

# ASSEGNAMENTO 2: BATTAGLIA NAVALE, IL GIOCO CONTRO IL COMPUTER

Informatica, Corso di Laurea in Fisica, Università di Pisa

AA 2018/19

## 1 Outline

Nel secondo assegnamento realizzeremo un semplice gioco interattivo per la battaglia navale in cui un umano gioca con il computer.

## 2 Come è fatta l'area di gioco

L'area in cui si trova la flotta è quadrata e suddivisa in sottoaree unitarie quadrate dette *celle*. Le navi possono occupare una, due o tre celle e sono dette di dimensione uno, due e tre rispettivamente. Non è possibile avere due navi in celle adiacenti.

L'area è rappresentata da una matrice di char memorizzata usando la tecnica dell'array di puntatori a righe ed abbiamo condizioni periodiche di bordo, ovvero il primo elemento e l'ultimo di ogni riga/colonna sono adiacenti. Ogni cella può essere vuota, contenere una nave o parte di essa o contenere una nave (o parte di una nave) colpita. Se la cella è vuota contiene il carattere '.', se contiene una nave di lunghezza 1 contiene il carattere '1', se contiene una nave lunga 2 il carattere '2', se contiene una nave lunga 3 il carattere '3'. Per segnalare navi o parti di navi colpite si usano: il carattere 'a' per le navi lunghe 1, il carattere 'b' per le navi lunghe 2, e il carattere 'c' per le navi lunghe 3. Il carattere 'X' segnala una casella vuota colpita (*acqua*).

Ad esempio una configurazione iniziale di 10x10 di celle con 4 navi di dimensione 1, 4 di dimensione 2 e 2 di dimensione 3.

```
2 . . . 1 . . 3 . 2
. . . . . . 3 . .
. . . . 2 2 . . .
. . . . . . 1 . .
. . 1 . . . . . .
. . . . . . . . .
. 3 . 2 . . . 1 . .
. 3 . 2 . 2 . . .
. 3 . . . 2 . . .
. . . . . . 3 . .
```

La stessa configurazione dopo 5 mosse in cui sono state colpite una nave lunga due e una lunga 3 e due colpi sono andati a vuoto ('X')

```
2 . . . 1 . . 3 . 2
. . . . . X c . .
. . . . 2 b X X . .
. . . . . . 1 . .
```

```

. . 1 . . . . .
. . . . .
. 3 . 2 . . . 1 .
. 3 . 2 . 2 . . .
. 3 . . . 2 . . .
. . . . . 3 . .

```

Rispetto al primo assegnamento ci sono delle importanti novità riguardo alla rappresentazione dell'area di gioco:

- la matrice non è più globale ma viene creata dinamicamente
- più matrici possono essere attive contemporaneamente (es. quella che utilizza il computer e la nostra durante la partita)
- non è più possibile posizionare navi adiacenti nell'area di gioco
- l'area di gioco di ogni giocatore viene rappresentata da una struttura con 4 campi: la matrice stessa, il numero di righe e colonne e il numero di navi ancora non affondate. questo ultimo valore va tenuto aggiornato durante la partita e utilizzato per decretare la fine del gioco e la vittoria di uno dei due giocatori.

### 3 Come si svolge la partita

Durante una partita vengono allocate due matrici (una per il computer e una per l'umano) e posizionato un numero uguali di navi delle diverse lunghezze per i due avversari, a seconda dell'input della funzione `gioca()`. Una volta posizionate le navi, il gioco procede a turni. Il giocatore di turno *spara un colpo* dichiarando un quadretto (per esempio, "1-5"). L'avversario controlla sulla propria griglia se quella cella è occupata da una sua nave. In caso affermativo risponde *colpito!*, e marca quel quadretto sulla propria tabella; in caso negativo risponde *acqua*. Quando un colpo centra l'ultimo quadretto di una nave non ancora affondata, il giocatore che subisce il colpo dovrà dichiarare "colpito e affondato!", e la nave si considera persa. A quel punto il turno passa all'altro giocatore. Vince il giocatore che per primo affonda tutte le navi dell'avversario.<sup>1</sup> Si può assumere che:

- Le navi di entrambi i giocatori vengano piazzate in modo casuale.
- L'umano sia il primo a giocare.
- Il computer scelga la mossa in modo casuale.

Funzionalità aggiuntive (come una strategia migliore per il computer, o piazzamento a scelta delle navi del giocatore umano) sono chiaramente ammesse, fintanto che vengono implementate tutte le funzionalità richieste dalla versione di base e superati i test 1, 2 e 3.

### Cosa deve essere realizzato

Lo studente deve realizzare le funzioni per allocare, deallocare e inizializzare le strutture e le matrici che rappresentano le aree di gioco. Inoltre deve aggiornare le funzioni `put*` per il posizionamento delle navi sviluppate nel primo assegnamento in modo da farle lavorare correttamente con la nuova segnatura e la nuova rappresentazione delle matrici. Ricordiamo che tali funzioni devono anche essere aggiornate per soddisfare il NUOVO vincolo di non contiguità delle navi. Inoltre dovranno essere realizzate: la funzione

<sup>1</sup>Wikipedia. [https://it.wikipedia.org/wiki/Battaglia\\_navale\\_\(gioco\)](https://it.wikipedia.org/wiki/Battaglia_navale_(gioco))

`esegui_mossa()` che verifica l'esito di un colpo su uno specifico quadretto e la funzione `gioca()` che gestisce un'intera partita di gioco tra umano e computer.

I prototipi delle funzioni da realizzare si trovano nel file `bnavale.h`. Nel file sono presenti le macro che definiscono i valori delle caselle in caso rappresentino acqua, navi o navi colpite o affondate.

I file `*test.c` contengono dei `main` che usano queste funzioni ed effettuano dei test sul loro funzionamento. Tali test possono essere attivati automaticamente utilizzando il `Makefile` come specificato nel file `README`. Solo il codice che supera con successo questi test può essere consegnato.

*Tuttavia, è bene ricordare che il superamento dei test non garantisce la correttezza completa della soluzione, quindi invitiamo gli studenti ad analizzare attentamente i risultati ottenuti e le stampe effettuate prima della consegna.*

Il codice deve essere adeguatamente commentato spiegando le idee e strategie adottate (senza esagerare). Opzionalmente, lo studente può realizzare parti aggiuntive e consegnare una breve discussione su quanto realizzato e come utilizzarlo sotto forma di file PDF.