda apropiarse fácilmente de la filosofía que subyace a un paradigma de programación que, en este caso, corresponde al paradigma de programación funcional en sus conceptos fundamentales.

1. Fundamento teórico

1.1 Notación Convencional

Desde la primer infancia se le enseña al estudiante que el método para escribir una expresión aritmética recurre a escribir el operador

entre, mínimamente, dos operandos. De esta manera, si se quisiera expresar la suma entre 5 y 2 entonces debería hacerse de la siguiente manera: 5 + 2, entendible, aparentemente, desde todo punto formativo. En esta expresión es claro que se necesita sumar el valor 5 con el valor 2, sin embargo, el uso permanente de este tipo de expresiones ha generado la costumbrede pensar primero en los datos que se va a operar y luego en la operación. De otra parte, si la expresión se torna más compleja, se debe tener un parámetro de jerarquía que permita resolverla apropiadamente. De esta manera una expresión como: 5 + 6 / 3 + 2, podría llegar a generar resultados diferentes y, por tanto, tener varias interpretaciones, si no se ponen cotas jerárquicas al respecto. Pensando silvestremente podría pensarse que la expresión 5 + 6 / 3 + 2 se puede interpretar así:

$$a)\ \frac{5+6}{3+2} \qquad b)\ 5+\frac{6}{3}+2 \qquad c)\ \frac{5+6}{3}+2 \qquad d)\ 5+\frac{6}{3+2}$$

El problema no se remite solo a la forma como se interpretaría esta expresión sino a los resultados que se obtienen pues, si se recurre a la aritmética entera (cuyas operaciones no generan decimales), los resultados obtenidos serían respectivamente:

$$a)\ 2 \qquad b)\ 9 \qquad c)\ 5 \qquad d)\ 6$$

Es aquí en donde podría dimensionarse la gran importancia de una interpretación correcta pues, como puede notarse, los valores son completamente diferentes y si se supone que corresponden a la inyección de mililitros de un determinado medicamento para sostener la vida de un enfermo, pareciera que no fueran muchas las esperanzas que la aritmética le diera a dicho enfermo. Sin embargo, es claro que solo una ha de ser la opción válida; es allí donde entra la jerarquía de operadores, condición *sine qua non* que debe plantearse para que una expresión tan simple como la del ejemplo pueda ser interpretada apropiadamente y su resultado totalmente confiable.

1.2 Jerarquía de operadores

Esta jerarquía hace referencia a la categorización de precedencia o importancia que tienen los operadores debido a que, aritméticamente, no todos tienen el mismo peso, de acuerdo con Frabetti (2002, 33). En su forma más simple, esta jerarquía establece que primero se evalúan las potencia, segundo las divisiones y multiplicaciones (indistintamente) y tercero se evalúan las sumas y restas (también indistintamente). De esta manera, la interpretación de la expresión 5 + 6 / 3 + 2 tendría una sola expresión pues si se recorre la expresión de izquierda a derecha

se notará que no contiene operaciones de potenciación y, por tanto, queda superado este nivel. En el segundo nivel, se vuelve a recorrer la expresión de izquierda a derecha encontrando una división, por tanto es esa la operación que se realiza primero. Entonces, la expresión se convierte en 5 + 2 + 2, debido a que la división 6 / 3 origina como resultado 2.

Luego se vuelve a recorrer la expresión de izquierda a derecha, actuando dentro del tercer nivel, encontrándose una suma y luego otra suma. Esto significa que se resuelve la primera suma y luego la segunda suma. De esta forma, la expresión se convierte en 7 + 2 cuando se resuelve la primera suma y se obtiene el resultado 9, cuando se resuelve la segunda suma.

Si bien se puede asegurar que el resultado es completamente entendible y razonable y que a partir de la jerarquía de operadores se puede obtener un único resultado, también es cierto que es un procedimiento aparentemente sencillo pero cuando se pone en escena en un salón de clases resulta no serlo tanto especialmente cuando la expresión se hace mucho más compleja.

Por ejemplo la expresión 5 * 4 / 2 * 3 + 2 / 3 * 2 * 6, tiene un nivel de complejidad mayor y si en la expresión inicial se obtuvieron cuatro resultados diferentes (sin la incorporación de la jerarquía de operadores), esta expresión originará tantos resultados que la situación, aritméticamente hablando, será mucho más confusa. Sin embargo, siguiendo el derrotero que marca la jerarquía de operadores, el resultado (paso a paso) sería el que se muestra en la Tabla 1.

