Pedagogía y Sociedad. Cuba. Año 19, no. 46, jul. - oct., 2016. ISSN: 1608 - 3784. RNPS: 1903

INDICACIONES CON CARÁCTER ALGORÍTMICO PARA LA ENSEÑANZA DE LOS CICLOS

INDICATIONS WITH ALGORITHM CHARACTER FOR LEARNING OF THE CICLES

Eduardo Hernández Martín[1]; Liosbel Fleites Cabrera[2]; Lisbel Valdés Leal[3]

[1]Licenciado en Cibernética Matemática. Máster en Didáctica de la Computación. Profesor Auxiliar de la Universidad de Sancti Spiritus "José Martí Pérez" Cuba. Email: martin@uniss.edu.cu [2]Licenciado en Educación. Especialidad Informática. Máster en Nuevas Tecnologías para la Educación. Profesor Asistente de la Universidad de Sancti Spiritus "José Martí Pérez" Cuba, Email lfleites@uniss.edu.cu [3]Licenciada en Educación. Especialidad Informática. Máster en Ciencias de la Educación, Mención Informática. Profesor Instructor de la Universidad de Sancti Spiritus "José Martí Pérez" Cuba, Email: lisbel@uniss.edu.cu

Resumen

Dentro de la programación estructurada, los ciclos son una temática esencial, que históricamente ha sido de difícil aprendizaje por los alumnos. Al aplicar el método de análisis y síntesis se concluye que en las diferentes fuentes consultadas, se abordan los ciclos dentro de un acápite denominado estructuras de control, donde esencialmente: se clasifican los ciclos, se explican cómo funcionan y se ejemplifica su uso. La anterior forma de explicar los contenidos no es suficiente para un buen aprendizaje, lo cual es un problema. Sobre la base de la experiencia pedagógica y como parte del proyecto de investigación "La formación didáctica del profesor para dirigir la solución de problemas desde las formas organizativas del proceso docente educativo", los autores del presente artículo científico de investigación cumplen el objetivo de proponer una sucesión de indicaciones con carácter algorítmico (SICA), obtenidas mediante el método de modelación, que al ser aplicadas en las clases impartidas al grupo de tercer año de la carrera Licenciatura en Educación, especialidad Educación Laboral e Informática, facilitaron el aprendizaje de los ciclos por los alumnos.

Palabras clave: Algoritmo; ciclos; estructuras de control; programación estructurada

Abstract

Within the structural programming, the cycles are essential topics, which have been difficult to study by students. When applying the analysis and synthesis method, it is concluded after consulting different sources that the cycles are tackled within an item called control structures, where essentially: the cycles are classified, are explained how they work and their use is exemplified. The previous way of explaining the contents is not enough to get a good learning, being a problem. Taking into account the pedagogical experience and as part of the research project "The didactic teacher training to run the solution of problems from the organizational forms of educational process", the authors of this scientific article fulfill the objective to propose a succession of indications with algorithm character (with its acronym in Spanish: SICA), obtained through the model method, that when they are applied in class to the group of third year of Degree in Education, Labor Education and Informatics specialty facilitate the learning of the cycles by students.

Key words: Algorithm; cycles; control structures; structured programming.

INTRODUCCIÓN

La disciplina Lenguajes y Técnicas de Programación (LTP) tuvo sus antecedentes en el plan de estudios A, que como consecuencia de un grupo de medidas del Estado Cubano para la introducción de la computación en la educación, se proyectó la asignatura Computación en la formación de los profesionales de la educación, como parte de las modificaciones que se hicieron a dicho plan.

En este momento, se concibió el estudio de la Computación en la educación superior, con el propósito de la resolución de problemas mediante computadoras haciendo uso de un lenguaje de programación. La asignatura tuvo en sus inicios un marcado énfasis teórico en el trabajo con algoritmos, que con la dotación de la tecnología de tableros inteligentes a los Institutos Superiores Pedagógicos posibilitó su codificación en el lenguaje MSX-BASIC.

Con posterioridad en el plan de estudio B, en la carrera Matemática y otras carreras se amplía el contenido informático a estudiar, incluyéndose contenidos relativos al Sistema Operativo y Sistemas de Aplicación además de los Lenguajes de Programación.

En la concepción del plan de estudio C, aparece la formación de profesionales de la educación para impartir la Informática en la educación general y politécnica, con la introducción de la carrera Matemática-Computación en curso diurno. En esta carrera dentro de las disciplinas del área de formación técnica, aparece la disciplina LTP para el estudio de la teoría de algoritmos y la programación estructurada.

En las modificaciones del plan C, en el curso 2001-2002 se establece la formación de profesores de Informática en curso por encuentros, que en el caso particular de la disciplina LTP se estructura el sistema de contenidos siguiendo la lógica de iniciar con el estudio de la algoritmia, después la programación estructurada, para culminar con la programación orientada a objetos y la programación conducida por eventos que confluían en la programación visual.

En el plan de estudios D, la carrera Educación Laboral-Informática asume la formación del profesional con doble perfil, que el caso de la Informática se prepara para impartir la asignatura en la educación general y politécnica. La disciplina LTP en esta carrera mantiene la lógica de contenido de la carrera de Informática en el Plan C, la que se propone seguir también para la carrera de Informática en curso diurno del Plan de Estudio E que se diseña actualmente.

En resumen las transformaciones, a través de la historia, realizadas a los programas de la disciplina han ido desde el estudio de lenguajes de bajo nivel, lenguajes intérpretes y compiladores hasta lenguajes de alto nivel, estudiándose en la actualidad los fundamentos de la programación modular y estructurada, orientada a objetos y finalmente los paradigmas modernos de programación visual en la que confluyen la programación orientada a objetos y guiada por eventos en una interfaz gráfica para el usuario.

