

Esercitazione – Sistemi

Utilizzando un linguaggio di programmazione a scelta si realizzi una libreria di funzioni per l’allocazione dinamica della memoria. La libreria dovrà fornire tre funzioni di base sulla falsariga delle seguenti:

```
void initmem(int size)
```

Questa funzione, da richiamare all’inizio di ogni programma che si vorrà avvalere della nostra libreria di allocazione dinamica della memoria, specifica la *dimensione della memoria complessiva* che si vuole gestire. Per generare questo blocco iniziale ovviamente ci si dovrà avvalere delle funzioni predefinite dal linguaggio per l’allocazione (ad esempio la malloc del C).

```
void *alloca(int size)
```

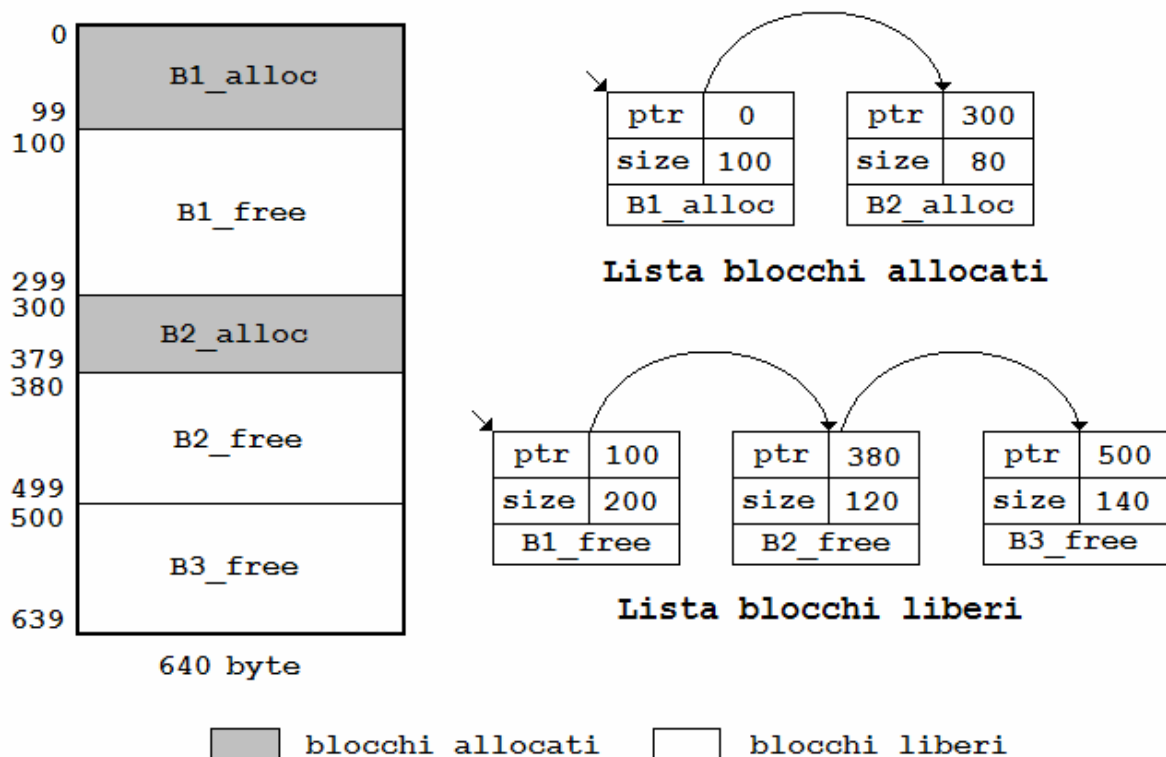
Questa funzione cerca di allocare all’interno della memoria gestita (creata con `initmem`) un blocco di `size` byte. In caso di esito positivo restituisce il puntatore al blocco allocato. Se non riesce a trovare un blocco libero sufficientemente grande la funzione restituisce `null`.

```
void libera(void *blocco)
```

Questa funzione libera la memoria precedentemente allocata per un blocco.

Suggerimenti

Si consideri la possibilità di utilizzare due liste concatenate per tenere traccia dei blocchi di memoria allocata e dei blocchi di memoria libera. L’elemento delle liste è una struttura dati che tiene traccia del puntatore all’inizio del blocco e la dimensione del blocco.



In questo scenario per allocare un nuovo blocco si deve scorrere la lista dei blocchi liberi alla ricerca di un *blocco adatto a contenere quello che si deve allocare*.

Esempi

Supponendo di eseguire una `alloca` da 100 byte potremmo individuare `B1_free` come blocco libero. Solo 100 byte di `B1_free` sono realmente necessari, per cui inseriremo nella lista dei blocchi allocati un blocco con `ptr 100` e `size 100`, modificando `B1_free` con i nuovi valori `ptr 200` e `size 100` (ovviamente se avessi voluto allocare 200 byte `B1_free` sarebbe sparito). La `alloca` restituirà in questo caso 100, il puntatore all'inizio del blocco allocato.

Eseguire a questo punto una `libera` con lo stesso puntatore (valore 100) comporterà cercare nella lista dei blocchi allocati il blocco relativo a tale puntatore. Una volta individuato, *tale blocco dovrà essere rimosso dalla lista e inserito nella lista dei blocchi liberi*.

Facoltativo (per voti superiori a 8)

Implementare una gestione avanzata dell'allocazione dinamica della memoria, che risolva in qualche modo il problema della **frammentazione**. Come si sarà notato nell'esempio precedente, *pur disponendo di 360 byte liberi non sarebbe stato possibile allocare un blocco di 300 byte*. Questo perché non esiste un singolo blocco grande abbastanza a contenere 300 byte. Questo fenomeno è appunto noto con il nome di *frammentazione*, e si manifesta dopo un po' che si lavora allocando e rilasciando blocchi.

Qualche idea...

- Scandire ogni tanto la lista dei blocchi liberi alla ricerca di blocchi consecutivi da 'accorpare'. Ad esempio `B2_free` e `B3_free` potrebbero essere fusi assieme creando un blocco più grande ed in grado di soddisfare richieste fino a 260 byte.
- Riorganizzare periodicamente le liste dei blocchi liberi e dei blocchi allocati effettuando un *compattamento* della memoria. Si tratta in buona sostanza di spostare tutti i blocchi allocati in modo che occupino uno spazio consecutivo senza 'buchi'.