Tabla 1. Resolución de una expresión más compleja

Nivel	Expresión	Operación
Inicial Expresión Original	5 * 4 / 2 * 3 + 2 / 3 * 2 * 6	
I Nivel	5 * 4 / 2 * 3 + 2 / 3 * 2 * 6	Recorriendo la expresión de izquierda a derecha, se buscan potencias y no se encuentran
II Nivel	20 / 2 * 3 + 2 / 3 * 2 * 6	Se buscan divisiones y multiplicaciones (indistintamente), y se resuelve la 1ª multiplicación (5*4)
II Nivel	10 * 3 + 2 / 3 * 2 * 6	Dentro del 2º nivel, se resuelve la división que se encuentra (20/2)
II Nivel	30 + 2 / 3 * 2 * 6	Todavía en el 2º nivel, se resuelve la multiplicación (10*3)
II Nivel	30 + 0 * 2 * 6	Se resuelve la siguiente división (2/3) en aritmética entera
II Nivel	30 + 0 * 6	Se resuelve la multiplicación que es la operación que sigue (0*2)
II Nivel	30 + 0	Se resuelve la última multiplicación (0*6)
III Nivel	30	Se buscan sumas y restas (indistintamente) y se resuelve la única suma que se encuentra (30+0)

Aparentemente es más sencillo resolver una expresión (sin paréntesis) haciendo uso de la jerarquía de operadores, pero curiosamente al realizar este tipo de ejercicios en un salón de clases de la asignatura Programación I en un programa de Ingeniería de Sistemas y Computación, y aún teniendo clara la jerarquía de operadores, la variabilidad y cantidad de resultados diferentes sorprende pues se esperaría que no se diera tal confusión.

1.3 Notación prefija (Notación polaca)

Lukasiewicz (1966,38) observó que, escribiendo los operadores antes o después de sus operandos, no es necesario seguir los lineamientos de la jerarquía de operadores, especialmente en operaciones que tuvieren paréntesis o hacer varios recorridos para evaluar la expresión. La notación prefija es una alternativa que posibilita la resolución de expresiones aritméticas sin paréntesis sin tener que especificar la jerarquía de operadores, es decir, sin ambigüedad (así esté implícita en el desarrollo y solución de la expresión).

La notación prefija o polaca se basa en la construcción de un árbol binario a partir de una expresión y, luego de hacerle un recorrido específico, permite obtener una expresión equivalente que, apoyado en el principio fundamental del paradigma funcional, finalmente obtiene un resultado único e inconfundible, según considera Van Roy & Haridi (2004, 97). Lo primero que se va a aclarar es lo que significa un árbol binario. Por árbol binario se entenderá un modelo compuesto por nodos y ramas en donde las ramas permiten que los nodos se interconecten manteniendo la idea de que cada nodo tenga un padre y solamente dos hijos, excepto el nodo raíz (inicio del árbol) y los nodos hojas (fin del árbol). Bajo este modelo, se puede establecer que la expresión 5 + 6 / 3 + 2 puede entenderse como la suma de dos elementos: el número 5 y el resto de la expresión. La forma de representarlo a partir del concepto de árbol binario se muestra en la Figura 1.