Dentro de la programación estructurada la temática de los llamados ciclos, lazos, bucles iteraciones o repeticiones es primordial, pues esta estructura algorítmica básica incluye dentro de ella la lineal y la alternativa, pues al realizar ciclos hay que entrar datos y muchas veces implementar condiciones; además los ciclos son de necesaria implementación para la utilización de estructuras de datos complejas como son: vectores, matrices y ficheros de registros datos.

Disímiles autores se han referido al aprendizaje de los ciclos: (Calderón y García, 2004; Díaz y Pérez, 2001; García de Jalón, 2013; Hernández, Hondal, y González, 2005; Hernández y Martínez, 2002; Hernández, Machín y Fleites, 2015; Ochoa, 2014; Martínez, 2012; Peralta, 2014; Pozo, 2013; Sivira, Longone y Rodríguez, 2015; Trejos, 2013; Vildósola et al, 1990) todos ellos hacen referencia a la temática de estructuras de control y dentro ella realizan alusión a los ciclos, clasificándolos y explicando cómo funcionan, pero ninguno de los autores consultados hace referencia a unas indicaciones generales que puedan ser aplicadas en la docencia y faciliten el aprendizaje de esta estructura algorítmica básica, el objetivo del presente trabajo es proponer una sucesión de indicaciones con carácter algorítmico obtenidas mediante el método de modelación, que al ser aplicadas en clases faciliten el aprendizaje de los ciclos.

MARCO TEÓRICO O REFERENTES CONCEPTUALES

Los autores consultados incluyen la temática de los ciclos dentro de las estructuras de control del flujo del programa, planteando esencialmente que: el ciclo o lazo en Computación permite que una secuencia de pasos se repita tantas veces como sea necesario, bajo condiciones o conociendo el número de repeticiones, es decir, para que exista un ciclo debe haber la presencia de un algoritmo o instrucciones de un programa que se ejecutan más de una vez. Todos coinciden en que no es programable un ciclo que se repita indefinidamente, por lo tanto es necesaria una condición para terminar la ejecución del lazo en dependencia de las circunstancias establecidas por el enunciado del problema.

Al respecto Díaz y Pérez (2001) plantean que:

> Un bucle es una estructura de control que permite ejecutar una misma sentencia múltiples veces. El número de veces que se ejecuta un bucle puede ser conocido de antemano, en cuyo caso se suele usar un bucle por contador, o bien no serlo, utilizándose entonces un bucle condicional. (p. 59).

Estos autores también realizan una serie de ejemplos: iniciada con el bucle con contador donde se explica el uso del For y que continúa con los bucles condicionales donde se estudian las sentencias While y Repeat.

En sus consideraciones Calderón y García (2004) escriben que:

> Existen problemas de la vida cotidiana que implican repetir una o más acciones para lograr los resultados necesarios. Por ejemplo, cuando se quiere calcular un promedio de notas, sumar varios valores, contar y de manera general cuando se quiere procesar listas de datos. En estos casos utilizaremos los algoritmos con lazo o ciclo. (p. 55).

Después de esta explicación que encabeza el epígrafe *Algoritmos con Lazo o Ciclo*, los anteriores autores, realizan un estudio de los tipos de ciclo que denominan determinados que incluye 5 ejemplos y a continuación se comenta acerca de otro tipo de ciclos que nombran indeterminados incluyendo 4 ejemplos.

Otra explicación para los ciclos plantea que: "estos tipos de sentencias son el núcleo de cualquier lenguaje de programación, y están presentes en la mayor parte de ellos. Nos permiten realizar tareas repetitivas, y *se usan en la resolución de la mayor parte de los problemas* [cursivas añadidas]." (Pozo, 2013, p. 33). En este texto, a continuación se realizan explicaciones teóricas de los ciclos while for y do - while incluyendo un ejemplo de cada uno. Este autor coincide con el criterio de los autores del presente artículo de forma explícita, al plantear, en la propia definición del concepto, que los ciclos son primordiales en la enseñanza de la programación.

A su vez, García de Jalón (2013) se refiere a los ciclos planteando:

> Además de bifurcaciones, en el lenguaje C existen también varias sentencias que permiten repetir una serie de veces la ejecución de unas líneas de código. Esta repetición se realiza, bien un número determinado de veces, bien hasta que se cumpla una determinada condición de tipo lógico o aritmético. De modo genérico, a estas sentencias se les denomina bucles. (p. 35)

Después a continuación expone tres ejemplos: uno para el while, otro para el for y un último para el do - while.

Por su parte, Trejos (2013) aborda la importancia de la programación estructurada, haciendo hincapié en que esta técnica se basa en tres estructuras básicas: lineal, alternativa y repetitiva, con respecto a esta última plantea que:

> La estructura cíclica o iterativa permite que se pueda ejecutar un conjunto de varias instrucciones tantas veces como una condición lo permita, de manera que

su evaluación consienta, tal como sucede en la estructura de condición, que se realicen las tareas iterativas que se hayan propuesto. (p. 97)

Otros autores también han realizado explicaciones profundas referidas a la temática de los ciclos, por ejemplo:

Vildósola et al. (1990) dedica íntegramente un capítulo de su libro a las estructuras repetitivas y en él, parte de explicar las repeticiones incondicionadas incluyendo un ejemplo, y después explica las repeticiones condicionadas incluyendo un segundo ejemplo, como aspecto digno de destacar hay que señalar que incluye un epígrafe para los elementos acumuladores, aspecto que otros autores no realizan.

