Tutoría 2: Estructuras de Datos

Cristóbal Ganther, Felipe Vera

2 de septiembre de 2012

Los contenidos que se verán en esta tutoría serán los siguientes.

- Punteros, punteros dobles, paso por valor, paso por referencia y operador `&`.
- Elaborar nuevos tipos de datos: `typedef` y `struct`.
- Reservar memoria en el *heap*: `stack`, `heap`, `malloc` y `free`.
- Ingresar datos en consola: `scanf`
- Manejar estructuras de datos dinámicas: listas, inserción en la cabeza, inserción en la cola, inserción en la posición especificada, búsqueda, eliminación.

Instrucciones

1. Introducción a listas enlazadas.
   a) Cree una estructura llamada `s_product` cuyos miembros sean: `name`, `trademark`, `amount` y `next`. Los miembros `name` y `trademark` deben ser de tipo `(char*);` El miembro next debe ser de tipo `(struct s_product *)`.
   b) Redefina el nombre del tipo de forma que se pueda referir al struct con el identificador `product`. (Note que anidar las sentencias typedef y struct no es recomendado por el estándar de codificación GNU).
   c) Finalmente defina una función con prototipo: `product * new_product (char * name, char * trademark, int amount)`. La nueva estructura y sus miembros deben estar almacenados en *HEAP*. Ayúdese de la función `strdup` para copiar los strings entregados a su función. ¿A dónde tiene que apuntar `next` en los nodos recién creados?
2. Insertando nodos al comienzo de la lista.
   a) Defina una función con prototipo: `push (product ** list, product * prod)`. Esta función debe agregar el producto `prod` al comienzo de la lista `list`. Procure que su función funcione correctamente si alguno de los parámetros es `NULL`.
3. Interfaz de usuario y `scanf`.
   a) En su función `main` permita que el usuario pueda agregar productos a una lista usando `scanf`.
   b) Una vez que el usuario termine de ingresar los productos, muestre la lista de todos los productos por pantalla. No necesita imprimir todos los miembros de cada nodo.
4. Lectura básica de una lista.
   a) Defina una función con prototipo: `product * get (product * list, int number)`. Esta función debe retornar el nodo en la posición `number` de la lista `list`.

b) Una vez que el usuario termine de ingresar los productos, permita que éste pueda hacer consultas utilizando la función definida en el punto *(4a)*.

5. Inserciones

   a) Defina una función con prototipo: `insert (product ** list, product * prod, int number)`. Esta función debe insertar un producto `prod` en la posición `number` de la lista `list`. ¡Ponga atención a los casos extremos!

   b) Permita al usuario insertar elementos a la lista.

6. Salida del programa y limpieza de memoria.

   a) Permita que el usuario pueda salir del programa.

   b) Defina una función con prototipo: `delete (product ** list, int number)`. Esta función debe eliminar el elemento `number` de la lista list y de el *HEAP* (función `free`). Recuerde que también los miembros `name` y `trademark` están almacenados en *HEAP*.

   c) Haga que antes de salir se eliminen todos los elementos en `HEAP`.

¡Recuerde dejar la máquina virtual en orden una vez que termine su trabajo!

¡Pregunte apenas tenga una duda!