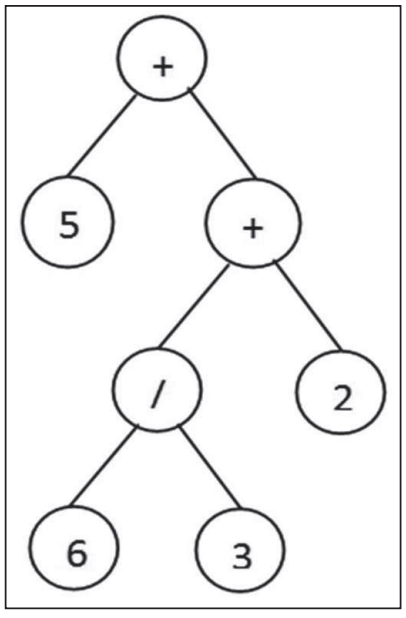

Figura 1. Expresión representada en árbol binario

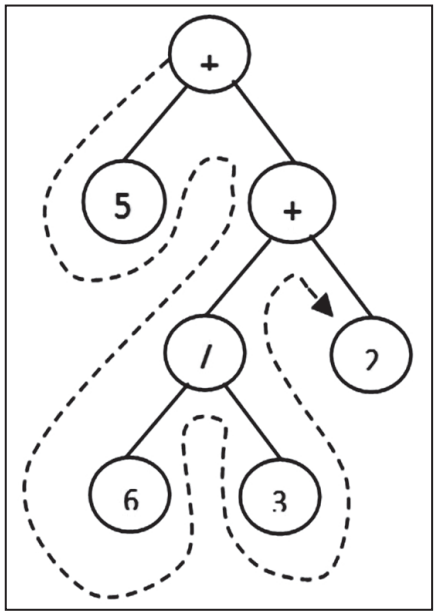

Figura 2. Recorrido por todos los nodos (Fuente: Original de los autores)

De la misma manera, se debe notar que la raíz del árbol (o sea el nodo que no tiene padre) es un operador, los nodos que tienen un padre y dos hijos son operadores y los nodos hoja (los que no tienen hijos) son operandos. Estableciendo un recorrido por todos los nodos acorde con la ruta que se muestra en la Figura 2 empezando en el operador + y terminando en el operando 2 final.

Si se escriben los operadores y los operandos que se van encontrando en este recorrido, se obtendrá la expresión + 5 + / 6 3 2 que comparada con la expresión original 5 + 6 / 3 + 2 pareciera ser un poco confusa pues no se ajusta a la forma como se enseñaron a escribir las expresiones aritméticas desde la primaria. No obstante, ya se ha visto en los párrafos anteriores, que resolver esta expresión por el método tradicional, aún siendo tan sencilla, pareciera incluir algunas ambigüedades bastante perjudiciales al momento de obtener resultados.

1.4 Metodología operador – dos operandos (Extender a n operandos)

Frente a la aparentemente confusa expresión + 5 + / 6 3 2 solo hace falta conocer su manera de ser resuelta. Para ello,se recurre a la esencia del pensamiento funcional que si bien todavía no se va a enunciar, de todas formas se va a utilizar puntualmente para luego generalizarse. Bajo este pensamiento, heredado del paradigma funcional, todo lo que se debe hacer con esta expresión es *resolver aquellos operadores que delante tengan dos operandos*. Esto lo hace muy sencillo pues, como se puede notar, no se habla de jerarquías ni de ninguna otra teoría, simplemente *operador – dos operandos* y listo. De esta forma, en la expresión + 5 + / 6 3 2 se encuentra que el primer operador + tiene por delante un número 5 y otro operador +, luego no se ajusta a la norma. El siguiente operador + (o sea el segundo +) tiene por delante un operador / y un número 6, luego tampoco se ajusta a la norma.

El siguiente operador / tiene por delante un número 6 y un número 3, encontrando la primera ocurrencia del formato *operados – dos operandos*. Por tanto se soluciona y la expresión original + 5 + / 6 3 2 se convierte en + 5 + 2 2, debido a que el resultado entero de dividir 6 / 3 es igual a 2. Al volver al inicio y comenzar de nuevo, se tiene una expresión mucho más simplificada.Se encuentra en la expresión + 5 + 2 2 que al primer operador + le siguen un número 5 y otro operador +, no se cumple la norma. Sin embargo, al segundo operador + le siguen dos operandos y por tanto se resuelve. De esta manera la expresión + 5 + 2 2 se convierte en + 5 4 dado que 2+2 es igual a 4.

Finalmente, se regresa al principio, con una expresión muchísimo más simplificada, encontrando un operador + seguido de dos operandos. Se resuelve y se obtieneel resultado final que es igual a 9 que corresponde al mismo valor que se encontró en la expresión escrita de manera convencional pero hallada por un camino más sencillo y simplificado; sin necesidad de recurrir a ninguna jerarquía de operadores y que es muy sencillo y entendible el principio de resolución de la expresión, a lo cual subyacen los fundamentos de pensamiento funcional heredado de la programación funcional. No hubo posibilidad a que se presentaran dos o más interpretaciones de la expresión y, por tanto, el resultado fue único e irrebatible.