Ochoa (2014) dedica un epígrafe a las estructuras de control, donde primero comenta la estructura alternativa con sus dos posibilidades: la sentencia condicional y la sentencia de selección, después pasa a explicar la estructura de ciclos, a partir del ejemplo de la media aritmética, analizando la necesidad de ejecutar una repetición, explicando las tres posibles variantes de esta estructura.

Por su parte Sivira, Longone y Ramírez (2015) desarrollan un epígrafe dedicado a las estructuras de control, son los únicos autores que incluyen en este acápite las tres estructuras básicas, a razón: la lineal, la alternativa y la repetitiva. Esto no es común en los textos de programación, pues lo normal es considerar solo la alternativa y la repetitiva como estructuras de control y a la secuencial considerarla como la ejecución elemental, pues va desde la primera orden hasta la última. En el caso de la estructura repetitiva el estudio coincide con los cánones tradicionales, es decir: una definición de ciclo y ejemplos de las tres estructuras de ese tipo.

Fidalgo (2012) parte de conceptualizar los ciclos, para ello utiliza un ejemplo interesante que es el de una locomotora de ferrocarril que tiene que hacer varios viajes desde un punto A hasta uno B, a partir de este ejemplo analiza aspectos esenciales para los ciclos como son: cantidad de repeticiones, órdenes que se repiten, incluso ilustra la idea con un gráfico. Después pasa a explicar los tres tipos de ciclos, es digno de destacar que en su explicación presenta los algoritmos de los ejemplos, tanto en forma de pseudocódigo como en diagrama de flujo.

Como resultado del proceso de análisis de la consulta a la anterior bibliografía se puede sintetizar que todos los autores después de dar una idea primaria acerca de lo que son

los ciclos, en donde algunos de ellos incluyen una noción de la posibilidad de clasificarlos acorde a, si se conoce o no, la cantidad de repeticiones, plantean a continuación ejemplos resueltos donde tratan de agotar las distintas variantes más típicas, esto puede ser considerado suficiente para el aprendizaje de esta estructura, pero la praxis demuestra que no es así, que solo con tener una idea teórica de cómo funcionan los ciclos y ver algunos ejemplos no basta para poder resolver nuevos ejercicios por parte de los estudiantes.

El autor principal del presente artículo ha venido trabajando la temática de los ciclos de forma estable desde el año 2002, en colaboración con diferentes profesores que han sido miembros del colectivo de disciplina que él dirige, al respecto:

En el 2002, Hernández y Martínez hacen una propuesta para la enseñanza de los ciclos en el lenguaje Visual Basic, donde existe un acercamiento al concepto de ciclo controlado por el usuario, precisando que la persona que utiliza el programa garantiza la ejecución o salida del ciclo, presionando o no sobre botones de comando. Además se plantea un ejemplo, donde sin estar definida la SICA, se sigue su lógica de forma empírica.

En el 2005, Hernández, Hondal y González realizan un estudio de los ciclos, bajo la óptica de la programación visual, en el lenguaje Object Pascal, allí retoman el concepto de ciclo controlado por el usuario y además plantean una clasificación de los ciclos que incluye dos grandes tipos: ciclos condicionados y ciclos determinados.

En el año 2015, Hernández, Machín y Fleites introducen la SICA para la enseñanza de los ciclos; pero sin fundamentarla teóricamente, también exponen un ejemplo de uso, sin las explicaciones pertinentes, por lo que se puede considerar que la ponencia elaborada fue un primer paso para escribir el presente artículo.

Los ciclos. Su importancia dentro de la enseñanza de la programación estructurada

La técnica de programación estructurada fue pionera en el proceso de enseñanza de la solución de problemas mediante la elaboración de código para ser ejecutado en la computadora. En este proceso históricamente se estudian tres estructuras algorítmicas básicas que son: la lineal, la alternativa y la repetitiva.

La estructura lineal se aplica en la entrada de datos, cálculo de fórmulas y salida de resultados. En esta etapa se introducen los conceptos de constantes, variables, asignación de variables, nombres de variables, programa, cuerpo del programa, tipos de datos, declaración de variables, operadores matemáticos e instrucciones de entrada y salida.

Cuando se estudia la estructura alternativa se pasa a la temática de la toma de decisiones por la computadora. En esta etapa se introducen los conceptos de operadores relacionales y lógicos, expresiones booleanas o condiciones, tablas de verdad de los operadores lógicos, instrucciones alternativas simple y doble, anidamientos alternativos, instrucciones de selección y bloque de instrucciones.

Cuando las anteriores estructuras están estudiadas, se puede pasar al estudio de las repeticiones, siendo este contenido muy complejo y de difícil aprendizaje por parte de los alumnos. Las repeticiones se presentan en casi todos los lenguajes de programación bajo dos modelos: repetición determinada, cuando se conoce la cantidad de veces que se va ejecutar el proceso que se repite; repetición indeterminada , cuando no se conoce esta cantidad de repeticiones y entonces hay que controlar el proceso mediante condicionales estudiadas en la estructura alternativa. A su vez las repeticiones indeterminadas tienen dos formas: una donde se controla al inicio de la repetición y otra donde se controla al final de la repetición.

A partir de que se estudian las repeticiones entonces se complejiza la enseñanza de la programación pasando a estructuras de datos complejas como son: vectores unidimensionales, matrices bidimensionales y ficheros de registros datos, así como métodos de búsqueda y ordenamiento. En esta etapa de utilización de estructuras de datos avanzadas es muy importante el dominio de los tipos de ciclos, pues para el trabajo con vectores hay que crear datos indizados, en donde la variable que sirve como sub índice, es la misma que se utiliza para controlar el ciclo, sea cual fuere el tipo de repetición. Lo mismo ocurre con los ficheros de registros datos donde para la creación, recorrido e impresión de la información se hace necesario establecer ciclos para ir tomando los datos de uno en uno, hasta llegar al final del listado que se esté manejando.

