ASSIGNATURA: Algorismes II

DATA: 10−03−1997

Para que no hubiese problemas de divisibilidad entre los caracteres y el signo del apuntador, he sustituido el caracter por el caracter ^.

Eliminar elemento

Eliminaremos un elemento que es el primero de la lista

Si top=NIL entonces "Lista vacia"

sino inicio

p:=top

top:=top^.sig

dispose(p)

fin

Si top es distinto de cero, la lista como minimo tendra un elemento.

Eliminamos el último elemento

Primero hemos de buscar el elemento ha eliminar. Para buscarlo haremos un recorrido de toda la lista. Sabremos que el elemento buscado sera el último porque tendra dirección NIL.

Si top = NIL entonces "Lista vacia"

sino inicio

p:=top

mientres (p^.sig <> nil) hacer

inicio

q:=p

p:=p^.sig

fin

dispose (p)

q^.sig:=nil

fin

Eliminar el nodo del medio

Para eliminar el nodo del medio hay que seguir los siguientes pasos:

1. Encontrar el nodo i guardar la dirección del anterior

2. Enlazar q con el sig de p

3. Eliminar p

Si lo hacemos con un puntero, los pasos son los siguientes:

1. Preguntamos si hay que eliminar ese nodo

2. Mirar si el siguiente elemento es el que se tiene que borrar

3. Como (p) tiene las dos direcciones (la del actual y la del siguiente), entonces borrara el

nodo del siguiente.

Lista lineal encabezado circular

Se supone que cada nodo tiene dos campos, puede tener más.

Este tipo de listas son circulares porque el último elemento de la lista, el campo siguiente, tiene la dirección del primer elemento de la lista. Esto genera una serie de consideraciones:

− Cada nodo de una lista circular es accesible desde cualquier otro nodo sin que se tenga

que volver acceder al principio de la lista.

− No hay un primer nodo, ni un último nodo

Estas listas circulares, generan un problema, que no sabemos cuando acaba la lista, y podemos dar vueltas indefinidamente (bucles infinitos).

Este problema se puede solucionar de dos formas: la primera es preguntar por el top y la otra solución , es utilizar un nodo centinela, que nos sirve para indicar cuando damos la vuelta. En el campo información del nodo centinela ponemos una marca que nos indica que ese es el nodo centinela. Por tanto en la lista tendremos un nodo más.

1 3 5 x

En vez de preguntar por el top, preguntaremos por el campo información.

Hay que recordar que estamos trabajando con enlaces simples de dirección, es decir, lo que une un nodo a otro es un enlace (una flecha).

Creación de una lista enlazada circular

En este algortimo no utilizaremos un nodo centinela.

Si top = NIL entonces

inicio

new (top) *reservamos espacio*

top^.info.='dato'

top

top^.sig:=top (*)

sino inicio

p:=top *buscamos el elemento*

mientras (p^.sig <> top) hacer *preguntamos por el top*

p:=p^.sig

fin

new(q) *reservamos espacio*

q^.info:='dato'

top p q

q^.sig:=top 1 2 (**)

top p q

p^.sig:=q 1 2

fin

(*) Ya tenemos la lista circular. Hemos creado el primer nodo.

(**) No importa si tenemos dos direcciones apuntando al TOP, lo importante es no

perder ninguna dirección.

Listas doblemente encadenada

Significa que tiene dos campos de dirección

TYPE nodo RECORD

Ant: ^nodo

info: integer

sig: ^nodo

END

Representación grafica:

ANT INFO SIG

ANT: Se guarda la dirección de memoria del elemento que le precede en la lista.

SIG: Guarda la dirección de memoria que le sucede en la lista.

INFO: Es la información del nodo.

Como la lista no es circular, habra un elemento con un NIL.

Como en el anterior (en las listas de simple enlace), tendremos una variable donde nos dice donde empieza la lista (no hay que perder esa dirección).

1 2 3

Convenientes de la lista doblemente enlazada: Se puede mover en dos direcciones. Es decir cada nodo puede acceder a su sucesor y a su predecesor.

Inconvenientes de la lista doblemente enlazada: Se gasta más memoria, ya que creamos un campo mas.

Propiedades:

1. P+.SIG+.ANT= P

P+.ANT+.SIG =P

Con las listas de doble enlace, p se puede referenciar de dos formas

2. Se necesitan dos punteros, p y q, para acceder a los dos extremos de la listas

3. Las operaciones de cerca, inserción y eliminación son más eficientes

4. Necesitamos un espacio adicional para almacenar otro puntero.

5. Es importante saber que no se puede utilizar una lista simple encadenada, junto a una

lista doblemente encadenada. Esto esta totalmente prohibido.