En el caso de la expresión 5 * 4 / 2 * 3 + 2 / 3 * 2 * 6, por su naturaleza más compleja se ha de suponer que tanto el árbol como la expresión resultan un poco más extensos pero, para su resolución y luego de obtenida la expresión a partir del recorrido del árbol binario, se mantiene la norma *Operador – Dos operandos* y el problema queda simplificado. Es de anotar que la expresión que resulta del árbol binario se conoce como Expresión en Notación Prefija o Notación Polaca. De acuerdo a lo expuesto, el árbol resultante de esta expresión sería el que se muestra en la Figura 3 en la cual se nota una mayor cantidad de nodos, de ramas y de hojas acorde con el espíritu de la expresión que tiene un nivel mayor de complejidad dado su cantidad de operadores y operandos, así como también se muestra el sentido del recorrido.

La expresión resultante es+ * 5 * / 4 2 3 * / 2 3 * 2 6 y, aplicando la norma *resolver operador dos operandos* la Tabla 2 muestra su resolución paso a paso, puede notarse que el resultado obtenido es el mismo que se obtuvo al resolver la expresión por jerarquía de operadores. Cabe anotar que esta expresión, afirma Kaufmann et al. (2008), por no tener paréntesis, tiene un nivel de interpretación por jerarquía de operadores que llega a generar diferentes resultados y es, precisamente, eso lo que se simplifica pues en la resolución a partir de la expresión escrita en notación polaca no es necesario tener en cuenta la jerarquía de

operadores de manera consciente dado que, de manera automática, se va resolviendo con la jerarquía inconscientemente.

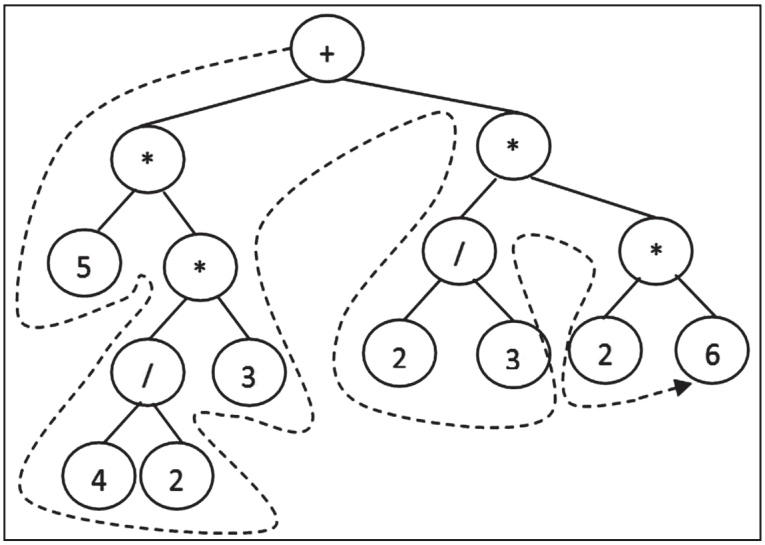

Figura 3. Árbol de una expresión más compleja

Tabla 2. Resolución de la expresión en notación polaca

Expresión	OperaciónRealizada	NuevaExpresión
+ * 5 * / 4 2 3 * / 2 3 * 2 6	/ 4 2	+ * 5 * 2 3 * / 2 3 * 2 6
+ * 5 * 2 3 * / 2 3 * 2 6	*2 3	+ * 5 6 * / 2 3 * 2 6
+ * 5 6 * / 2 3 * 2 6	/ 2 3	+ * 5 6 * 0 * 2 6
+ * 5 6 * 0 * 2 6	*2 6	+ * 5 6 * 0 12
+ * 5 6 * 0 12	*0 12	+ * 5 6 0
+ * 5 6 0	*5 6	+ 30 0
+ 30 0	+ 30 0	30

1.5 Expresiones lógicas

Tal como se hace con las expresiones aritméticas, también se procede con las expresiones lógicas y por tanto una expresión del tipo (a > b) AND (b < c), que estaría escrita en el sistema convencional, originará un árbol binario. El recorrido de la expresión lógica: (AND > a b <b c) se resuelve de la misma manera que se resolvieron las expresiones aritméticas escritas bajo la norma de la notación polaca, es decir, *operador – 2 operandos*. La secuencia de comparaciones sería la siguiente: primero se resuelve la expresión > a b, luego se resuelve la expresión < b c y por último se obtiene el resultado lógico de comparar con una operador AND los dos resultados anteriores.