Para reafirmar la necesaria utilización de los ciclos en la programación de estructuras de datos complejas, se puede analizar cuál es la forma que adoptan en la memoria de la computadora esas estructuras:

1. **Vectores unidimensionales**.

La representación interna de esta estructura de datos es una lista indizada, que empieza en uno y termina en n, por ejemplo el vector X, puede ser representado así: $X=X_1, X_2, X_3, X_4, X_5,....., X_n.$ Este tipo de organización de la información, implica para su manejo informático se utilice un ciclo, donde la variable de control va desde 1 hasta n, con un incremento de uno, entonces para acceder al elemento i del vector escribimos $X_{(i)}$

2. **Matrices bidimensionales**.

La representación interna de esta estructura de datos es una tabla con filas y columnas, indizadas desde 1 hasta n y desde 1 hasta m respectivamente, por ejemplo la matriz X, puede ser representada así:

$X_{1,1}$ $X_{1,2}$ $X_{1,3}$ $X_{1,4}$ ..... $X_{1,m}$

$X_{2,1}$ $X_{2,2}$ $X_{2,3}$ $X_{2,4}$ ..... $X_{2,m}$

$X_{3,1}$ $X_{3,2}$ $X_{3,3}$ $X_{3,4}$ ..... $X_{3,m}$

.....

$X_{n,1}$ $X_{n,2}$ $X_{n,3}$ $X_{n,4}$ ..... $X_{n,m}$

Este tipo de organización de la información, implica para su manejo informático de dos ciclos, donde: una variable de control va desde 1 hasta n, con un incremento de uno, para las filas y otra variable de control va desde 1 hasta m, con un incremento de uno, para las columnas, entonces para acceder al elemento i, j de la matriz escribimos $X_{(i,j)}$.

3. **Ficheros de registros de datos**.

Este tipo de organización de la información se comporta como un vector, pero con disímiles componentes que caracterizan a la entidad a programar, a diferencia del vector en vez de quedar almacenado en la memoria RAM, se utiliza para guardar información en un dispositivo de memoria externa, su representación es la siguiente:

Entidad 1: Característica 1, Característica 2, Característica 3,…. Característica Final.

Entidad 2: Característica 1, Característica 2, Característica 3,…. Característica Final.

Entidad 3: Característica 1, Característica 2, Característica 3,…. Característica Final.

……..

Entidad n: Característica 1, Característica 2, Característica 3,…. Característica Final.

En una primer análisis de la representación anterior, aparentemente se puede operar de forma similar a la matriz bidimensional; pero no es así, pues las características pueden ser de diferentes tipos, entonces lo que se hace es utilizar un ciclo desde 1 hasta n con un incremento de uno para operar cada entidad y se utiliza un operador, que en la mayoría de los lenguajes de programación es un punto. Entonces para acceder a las características de la entidad i escribimos Entidad $_{(i)}$.Característica 1, Entidad $_{(i)}$.Característica 2, y así respectivamente, de una forma similar al vector unidimensional.

Por todo lo anteriormente planteado se puede plantear que la comprensión y dominio de la estructura repetitiva es primordial en el desarrollo de las capacidades de los alumnos para resolver problemas mediante las técnicas de programación.

Los problemas, el algoritmo y la SICA en la programación estructurada.

Dentro de la enseñanza de la programación los algoritmos son parte esencial, mientras que la mayoría del resto de las disciplinas de la informática e incluso de otras ciencias los algoritmos son aprendidos para ser utilizados, en la programación, sobre todo en la estructurada, la esencia es elaborar un algoritmo que resuelva un problema.

Como categoría científica, el concepto problema adquiere diferentes acepciones, en correspondencia al área de conocimiento que se trate; se puede hablar de un problema en la lógica dialéctica, como un concepto filosófico superior; también se puede asumir desde el punto de vista psicológico o didáctico. Los estudiosos de la Matemática y su didáctica han efectuado caracterizaciones valiosas al respecto.

A partir de estudiar diferentes criterios, los autores del presente artículo, consideran un problema como un ejercicio que cumple las siguientes condiciones:

- Refleja determinadas situaciones a través de elementos y relaciones del dominio de la ciencia o la práctica haciendo uso de lenguaje común y requiere de medios para su solución.
- Exista una situación inicial y una exigencia que obliga a transformarla.
- La vía para pasar de la situación inicial a la nueva situación exigida tiene que ser desconocida y se obtiene con ayuda de procedimientos heurísticos.
- La persona debe necesitar hacer la transformación.

En el caso particular la Informática, la definición de problema adquiere connotación especial, pues estos se estudian estrechamente relacionados con los medios o recursos informáticos que se utilizan para su solución. Al respecto Expósito et al., (2001) plantearon que:

> Se considera que un problema informático es un ejercicio expresado mediante una formulación lingüística que contiene los elementos estructurales siguientes:
>
> - Datos o informaciones conocidas y necesarias.
> - Resultados o informaciones desconocidas.
>
> Y que tiene como propósito u objetivo esencial la búsqueda de un modelo o algoritmo para resolverlo. (p. 35)

En el año 2012, Hernández, Martínez y Fleites escribieron acerca de la utilización de problemas en la enseñanza de la programación con vistas a motivar a los alumnos por el aprendizaje de esta disciplina, al respecto plantearon que:

> Se considera que es esencial buscar vías, para que los alumnos que deben cursar la disciplina Lenguajes y Técnicas de Programación, "se enamoren" de la programación, esta relación afectiva, hay que iniciarla desde los contenidos de la técnica de programación imperativa y estructurada que es momento inicial. (p. 4).

