aprenderaprogramar.com

Verificación, prueba y depuración de algoritmos aprendiendo a programar. (CU00229A)

Sección: Cursos

Categoría: Curso Bases de la programación Nivel II

Fecha revisión: 2024

Autor: Mario R. Rancel

Resumen: Entrega nº 28 del Curso Bases de la programación Nivel II

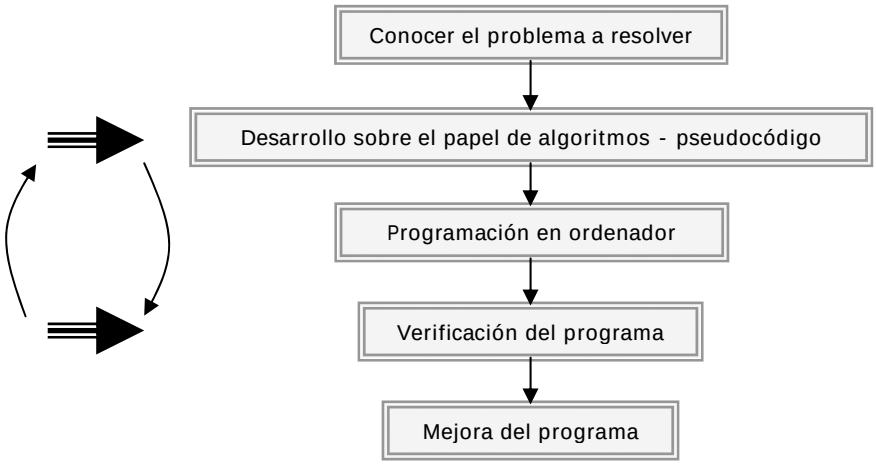
24

VERIFICACIÓN, PRUEBA Y DEPURACIÓN DE ALGORITMOS

Llamamos verificación de algoritmos a la comprobación del correcto funcionamiento del pseudocódigo planteado. Los conceptos de verificación, prueba y depuración son en cierta medida similares y en cierta medida distintos. Nosotros de momento no vamos a entrar en detalles de momento. Hemos comentado en diferentes ocasiones que el concepto de algoritmo se entrelaza con el de programa hasta, en ocasiones, llegar a confundirse. En este caso hablaremos de verificación de algoritmos pero todo lo expuesto nos será útil para enfrentarnos a la verificación del programa.

Al hablar de verificación estamos tratando una parte de lo que supone el desarrollo de algoritmos – pseudocódigo así como del proceso posterior a disponer del código del programa. Habrá que entender que para programas complejos, aunque se parta de un conocimiento profundo del problema a resolver, el proceso que va desde generar algoritmos hasta mejorar el programa puede convertirse en un recorrido de ida y vuelta con cierta interposición entre fases. Es decir, mientras que en un problema sencillo el esquema lineal puede ser ajustado a la realidad, en un problema complejo puede ser necesaria una cierta superposición entre desarrollo, programación, verificación y mejora. Es posible que al realizar la verificación del programa o partes del programa descubramos defectos que nos obliguen a volver a la parte de desarrollo. Las verificaciones, aunque tienen momentos principales, también es habitual que se extiendan a lo largo de las fases de desarrollo, programación y mejora.

Si recordamos un esquema que nos sirve de guía en el proceso de programación:



En resumen, si en su momento dijimos que aprender a desarrollar algoritmos eficientes es aprender a programar, diremos ahora que aprender a verificar algoritmos es aprender a verificar programas.

Todo programador, de forma natural, desarrolla un algoritmo en base a una construcción mental preliminar que le da pie a pensar que el algoritmo funciona.

Si escribimos:

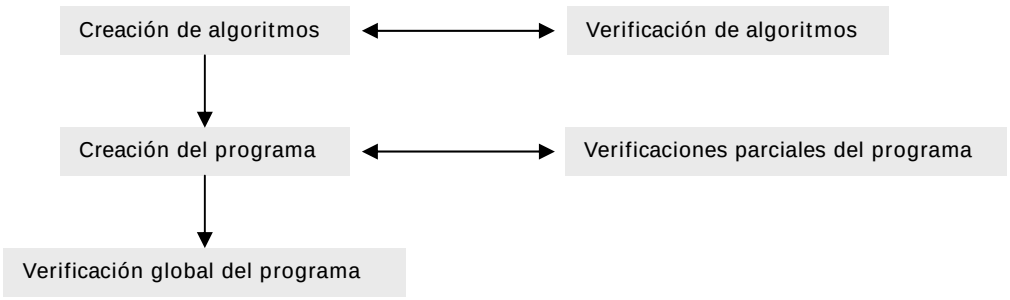
```
[Ejemplo aprenderaprogramar.com]
Desde i = 1 hasta 5 Hacer
        Desde j = 1 hasta 5 Hacer
                A = i * j
                Mostrar A
        Siguiente j
Siguiente i
```

Nuestro cerebro ha creado una estructura en base a nuestra experiencia, conocimientos, etc. y preliminarmente valora que debe funcionar para cumplir nuestros objetivos. Esta verificación mental preliminar no debe bastar desde el momento en que exista un mínimo de complejidad, pues supondría arriesgarnos a que existan errores que pueden comprometer todo el desarrollo del programa. Por ello, usaremos una o varias de las técnicas indicadas a continuación:

- Verificación mental.
- Verificación por seguimiento escrito.
- Verificación por seguimiento con tablas de variables.
- Verificación por seguimiento con desarrollo en un lenguaje.
- Verificación por seguimiento con un lenguaje y paso a paso.

Aunque hemos dicho que la verificación del algoritmo guarda estrecha relación con lo que sería verificación del programa, podemos matizar esta afirmación.

El proceso viene siendo algo así:



La creación del programa debe comenzar con los algoritmos que lo constituyen ya verificados. Veamos por qué. Un programa correcto se construye a partir de algoritmos eficientes + uso eficiente del lenguaje y de los recursos del ordenador. Si partimos de algoritmos eficientes cuando nos ponemos delante del ordenador a escribir el programa, sólo nos quedan por resolver aspectos del lenguaje y

recursos del ordenador. Si partimos de algoritmos no eficientes, los dos problemas se superponen y se magnifican al añadirse a la aparición de un error la dificultad para saber si está asociado al diseño del algoritmo o al uso del lenguaje y recursos.

La verificación de algoritmos suele realizarse para procesos parciales como los contenidos en módulos o partes de módulos, ya que una verificación global del algoritmo puede resultar muy costosa y compleja. Durante la creación del programa es posible que se haga necesario retocar algunos aspectos de algoritmos previamente verificados, en términos de puesta a punto.

Veremos a continuación las diferentes técnicas de verificación usando como ejemplo base el doble anidamiento de bucles *Desde* empleado al principio de este apartado, con un bucle externo controlado por la variable *i*, un bucle interno controlado por la variable *j* y una variable *A* que adquiere valores en función de *i* y *j*.

Próxima entrega: CU00230A

Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente:
http://www.aprenderaprogramar.com/index.php?option=com_content&view=category&id=36&Itemid=60