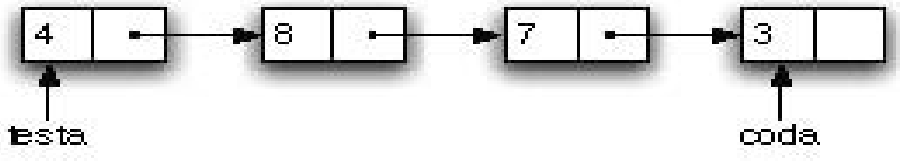
Università di Firenze – Laurea Triennale in Informatica

Corso di *Architetture degli Elaboratori* – A.A. 2018/2019

Sbrk, puntatori e liste puntate

Familiarizzare con l'utilizzo di sbrk. Le *liste collegate*, o liste a puntatori, sono strutture dati molto convenienti per la memorizzazione di insiemi di dati la cui dimensione può variare a tempo d'esecuzione. Una lista a puntatori consiste di un insieme di blocchi, detti anche *record*. Per esemplificare, supponiamo di voler realizzare una lista di elementi interi. Allora ciascun record consiste logicamente di due campi: un campo informazione che contiene, nel nostro caso, un numero intero, e un campo puntatore, ovvero l'indirizzo di memoria del successivo blocco della lista. È di fondamentale importanza assimilare questo punto: un puntatore non è altro che un indirizzo di memoria. Il campo puntatore dell'ultimo record della lista contiene un valore convenzionale `nil`, che nel nostro caso può essere identificato con 0. In generale, è importante poter accedere direttamente al primo e all'ultimo record della lista. Per tale motivo vengono generalmente previste due variabili-puntatore globali, diciamo Testa e Coda. Una lista a puntatori contenente la sequenza di numeri 4,8,7,3 potrebbe essere rappresentata graficamente come segue:



Le operazioni fondamentali che si eseguono su una lista sono quelle di inserimento, ricerca e cancellazione di un elemento.

Inserimento: le inserzioni vengono effettuate tramite il puntatore `Coda`: creato un nuovo record, bisognerà che:

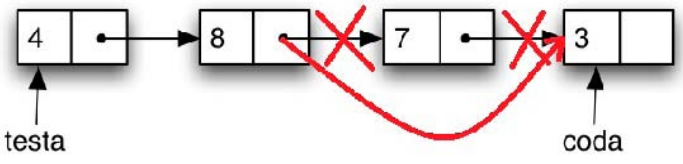
(a) il campo puntatore dell'ultimo elemento della lista (quello puntato da `Coda`) sia modificato per puntare al nuovo record; e (b) a `Coda` sia poi assegnato l'indirizzo del nuovo record. Occorre porre attenzione al primo inserimento: quando la lista è vuota, convenzionalmente si pone `Testa=Coda=0`. Se all'atto dell'inserimento risulta che `Testa==0` (la lista era vuota prima di questo inserimento), dopo le due operazioni precedenti occorrerà anche: (c) assegnare a `Testa` il valore di `Coda`.

Ricerca: la ricerca viene effettuata tramite il puntatore `Testa`: dato l'elemento da ricercare si scorre tutta la lista seguendo opportunamente i puntatori e controllando se l'elemento da cercare coincide col campo dati. La ricerca procede fino a che non si trova l'elemento o la fine della lista (puntatore al prossimo elemento = 0).

Cancellazione: la cancellazione consiste in due passi: (a) ricerca dell'elemento da cancellare (vedi la procedura di ricerca descritta sopra) (b) cancellazione dell'elemento (dipende dalla posizione dell'elemento da cancellare):

- se l'elemento da cancellare è quello di testa si modifica il puntatore `Testa` all'elemento successivo;

- se l'elemento da cancellare è quello di coda si modifica il puntatore `Coda` all'elemento precedente e si annulla il campo puntatore di tale elemento (la nuova coda) con 0;
- altrimenti si deve modificare il campo puntatore dell'elemento precedente a quello da cancellare e lo si deve far puntare all'elemento successivo a quello da cancellare. Es: supponendo di dover cancellare 7 dalla lista precedente, otterremo:



Implementazione in assembly MIPS delle liste collegate. Per la creazione di un nuovo blocco si fa uso della chiamata di sistema `sbrk` (sycall con codice 9). Essa restituisce in `$v0` l'indirizzo di un blocco libero della dimensione specificata in `$a0`. La semantica di `sbrk` è dunque analoga a quella del comando `new` in Pascal o `malloc` in C.

Esempi:

alloca_memoria.s: legge un intero e lo memorizza in un'area di memoria da 4 byte allocata dinamicamente.

listapunt.s: esemplifica il trattamento delle liste in MIPS. Esso crea una lista puntata contenente degli interi inseriti dall'utente (0 per terminare l'inserimento) e in seguito li stampa separati da spazi.

listapunt - doppia.s: Variante del precedente esercizio. Esso crea una lista doppiamente concatenata (in una lista doppiamente concatenata ogni record ha un puntatore al record precedente ed uno a quello successivo) contenente degli interi inseriti dall'utente (0 per terminare l'inserimento). Poi, per esemplificare la gestione dei puntatori, elimina il penultimo elemento della lista e stampa la lista risultante. NOTA: La lista deve essere composta da almeno 3 records, altrimenti il codice NON funziona.