1.6 Pensamiento funcional

El pensamiento funcional, base para la apropiación de conceptos del paradigma de programación funcional como vertiente de la programación declarativa, implica tres principios fundamentales que, en su orden, son: a) ¿Qué se va a hacer?, b) ¿Con qué se va a hacer? Y c) ¿Cómo lo va a hacer? (Felleisen et al., 2003). En este orden, y solo en este orden, el paradigma funcional provee los elementos de juicio que permiten que el pensamiento funcional tenga las bases necesarias para que se comprenda, se analice, se apropie y se aplique el conjunto de ventajas que tiene el paradigma funcional como base para la programación de computadores.

Si bien el paradigma de programación funcional se basa en funciones que son, en términos sencillos, pequeños núcleos de código que permiten alcanzar pequeños objetivos y que, asociadas con otras funciones, posibilitan el logro de objetivos más ambiciosos en términos de programación, estas funciones no son ajenas a la misma filosofía. A continuación se presenta un ejemplo simple escrito en lenguaje de programación *DrScheme* y que cumple, per se, con los postulados básicos del pensamiento funcional:

```
(define (area_del_triangulo Base Altura)
( / * Base Altura 2))
```

En esta función tan sencilla, pero que sirve de ejemplo para la explicación, se evidencian los tres postulados mencionados: ¿qué se va a hacer? Calcular el área de un triángulo (eso se supone a partir del nombre de la función area_del_triangulo); ¿con qué se va a hacer? Con dos argumentos: uno se llama Base y el otro se llama Altura. ¿Cómo se va a hacer?, aplicando la ecuación del área del triángulo que, en este ejemplo, está escrita en notación prefija. Ahora bien, si se quisiera construir una función que reciba los datos Base y Altura y los envíe a esta función para que haga el cálculo correspondiente entonces una solución podría ser:

```
(define (lectura_de_datos Base Altura)
(display "Digite la base")
(set! Base (read))
(display "Digite la altura")
(set! Altura (read))
(display "El área del triangulo es:")
(area_del_triangulo Base Altura))
```

De nuevo, si se analiza la función a partir de los postulados básicos del pensamiento funcional se obtienen las siguientes respuestas:

- ¿qué se quiere hacer? Leer unos datos o, dicho de mejor forma, realizar una lectura de datos (eso es lo que indica el nombre de la función *lectura_de_datos*);
- ¿con qué se quiere hacer? Con los argumentos que acompañan el nombre de la función y que, para este caso, se llaman Base y Altura;
- ¿Cómo se quiere hacer? Mostrando unos títulos alusivos a la lectura, almacenando lo que digite el usuario en los argumentos correspondientes y llamando, finalmente, a la función area_del_triangulo a la cual se le envían los valores de los dos argumentos leídos.

2. Metodología

La enfatización de los postulados básicos del pensamiento funcional, base para la apropiación del paradigma funcional en programación de computadores en la asignatura Programación I, se realizó en dos fases:

- En la primera fase se tomaron como base dos semestres de la asignatura en mención que correspondieron a los semestres I y II de 2010 y, durante esos cursos, se impartió el contenido completo sin hacer énfasis en los postulados fundamentales del pensamiento funcional. De esta forma se cumplió con lo establecido en lo normativo pero, en lo metodológico, se adoptaron las estrategias tradicionales de enseñanza sin que se hiciera el énfasis mencionado. Se tomaron notas a lo largo del semestre en relación con el rendimiento y apropiación del conocimiento de los estudiantes alrededor de los diferentes conceptos y temas que forman parte del contenido de la asignatura Programación I. Se hicieron observaciones al respecto de los resultados de las evaluaciones, los quices y las pruebas que se aplicaron como talleres y programas. Estas revisiones se realizaron tanto desde el punto de vista cuantitativo como desde la óptica cualitativa. El objetivo de esta estrategia radicaba en tener elementos de juicio que permitieran establecer relaciones comparativas entre la asignatura con énfasis en los postulados fundamentales a partir de estrategias como la notación polaca y la misma asignatura pero sin énfasis en los postulados fundamentales.
- En la segunda fase se tomaron como base los grupos de la asignatura durante los semestres I y II de 2011 de manera que, desarrollando los mismos contenidos pero haciendo un énfasis muy marcado en

