Metodología para la solución de problemas

DISEÑO ESTRUCTURADO DE ALGORITMOS

M.C. Carlos Rojas Sánchez[1]

[1]Licenciatura en Informática
Universidad del Mar :: Puerto Escondido

.: Diseño Estructurado de Algoritmos :.

Etapas

Etapas

- Definición del problema

Etapas

- Definición del problema
- Análisis de los datos

Etapas

- Definición del problema
- Análisis de los datos
- Diseño de la solución

Etapas

- Definición del problema
- Análisis de los datos
- Diseño de la solución
- Codificación

Etapas

- Definición del problema
- Análisis de los datos
- Diseño de la solución
- Codificación
- Prueba y depuración

Etapas

- Definición del problema
- Análisis de los datos
- Diseño de la solución
- Codificación
- Prueba y depuración
- Documentación

Etapas

- Definición del problema
- Análisis de los datos
- Diseño de la solución
- Codificación
- Prueba y depuración
- Documentación
- Mantenimiento

1. Definición del problema

Definición del problema

Esta fase la proporciona el enunciado del problema, el cual requiere una definicón clara y precisa (no debe ser ambiguo). Es importante que se entienda prefectamente lo que pretendemos que haga la computadora para poder continuar con la siguiente etapa.

2 Análisis de los datos

Análisis de los datos

Análisis del problema

- ¿Qué información se necesita para obtener el resultado deseado? - Datos de Entrada
- ¿Qué información desea producir? - Datos de Salida
- Los métodos y fórmulas que se necesitan para procesar los datos y producir esa salida

3 Diseño de la solución

Diseño de la solución

Diseño y técnicas para la formulación de un algoritmo

La etapa de diseño se centra en desarrollar el algoritmo basándonos en las especificaciones de la etapa del análisis; podemos representar un algoritmo mediante el diagrama de flujo o el pseudocódigo.

4 Codificación

Codificación

En la etapa de codificación se transcribe el algoritmo definido en la etapa de diseño en un código reconocido por la computadora; es decir, en un lenguaje de programación; a éste se le conoce como código fuente.

5 Prueba y depuración

Pruebas y depuración

Las pruebas consisten en capturar datos hasta que el programa funcione correctamente. A la actividad de localizar errores se le llama depuración.

Pruebas

Sintaxis

Las pruebas de sintaxis se ejecutan primero, son las más sencillas y las realiza el compilador del programa cada vez que se ejecuta el programa hasta que el código no presente errores, es decir que la sintaxis que requiere el lenguaje sea la correcta, de lo contrario el propio compilador va mostrando los errores encontrados para que se modifiquen y se pueda ejecutar el código; estos errores pueden ser falta de paréntesis, o puntos y comas o palabras reservadas mal escritas.

Pruebas

Lógica

Las puebas lógicas son las más complicadas ya que éstas las realiza el programador; consisten en la captura de diferentes valores y revisar que el resultado sea el deseado, es decir el programador tendría que modificar el código hasta que el programa funcione correctamente.

6 Documentación

Documentación

- Es la guía o comunicación escrita que permite al programador o al usuario conocer la funcionalidad del programa.
- La documentación sirve para que el código fuente sea más comprensible para el programador o para otros programadores que tengan que utilizarlo, así como para facilitar futuras modificaciones (mantenimiento)

Documentación

Tipos

Interna	Se genera en el mismo código y generalmente es mediante comentarios.
Externa	Son los manuales y es independiente al programa. También puede ser la ayuda en el mismo software

7 Mantenimiento

Mantenimiento

- Se dice que un programa no se termina al 100 %, ya que es necesario hacer algún cambio, ajuste o complementación para que siga funcionando correctamente; para llevarlo a cabo se requiere que el programa esté bien documentado.
- Todos los programas tienen actualizaciones, por lo que surgen versiones diferentes.

8 Ejemplo: Algoritmo de Burbuja

Ordenamiento por intercambio

Funciona revisando cada elemento de la lista que va a ser ordenada con el siguiente, intercambiándolos de posición si están en el orden equivocado. Es necesario revisar varias veces toda la lista hasta que no se necesiten más intercambios, lo cual significa que la lista está ordenada. Este algoritmo obtiene su nombre de la forma con la que suben por la lista los elementos durante los intercambios, como si fueran pequeñas "burbujas". También es conocido como el método del intercambio directo. Dado que solo usa comparaciones para operar elementos, se lo considera un algoritmo de comparación, siendo el más sencillo de implementar.