Además en ese artículo fundamentan, cómo los problemas pueden despertar intereses intrínsecos en los estudiantes, sobre todo si se utiliza de forma eficiente la interdisciplinariedad, ejemplificando desde la programación estructurada.

¿Qué es un algoritmo?, la palabra algoritmo se deriva de la traducción al latín de la palabra árabe Alkhowarizmi, nombre de un matemático y astrónomo árabe que escribió un tratado sobre manipulación de números y ecuaciones en el siglo IX.

Existen varias definiciones del término, una de ellas plantea que: "Un algoritmo se puede definir como una secuencia finita de instrucciones cada una de las cuales tiene un significado claro y puede ser efectuada con una cantidad finita de esfuerzo en una longitud de tiempo también finito." (Peralta, 2014, p. 1). En la anterior definición se aprecia que la finitud es una característica esencial de un algoritmo y esto es un aspecto válido y acertado; pero no se habla acerca de resolver un problema y esto es una carencia.

Otro concepto de algoritmo, donde se aprecia la importancia de la solución del problema, sin perder la característica de la finitud, plantea que “Un algoritmo se define como un método que se realiza paso a paso para solucionar un problema que termina en un número finito de pasos”. (Martínez, 2012, p. 1).

Los autores del artículo se acogen al concepto, muy difundido en la enseñanza de la computación en las carreras pedagógicas, y que dice: “La lista, detallada, finita ordenada, de los pasos que necesitamos ejecutar para resolver un problema, se conoce en Computación, y en otras ciencias, como algoritmo de solución”. (Vildósola et al, 1990, p. 11); en esta definición se pueden apreciar varias características de los algoritmos y además en ella se hace referencia al problema a resolver.

¿Qué es la SICA?

La Sucesión de Indicaciones de Carácter Algorítmico, es un término planteado por especialistas dedicados a la Metodología de le Enseñanza de la Matemática, y que se caracteriza como “una sucesión de órdenes o indicaciones para realizar un cierto sistema de operaciones en un orden determinado, que inducen a operaciones unívocas, rigurosamente determinadas.” (Ballester et al, 2007, p. 246). En el caso de la enseñanza de la programación, particularmente de los ciclos, los autores del artículo proponen una SICA, que permite al alumno organizar su pensamiento y colocar en el orden adecuado las ideas que él obtenga, en aras de hallar el algoritmo que solucione un problema.

MATERIALES Y MÉTODOS

Desde los momentos iniciales del estudio de la técnica de programación estructurada, es decir cuando se emprende la enseñanza de las estructuras lineales y alternativas se recomienda utilizar los siguientes modelos:

Planteado por Expósito et al. (2001), se puede apreciar el siguiente modelo general, que es válido tanto para la enseñanza de los Sistemas de Aplicación como de las técnicas de programación.

Pasos del programa heurístico general de informática,

1. Determinar los elementos formales que integran el problema.
2. Determinar y describir los pasos principales de la solución.

3. Seleccionar las opciones necesarias y suficientes (para Sistemas de Aplicación), o codificar los pasos principales del algoritmo (para Lenguajes de Programación).
4. Controlar los resultados.
5. Introducir las acciones correctivas. (p. 62)

Para concretar los anteriores pasos generales, en la programación estructurada, los autores del artículo se acogen al siguiente modelo elaborado por Vildósola et al. (1990), que se ha hecho tradicional en Cuba:

Pasos para resolver problemas en la programación estructurada.

1. Análisis del problema
   [¿Qué dan como datos de entrada?
   ¿Qué debo obtener como resultado final?
   ¿Cuál sería una idea primaria de solución?
   ¿Qué datos auxiliares debo utilizar?]
2. Determinación del algoritmo de solución.
3. Escritura del programa en un lenguaje de programación.
4. Prueba del programa
5. Documentación del programa
6. Ejecución y uso del programa (p. 9)

Para el caso particular de los ciclos, los autores proponen utilizar dos elementos: una clasificación que discrimine teóricamente todas las posibles variantes y la SICA, ambos se presentan a continuación:

Clasificación que se propone utilizar para la enseñanza de los ciclos

1. Ciclo indeterminado: es un grupo de pasos del algoritmo, o de instrucciones del programa, que se pueden ejecutar más de una vez, de acuerdo a si se cumplen o no determinadas condiciones. Dentro de este tipo de ciclo se presentan dos situaciones:
   1.1. Evitable: Primero se realiza el chequeo de la condición, lo que impide que el último se procese, es decir, este dato no pertenece al conjunto que se desea procesar, su existencia solo se debe a la misión de detener la ejecución del ciclo. Además en este tipo de ciclo puede ser que no se ejecute el proceso

repetitivo ninguna vez. Se necesita que se entre un dato antes del ciclo llamado centinela, que deberá ser entrado o modificado dentro del ciclo para que pueda finalizar

1.2. Inevitable: Se realiza el proceso y se chequea después la condición, por lo que el último dato se procesa dentro del conjunto y obligatoriamente se ejecuta al menos una vez el proceso repetitivo. Se conoce a priori las características del último dato a procesar.

2. Ciclos determinados: se conoce la cantidad de datos que se van a procesar, pueden ser:
   2.1. Fijos: cuando el programador conoce la cantidad de repeticiones.
   2.2. Variable cuando el programador, por el contexto del problema, presupone que el usuario del programa conoce ese cantidad y lo va a pedir antes de que el programa ejecute el ciclo.