la apropiación de los postulados básicos del pensamiento funcional, se pudieran hacer las comparaciones entre los dos estilos de servir la asignatura de manera que posibilitaran elementos de juicio acerca de la apropiación y asimilación de los conceptos básicos de la programación funcional y de su paradigma asociado. Al igual que en la primera fase, explicada en el ítem anterior, durante toda la experiencia se tomaron notas a lo largo del semestre en relación tanto con el rendimiento académico como con la apropiación del conocimiento especialmente en lo referente a los postulados básicos del pensamiento funcional. Se hicieron observaciones al respecto de las evaluaciones, los quices y todas las formas de seguimiento académico que se adoptaron, de manera que se facilitara la comparación con la otra experiencia. También se aplicaron criterios cuantitativos y cualitativos en el análisis de la información recogida.

A esta propuesta metodológica podría surgir una pregunta: ¿Porqué la experiencia no se hizo en paralelo? Una de las razones por las cuales no se hizo en paralelo obedece al hecho de que los cursos de Programación I involucran estudiantes que son muy cercanos en lo personal, sea que estén en el mismo curso o no y no se quería que un grupo *contaminara* al otro, siguiendo a Medina (2011), es decir, que la metodología con un grupo reflejara sus efectos en el otro y que, al final, los resultados de esta investigación no reflejaran lo que se quería comprobar, según Barrera et al. (2007).

Para no correr ese riesgo, se procedió a realizarlo en tiempos diferentes durante el 2010 (sin énfasis en los postulados fundamentales del pensamiento funcional) y en el 2011 (con un énfasis marcado en dichos postulados).Otra razón por la cual no se adoptó en paralelo la metodología propuesta obedece a que la valoración cualitativa partió de una interacción muy activa delos docentes de la asignatura (autores de este artículo) con cada uno de los estudiantes y se quería tener un tiempo suficientemente prudencial pero amplio para dedicarlo al proceso de comunicación con los alumnos y, de allí, poder tener los elementos de juicio suficientes que permitieran dicha valoración cualitativa. Es de anotar que durante la experiencia con los dos enfoques se procuró desarrollar actividades, evaluaciones, quices, pruebas, talleres y actividades muy similares, sin ser copia una de la otra, pero se quería mantener el rigor que exige especialmente la investigación cualitativa, de acuerdo con Small & Vorgan (2009, 89).

3. Resultados

3.1 Descripción

Se muestran en la tabla 3 los resultados cuantitativos que se obtuvieron de la adopción de la metodología expuesta en el numeral anterior. Se puede notar un sensible incremento en el promedio de las pruebas 1º parcial y 2º parcial entre los cursos con énfasis tradicional y los cursos en énfasis en postulados básicos del pensamiento funcional. Es de anotar que se mantuvo la tendencia en las tres pruebas en todos los cursos, es decir, el promedio de notas del 1º parcial fue más alta que el promedio de notas del 2º parcial y en el examen final, el promedio vuelve a aumentar comparándolo con la nota del 2º parcial.

Dicha situación pareciera obedecer al hecho que en el 1º parcial, la temática que se incluye corresponde a conceptos y teoría que constituyen la base para el desarrollo del curso, de la misma manera los talleres, *quices* y demás actividades evaluativas cubren un espectro de conocimiento pleno de conceptos sencillos. En el 2º parcial, la situación es completamente diferente dado que el conjunto de conocimientos que incluye esta segunda evaluación formal incluye el tema de la recursión, base para la construcción de procesos iterativos.