Ejemplo

Ordenar del menor al mayor los siguientes números: "9 6 5 8 2 1"

Primera vuelta

$$
\begin{aligned}
(\ 9\ 6\ 5\ 8\ 2\ 1\ ) &\rightarrow (\ 6\ 9\ 5\ 8\ 2\ 1\ ) \\
(\ 6\ 9\ 5\ 8\ 2\ 1\ ) &\rightarrow (\ 6\ 5\ 9\ 8\ 2\ 1\ ) \\
(\ 6\ 5\ 9\ 8\ 2\ 1\ ) &\rightarrow (\ 6\ 5\ 8\ 9\ 2\ 1\ ) \\
(\ 6\ 5\ 8\ 9\ 2\ 1\ ) &\rightarrow (\ 6\ 5\ 8\ 2\ 9\ 1\ ) \\
(\ 6\ 5\ 8\ 2\ 9\ 1\ ) &\rightarrow (\ 6\ 5\ 8\ 2\ 1\ 9\ )
\end{aligned}
$$

Segunda vuelta

$$
\begin{aligned}
(\ 6\ 5\ 8\ 2\ 1\ 9\ ) &\rightarrow (\ 5\ 6\ 8\ 2\ 1\ 9\ ) \\
(\ 5\ 6\ 8\ 2\ 1\ 9\ ) &\rightarrow (\ 5\ 6\ 8\ 2\ 1\ 9\ ) \\
(\ 5\ 6\ 8\ 2\ 1\ 9\ ) &\rightarrow (\ 5\ 6\ 2\ 8\ 1\ 9\ ) \\
(\ 5\ 6\ 2\ 8\ 1\ 9\ ) &\rightarrow (\ 5\ 6\ 2\ 1\ 8\ 9\ ) \\
(\ 5\ 6\ 2\ 1\ 8\ 9\ ) &\rightarrow (\ 5\ 6\ 2\ 1\ 8\ 9\ )
\end{aligned}
$$

Tercera vuelta

$$(\ 5\ 6\ 2\ 1\ 8\ 9\ ) \rightarrow (\ 5\ 6\ 2\ 1\ 8\ 9\ )$$
$$(\ 5\ 6\ 2\ 1\ 8\ 9\ ) \rightarrow (\ 5\ 2\ 6\ 1\ 8\ 9\ )$$
$$(\ 5\ 2\ 6\ 1\ 8\ 9\ ) \rightarrow (\ 5\ 2\ 1\ 6\ 8\ 9\ )$$
$$(\ 5\ 2\ 1\ 6\ 8\ 9\ ) \rightarrow (\ 5\ 2\ 1\ 6\ 8\ 9\ )$$
$$(\ 5\ 2\ 1\ 6\ 8\ 9\ ) \rightarrow (\ 5\ 2\ 1\ 6\ 8\ 9\ )$$

Cuarta vuelta

$$(\ 5\ 2\ 1\ 6\ 8\ 9\ ) \rightarrow (\ 2\ 5\ 1\ 6\ 8\ 9\ )$$
$$(\ 2\ 5\ 1\ 6\ 8\ 9\ ) \rightarrow (\ 2\ 1\ 5\ 6\ 8\ 9\ )$$
$$(\ 2\ 1\ 5\ 6\ 8\ 9\ ) \rightarrow (\ 2\ 1\ 5\ 6\ 8\ 9\ )$$
$$(\ 2\ 1\ 5\ 6\ 8\ 9\ ) \rightarrow (\ 2\ 1\ 5\ 6\ 8\ 9\ )$$
$$(\ 2\ 1\ 5\ 6\ 8\ 9\ ) \rightarrow (\ 2\ 1\ 5\ 6\ 8\ 9\ )$$

Quinta vuelta

$$
\begin{aligned}
(\ 2\ 1\ 5\ 6\ 8\ 9\ ) &\rightarrow (\ 1\ 2\ 5\ 6\ 8\ 9\ ) \\
(\ 1\ 2\ 5\ 6\ 8\ 9\ ) &\rightarrow (\ 1\ 2\ 5\ 6\ 8\ 9\ ) \\
(\ 1\ 2\ 5\ 6\ 8\ 9\ ) &\rightarrow (\ 1\ 2\ 5\ 6\ 8\ 9\ ) \\
(\ 1\ 2\ 5\ 6\ 8\ 9\ ) &\rightarrow (\ 1\ 2\ 5\ 6\ 8\ 9\ ) \\
(\ 1\ 2\ 5\ 6\ 8\ 9\ ) &\rightarrow (\ 1\ 2\ 5\ 6\ 8\ 9\ )
\end{aligned}
$$

Bibliografía I

Joyanes Aguilar, Luis.
Fundamentos de programación, algoritmos y estructura de datos.
McGraw-Hill. 2003, 3ª Edición.

Cairó Battistutti, Osvaldo Gabriel.
Metodología de la programación: algoritmos, diagramas de flujo y programas.
Alfaomega. 2005, 3ª Edición.

```
1  for(i=1; i < TAM ; i++)
2      for(j=0 ; j < TAM-1 ; j++)
3          if(lista[j] > lista[j+1]){
4              temp = lista[j];
5              lista[j] = lista[j+1];
6              lista[j+1] = temp;
7          }
```