Manejo de memoria dinámica

Al escribir un programa, hemos tenido que decidir sobre la cantidad máxima de memoria que sería necesaria para nuestras matrices, y dejar esto de lado en las declaraciones. Si ejecutamos el programa en una pequeña muestra, entonces gran parte de este espacio nunca será utilizado. Si decidimos ejecutar el programa en un gran conjunto de datos, entonces podemos agotar el espacio establecido a un lado y encuentro desbordamiento, incluso cuando la memoria de la computadora en sí misma no se utiliza por completo, simplemente porque nuestros límites originales en la matriz eran demasiado pequeños. Incluso si tenemos cuidado de declarar nuestras matrices lo suficientemente grandes como para agotar todas las matrices disponibles memoria, todavía podemos encontrar desbordamiento, ya que una matriz puede alcanzar su límite mientras gran parte del espacio no utilizado permanece en otros. Como diferentes ejecuciones del mismo programa puede hacer que diferentes listas crezcan o se encojan, puede ser imposible saber antes de programa realmente ejecuta qué listas se desbordarán. Ahora exhibimos una forma de mantener listas y otras estructuras de datos en la memoria sin utilizando matrices, por lo que podemos evitar estas dificultades.

Se entiende el hecho de crear variables anónimas, es decir, reservar espacio en memoria para estas variables en tiempo de ejecución, y también de liberar el espacio ocupado en memoria por una variable anónima, asimismo en tiempo de ejecución, cuando esa variable ya no es necesaria; asignación y posible recuperación de memoria durante la ejecución de un programa y bajo su control.

La memoria que ocupan los objetos con tiempo de vida no predecible a partir del código fuente se gestiona dinámicamente: cuando se crea un objeto en tiempo de ejecución, se pide memoria para él, y cuando se elimina, la memoria se libera. En algunos lenguajes encontramos características que necesitan una gestión de memoria con reserva dinámica:

- Los operadores new y dispose del PASCAL.

- La creación de elementos de listas en LISP y su liberación automática (garbage collection) por ausencia de referencias a la memoria.
- Las llamadas a funciones de biblioteca malloc y free del C.
- La creación de objetos en Java y su eliminación cuando ninguna variable mantiene referencias a dichos objetos.
- Los operadores new y delete o las llamadas implícitas que comportan la creación y destrucción de objetos en C++.
- Etc.

La zona de memoria destinada a atender las demandas de memoria dinámica se denomina heap. Vamos a estudiar muy por encima el problema de la gestión del heap.

Referencias

1. Estructura de datos, Algoritmos, abstracción y objetos, 3ª edición, McGraw-Hill, Madrid, 1999.
2. Data Structures and Program Design, Robert L. Kruse, Pretince-Hall, New Jerser, 1984, Pag. 55.
3. Data Structures and Program Design, Robert L. Kruse, Pretince-Hall, New Jerser, 1984, Pag. 53.
4. Aho V, Hopcroft, y Ullman J, Estructura de datos y algoritmos, México, Prentice-Hall, 1983.
5. Carrano M, Hellman P, Data Structures annual problems solving with turbo pascal, California, Benjamin/Cummings, 1993.