Tabla 3. Resultados cuantitativos

Énfasis	Semestre	1º Parcial	2º Parcial	ExamenFinal
Tradicional	I 2010	4.0	3.7	3.8
	II 2010	3.9	3.5	3.5
Postulados Básicos	I 2011	4.3	4.0	4.0
	II 2011	4.2	4.1	3.9

Este concepto de recursión mueve el piso lógico de la programación tradicional y su asimilación requiere un poco más de tiempo del esperado por parte de muchos estudiantes. El examen final, como era de esperarse, pareciera ser el promedio numérico de las notas de los parciales algo que, en términos de aprendizaje, tiene todo el sentido posible.No ha de olvidarse que las pruebas evaluativas parciales, los quices, los talleres y las demás formas de evaluación fueron realizadas en todos los cursos tratando de mantener características de similaridad para que los resultados pudieran ser comparables tanto en lo cuantitativo como en lo cualitativo. A pesar de que se revisaron varios ítems durante el curso, estos parecieran ser no solo los más relevantes sino también los que resumen la valoración cualitativa de los otros. En segunda instancia, vale la pena recordar que las valoraciones cualitativas no corresponden a una asociación con alguna escala cuantitativa; las valoraciones cualitativas surgen de la interacción comunicativa con los estudiantes

en sesiones que, siendo cortas, fueron muy frecuentes y posibilitaron que se dedicara el tiempo suficiente a cada alumno.

Se evidencia que los conceptos cualitativos que predominan en la valoración del rendimiento académico en los ítems mencionados en párrafos anteriores son notoriamente mejores en los dos semestres en los cuales se adoptó el énfasis sobre los postulados básicos del pensamiento funcional en comparación con los resultados del mismo proceso en el semestre en que este énfasis no se adoptó. Vuelve a encontrarse un escollo en relación con el concepto de recursión y su apropiación para el desarrollo de programas bajo el paradigma funcional.

En el desarrollo de la asignatura Programación I, y a lo largo del semestre, se detectaron algunos problemas; algunos asociados con los temas propios del contenido de la asignatura y, otros, asociados con los ajustes naturales que se han de hacer a metodologías innovadoras cuando éstas se incorporan en reemplazo de métodos tradicionales.

Entre las dificultades encontradas se pueden relacionar las siguientes: a) dificultades con la interpretación de la expresión original, es decir, la expresión aritmética escrita en forma lineal pero sin paréntesis, b) dificultades en la ubicación de los paréntesis, c) dificultades en la construcción del árbol binario que se deriva de la interpretación de la expresión, d) dificultades con la interpretación y ubicación apropiada de las operaciones resta y división.

Ahora bien, en relación con los logros obtenidos a partir de este enfoque metodológico se pueden citar los siguientes: a) asimilación inmediata del recorrido en prefijo de un árbol binario, b) asimilación inmediata de la obtención de la expresión resultante en notación prefija, c) asimilación moderada de la forma de resolver una expresión escrita en notación prefija, d) alta aproximación al pensamiento funcional, e) alta simplificación del pensamiento matemático respecto de la resolución de expresiones aritméticas.

3.2 Discusión

Por los resultados obtenidos y su análisis en relación con la metodología adoptada y con todos los elementos que se involucraron en esta experiencia, pareciera que, para los estudiantes la apropiación, asimilación y aplicación del pensamiento funcional (y del paradigma funcional), fuera mucho más sencillo cuando participan de un curso de programación con énfasis en los postulados básicos del pensamiento funcional que cuando no se adopta esta estrategia. Se nota, luego de varios ejercicios, talleres y actividades evaluativas, que para el estudiante es mucho más simple concebir, escribir y resolver una expresión

aritmética desde el pensamiento funcional que desde el pensamiento tradicional (Brendan, 2005).

Las opiniones de los estudiantes, tomadas durante el proceso de comunicación directa con ellos, constituyen una base para poder asegurarlo; es de anotar que si bien este no es un concepto totalmente generalizado, en un grupo determinado discrepan de esta posición uno o dos estudiantes. Los demás parecen encontrar éste como un camino más fácil para la resolución de expresiones y así lo corroboran en las actividades evaluativas.

4. Conclusiones

- Pareciera ser útil, tanto desde lo conceptual como metodológico e instrumental, adoptar un énfasis marcado en los postulados del pensamiento funcional en un curso de Programación I que se sirva del paradigma funcional con lo cual resulta de gran utilidad realizar valoraciones y análisis tanto cuantitativos como cualitativos para tener un panorama mucho más amplio de este tipo de experiencias, en concordancia con Ausubel (1986, 189), tal como lo demuestran los resultados de la Tabla 3.
- Se hace necesario que en los primeros semestres de un programa de formación universitario se fortalezca el concepto de Comunicación entre el docente y los estudiantes para conocer un poco más al alumno en su dimensión como ser humano acudiendo a la experiencia expuesta y a los fundamentos del Aprendizaje Significativo.
- Conviene que se tenga en cuenta el paradigma funcional como contenido de la primera asignatura de programación de un programa de ingeniería dada su alta cercanía con el pensamiento humanoasí como adoptar la notación polaca como mecanismo para la simplificación del pensamiento matemático en las asignaturas del área de Matemáticas. Los resultados al respecto del área de programación parecieran favorecer esta conclusión.