SICA, que deberá ser aplicada en la etapa de elaboración del algoritmo, en esta propuesta se organizan los aspectos a programar en tres etapas esenciales:

Antes del ciclo

a. Inicialización de variables (contadores, sumadores, mayores, menores)
b. Pedir datos previos
   i. Cantidad de repeticiones (caso de ciclo determinado variable)
   ii. Dato centinela (caso de ciclo indeterminado evitable)

2. Durante el ciclo
   a. Se entran los datos que se repiten
   b. Se hacen los procesos repetitivos
      i. cálculos de acumulación (conteos, sumatorias)
      ii. Se programan las condicionales parciales (if del mayor – menor, if de sublistas, if de búsquedas)
3. Después del ciclo
   a. Se hacen los cálculos generales (porcentaje, promedio, sumas, conteos)
   b. Se hacen las impresiones finales.

La anterior SICA puede ser dividida en tres versiones, una para tipo de ciclo, lo cual permite apoyar aún más a los estudiantes de bajo aprovechamiento.

Versión 1. SICA modificada para algoritmos determinados variables.

a. Antes del ciclo:
    i. Inicializaciones
    ii. Entrar n
b. Dentro del ciclo
    i. Entrar datos
    ii. Procesos repetitivos
        1. Hacer acumuladores
        2. Los if para mayor y menor
        3. Los if para algún criterio
c. Fuera del ciclo
    i. Cálculos generales (promedios, porcientos)
    ii. Impresiones generales

Versión 2. SICA modificada para algoritmos indeterminados evitables.

a. Antes del ciclo:
    i. Inicializaciones
    ii. Entrar dato centinela (escogido por el programador)
b. Dentro del ciclo
    i. Entrar otros datos (en caso de existir varios)
    ii. Procesos repetitivos
        a. Hacer acumuladores
        b. Los if para mayor y menor
        c. Los if para sacar algún criterio
        d. Volver a entrar dato centinela
c. Fuera del ciclo
    i. Cálculos generales (promedios, porcientos)
    ii. Impresiones generales

Versión 3. SICA modificada para algoritmos indeterminados inevitables.

a. Antes del ciclo:
    i. Inicializaciones
b. Dentro del ciclo

i. Entrar datos
ii. Procesos repetitivos
    a. Hacer acumuladores
    b. Los if para mayor y menor
    c. Los if para sacar algún criterio

c. Fuera del ciclo
    i. Cálculos generales (promedios, porcientos)
    ii. Impresiones generales.

RESULTADOS, ANÁLISIS Y DISCUSIÓN

La experiencia fue aplicada inicialmente en el grupo de tercer año de la carrera Educación Laboral e Informática de la facultad de Ciencias Técnicas; este grupo constaba con 7 alumnos en el momento en que se aplicó la experiencia: curso escolar 2014-2015, todos son varones. Solo uno de ellos proveniente del IPI (Instituto Politécnico de Informática), conocía la técnica de programación estructurada, el resto solo había visto en su vida estudiantil la técnica visual, cuando recibieron el Visual Basic en grado 12 del preuniversitario. Ninguno de ellos había profundizado en la programación de forma individual o acudiendo a los Joven Club u otra fuente de conocimiento alternativa a la docencia recibida en la escuela media. El profesor que aplicó la experiencia fue el máster Eduardo Hernández Martín, profesor Auxiliar, responsable de la disciplina LTP, que es uno de los autores del presente artículo. El docente al momento de aplicar la experiencia, llevaba 27 años de labor en la enseñanza de la programación en la educación universitaria.

La esencia de la aplicación de la experiencia se describe a continuación: en la conferencia inicial de la temática de los ciclos se expuso la clasificación y la SICA, demostrando al alumno cómo proceder con varios ejemplos; en las clases prácticas se utilizó en dos sentidos para resolver los problemas, es decir para elaborar el algoritmo que le daría solución al problema y para verificar que cualquier solución encontrada de forma heurística debía cumplir con la SICA, todo esto en el plano teórico, pues al final, la codificación y puesta a punto en máquina permitía demostrar la validez de la solución encontrada.

Esto permitió que los alumnos pudieran resolver los problemas más eficientemente, tomando en cuenta dos indicadores: tiempo para encontrar la solución y corrección de la solución encontrada.

Otro aspecto muy importante es que los alumnos pudieron resolver problemas sin aplicar el método de ensayo y error, muy generalizado en el aprendizaje de las técnicas de programación y muy criticado por los profesores de esta asignatura, pues contribuye poco al desarrollo del pensamiento lógico, en detrimento de uno de los objetivos principales en el aprendizaje de la programación de computadoras.

Este método se utiliza mucho por los alumnos, cuando están en el laboratorio de computadoras, pues en vez de hacer un proyecto bien analizado como antesala a la puesta a punto, van organizando su programa poniendo instrucciones, muchas veces en lugares no adecuados y corrigiendo ese código a medida que la computadora les da los resultados de la ejecución.

Esta situación también se traslada cuando están programando sin el uso de la computadora, ya que colocan órdenes que saben que deben ser utilizadas para resolver el problema, fuera de contexto, es decir en lugares no adecuados, es aquí donde la SICA tiene un uso potencial elevado, pues le permite al alumno organizar las órdenes que va a dar la computadora y entonces el programa queda correctamente elaborado.