Referencias bibliográficas

AAMODT, S. & WANG, S. (2008). Entra en tu cerebro. Barcelona (España): Ediciones B. 335 p. ISBN: 978-846-663-738-1.

AUSUBEL, P. (1986). Sicología Educativa: Un punto de vista cognoscitivo. México (México): Trillas. 623 p. ISBN 978-968-241-334-6

BARRERA ZAPATA, A. F.; CEPEDA CIFUENTES, A del S.; DÍAZ SANTOS, D. C. & HURTADO MACHADO, D. M. (2007). El aprendizaje significativo como método para el desarrollo de la creatividad. En: Aprendizaje Significativo y Creatividad. V. 3. Bogotá (Colombia): Universidad de la Sabana. 25 p. <http://ictllp.wikispaces.com/file/view/aprendizaje+significativo+spanish.pdf> [consulta: 12/02/2013]

BATESON, G (1972). Pasos hacia una ecología de la mente. Buenos Aires (Argentina): Ediciones Lohlé-Lumen. ISBN: 950-724-700-9.

BRANSFORD, J. (2003). How people learn. Washington D.C. (USA): National Academic Press. 344 p. ISBN: 0-309-06536-4.

BRENDAN, B. (2005). A new approach to the computer science in the Liberal Arts [online]. In: Journal of Computing Sciences in Colleges, Vol. 20, No. 5 (may), Allentown (PA, USA): Consortium for Computing Sciences in Colleges, CCSC. EISSN: 1937-4763 *<http://dl.acm.org/signin.cfm?id=1059943&CFID=200391977&CFTOKEN=43819227> [consult: 02/04/2013]*

COHEN, H. (2001). Todo es negociable: Como conseguir lo que se quiere. Barcelona (España): Planeta. 224 p. ISBN: 950-9126-68-2.

ENZENSBERGER, H. M. (2000). El diablo de los números. Madrid (España): Siruela. 255 p. ISBN: 84-339-0380-2.

FELLEISEN, M.; FINDER, R. B.; FLATT, M. & KRISHNAMURTHI, S. (2003). How to design program: An introduction to programming and computing. 2 ed. Cambridge (MA, USA): The MIT Press. 728 p. ISBN: 0262062186

FRABETTI, C. (2002). Malditas Matemáticas: Alicia en el país de los números. Colección Próxima Parada 12 años. Madrid (España): Alfaguara. 136 p. ISBN: 84-204-6495-3.

KAUFMANN, E. K.; ROBINSON, J. S.; BELLAH, K. A.; AKERS, C.; HAASE-WITTLER, P. & MARTINDALE, L. (2008). Engaging students with Brain-Based Learning. In: Techniques (sep). Alexandria (VA, USA): Association for Career and Technical Education. p. 50-55. ISSN: 1091-0131.

LUKASIEWICZ, J. (1966). Elements of Mathematical Logic.2 ed. Oxford (Oxfordshire, UK): Pergamon Press.134 p. ISBN: 0080103936.

MEDINA, J. (2011). Los principios del cerebro en los niños: Cómo tener hijos listos y felices. Santafé de Bogotá (Colombia): Grupo Editorial Norma. 328 p. ISBN: 9789584533999

SMALL, G. & VORGAN, G. (2009). El cerebro digital. Barcelona (España): Urano. 256 p. ISBN 97884-7953-715-9.

VAN ROY, P. & HARIDI, S. (2004). Concepts, Techniques, and Models of Computer Programming. Cambridge (MA, USA): The MIT Press. 936 p. ISBN: 978-0262220699

VERLEE, W.L. (1986). Aprender con todo el cerebro. Barcelona (España):Martínez-Roca. 242 p. ISBN: 9788427010055