En un segundo nivel de aplicación se utilizó en consulta a 10 alumnos de bajo rendimiento, de tercer año de la carrera Educación Laboral e Informática, al finalizar el primer semestre del actual curso 15-16. Estos alumnos estaban diagnosticados con posibilidades de desaprobar el examen final de la asignatura Lenguajes y Técnicas de Programación 1, pues estaban categorizados de mal o regular en el corte evaluativo, previo a la culminación del semestre.

La profesora de esta asignatura fue la Lic Ivelisse Machín Torres, graduada de Ingeniería Informática, con categoría de profesor Instructor y dos años de experiencia, el tema de ciclos fue impartido bajo los cánones tradicionales, sin la utilización de la SICA y en el examen final estaba involucrado en tres preguntas: una propiamente de ciclos, otra de arreglos unidimensionales y otra de arreglos con datos de tipo registros, por ende el dominio de esta estructura era vital para aprobar el examen.

Con la anuencia de la docente, se estableció un sistema de consultas, impartidas por el docente Eduardo Hernández Martín, profesor principal de disciplina, y se obtuvieron excelente resultados, pues de los 10 alumnos, solo 2 desaprobaron el examen ordinario y después pudieron vencer el examen en la segunda convocatoria.

A continuación se analiza un ejemplo de cómo se puede utilizar el modelo tradicional para la solución de problemas incluyéndole la SICA particular, en aras de elaborar el programa que resuelva un problema de la estructura cíclica determinada variable.

Problema: Se tiene un listado de n estaturas y sexos. Se desea obtener la menor estatura de los hombres y el promedio de estatura general.

Al aplicar el modelo para resolver problemas mediante la programación estructurada se obtiene lo siguiente:

Paso 1. Análisis del problema

1. ¿Qué dan como datos de entrada?

Respuesta: Los datos son:

n que es la cantidad de personas.

n estaturas

n sexos

2. ¿Qué debo obtener como resultado final?

El menor valor de la lista de estatura, siempre y cuando en la lista de sexos esté indicado que el sexo correspondiente es masculino y el promedio de estatura general.

3. ¿Cuál sería una idea primaria de solución? En la respuesta a esta pregunta influye la clasificación de los ciclos propuesta por los autores del artículo.

Un ciclo determinado variable, esto se obtiene al aplicar la clasificación de los ciclos, pues de la respuesta a la primera pregunta aparecen el dato n (cantidad de repeticiones), siendo esta información válida para reconocer el tipo de ciclo.

En esta repetición se entraran las dos listas, una con el sexo y otra con la estatura, desde la misma entrada hay que ir sumando las estaturas (para calcular el promedio) y además se debe ir filtrando (mediante un if) para obtener solo a los masculinos, entonces cuando ese garantice esa condición se debe comparar y actualizar la estatura contra un valor inicialmente prefijado y que se va

actualizando (if anidado) y así obtener la menor estatura. Finalmente se divide la suma entre n y se obtiene así el promedio de estatura.

4. ¿Cuáles serían los datos auxiliares?
    a. La variable de control del ciclo (i)
    b. El sumador de estaturas (s)

Paso 2. Determinación del algoritmo de solución. (**Aquí es donde se introduce la SICA, en este caso se va a introducir el caso particular del ciclo determinado variable).**

Utilizando el modelo tradicional, a partir del anterior análisis o de uno cercano a él, y habiendo estudiado la teoría y ejemplos, se presupone que los alumnos son capaces de obtener un algoritmo correcto y su correspondiente código; pero la experiencia de varios años impartiendo la disciplina de programación, permite aseverar que solo alumnos aventajados son capaces de elaborar algoritmos correctos partiendo de las bases anteriores.

Los errores más comunes están dados por: realizar inicializaciones dentro del ciclo, hacer cálculos generales dentro del ciclo, no hacer inicializaciones, imprimir dentro del ciclo, hacer condiciones fuera del ciclo, entrar datos antes del ciclo.

En esta etapa, es donde la SICA es una herramienta de gran apoyo, pues guía a los alumnos acerca de las órdenes esenciales que debe realizar y ubica en qué parte de la estructura deben quedar esas órdenes.

En el ejemplo que se está explicando, el alumno sabe que tiene que calcular el menor dato, sabe por teoría que un menor se inicializa en un número grande y la SICA le indica donde se coloca esa inicialización, también de la etapa de análisis se desprende que debe tener un sumador para calcular el promedio, por la teoría del sumador sabe que debe inicializarlo en cero y nuevamente la SICA le dice dónde debe colocar esa orden, de igual forma se comporta la entrada de datos, pues en la etapa de análisis hay una pregunta para organizar el pensamiento acerca de cuáles son los datos de entrada y la SICA le permite al alumno ubicar esta entrada dentro del ciclo y no cometer uno de los errores más comunes que es colocarlas antes del ciclo, de la misma forma se comportan el cálculo del sumador y la estructura alternativa que permite obtener el menor dato, que muchas veces son colocadas por el alumno fuera del ciclo y que la

SICA los ubica con claridad dentro del ciclo, que es donde deben ir; ya después en la etapa fuera del ciclo, se colocan las instrucciones que permiten calcular el promedio, e imprimir los cálculos obtenidos.

Tabla 1. Ejemplo de aplicación de la SICA para obtener el algoritmo.

SICA (ciclo determinado variable)	Algoritmo obtenido
Antes del ciclo Inicializaciones Entrar n	me ← 10000 s ← 0 Entrar n
Dentro del ciclo Entrar datos Procesos repetitivos	Para i←1 hasta n hacer Entrar sexo Entrar estatura s←s+estatura Si sexo = “M” entonces Si estatura < me entonces Me ← estatura Fin del para
Fuera del ciclo Cálculos finales Impresiones finales	prom ← suma/n Mostrar Me Mostrar Prom

Fuente: Elaboración propia

Paso 3. Elaboración del programa.

Tabla 2. Codificación del algoritmo ya elaborado. Esta etapa es más simple, pues consiste en la traducción de un algoritmo a un lenguaje de programación, como es lógico, un buen resultado, depende de la calidad del algoritmo elaborado en la etapa anterior.

Algoritmo obtenido	Programa (Object Pascal)

me ← 10000
s ← 0
Entrar n
	Begin
	me :=10000;
	s:=0;

	Write(‘Entre n ‘); readln(n);

Para i←1 hasta n hacer
Entrar sexo
Entrar estatura
s←s+estatura
Si sexo = “M” entonces
Si estatura < me entonces
Me ← estatura
Fin del para
	for i.=1 to n do
	begin
	write(‘Entre sexo ‘); readln(sexo);
	write(‘Entre estatura ‘);readln(est);
	s:=s+est;
	if sexo =’M’’ then
	if est < me then me .=est;
	end;

prom ← suma/n Mostrar Me Mostrar Prom	prom:=s/n; writeln(‘Menor estatura = ‘,me:6:2); writeln(‘El promedio pedido =’,prom:6:2); end.

Fuente: Elaboración propia

Paso 4. Prueba del programa. Este paso se orienta a los alumnos que lo realicen en tiempo de máquina.

Paso 5. Documentación del programa. Este paso solo se orienta realizarlo cuando se discute un proyecto o tarea extra clase.

Paso 6. Ejecución y uso del programa. Este paso solo se ejecuta cuando se realiza un software a nivel empresarial o un producto que será aplicado.

CONCLUSIONES

Con el presente trabajo se ha brindado una sucesión de indicaciones de carácter algorítmico, apoyada con un ejemplo de su uso, que facilita a los profesores y alumnos de programación la realización del proceso de enseñanza aprendizaje de los ciclos, desde el punto de vista de la programación estructurada.

REFERENCIAS BIBLIOGRÁFICAS

Ballester, S., Santana H., Hernández S., Cruz I., Arango C., Garcìa M.,…, Torres P. (2007). *Metodología de la enseñanza de la matemática*. La Habana, Cuba: Editorial Pueblo y Educación.

Calderón, M., García C. (2004). *Lógica de Programación*. La Habana, Cuba: Editorial Pueblo y Educación.

Díaz, J., Pérez F. (2001). *Delphi 5 Básico*. La Habana, Cuba: Editorial Pueblo y Educación.

Expósito, C., Cruañas, J., Gener, E., de la Noval, N., Rivero, A., Peñalver, L. (2001). *Elementos de la Metodología de la Informática*. La Habana, Cuba: Editorial Pueblo y Educación.

Fidalgo, A. (2012). *Bucles.* Universidad Politécnica de Madrid. Recuperado de https://innovacioneducativa.files.wordpress.com/2012/10/buccles.pdf

García de Jalón, J. (2013). *Aprenda Lenguaje ANSI C como si estuviera en primero*. San Sebastián: Editorial Universitaria de Navarra.

Hernández, E., Hondal, V., González, N. (2005). Los lazos. Su didáctica bajo la luz de la programación visual*. Revista Pedagogía y Sociedad*. *6*(13). Recuperado de www.pedsoc.rimed.cu

Hernández, E., Martínez, Y. (2002). Los ciclos en la programación visual*. Revista Infociencia*. *6*(2). Recuperado de www.infociencia.idict.cu.

Hernández, E., Martínez, Y., Fleites, L. (2012). Utilización de Problemas en las clases de Programación para la motivación de los alumnos. *Revista Pedagogía y Sociedad*. *15*(35). Recuperado de www.pedsoc.rimed.cu

Hernández, E., Machín, I., Fleites, L. (2015). Los lazos. Su didáctica bajo la luz de la programación estructurada. [CD]. Memorias de Universidad 2016. Sancti Spíritus, Cuba: Universidad de Sancti Spiritus "José Martí Pérez".

Martínez, A. (2012). Concepto de algoritmo, Diagrama de flujo y pseudocódigo. Recuperado de https://andresmtzg.wordpress.com/2012/09/27/concepto-de-algoritmo-diagrama-de-flujo-y-pseudocodigo

Ochoa, G. (2014). Estructuras repetitivas. Universidad Simón Bolívar. Caracas. Recuperado de www.ldc.usb.ve/ gabro/teaching/CI2125/Clase5_ciclos.htm

Peralta, L. (2014). Introducción a la programación. Recuperado de: http://enriquebarrueto0.tripod.com/algoritmos/algor01.pdf

Pozo, S. (2013). *Curso de C++.* Madrid. Recuperado de http://c.conclase.net

Sivira, A., Longone M. T., Rodríguez A. (2015). Programación Estructurada. Universidad Politécnica Territorial del Estado Lara "Andres Eloy Blanco". Recuperado de: http://aprendealgoritmo.com.ve

Trejos, O., (2013). Relaciones de aprendizaje significativo entre dos paradigmas de programación a partir de dos lenguajes de programación. *Revista Tecnura*. *18*(41). 91-102. Recuperado de www.scielo.org.co/scielo.php?script=sci-arttext&pid=s0123-921x2014000300008&lang=pt

Vildósola, S., del Cañal, E., González, S., Mateu, M., Toledo, M., de la Torres, E. (1990). Fundamentos de Programación. La Habana, Cuba: Editorial Pueblo y Educación.

Recibido: 14 de junio de 2015

Aprobado: 3 de marzo de 2016