

# Nim Game

Eriberto Momentè

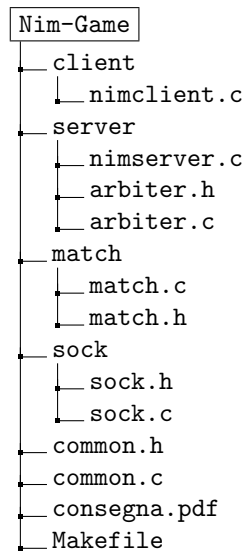
Laboratorio di Sistemi Operativi  
A.S. 2016/2017

## Consegna

La consegna originale del progetto la si può trovare nella directory radice del progetto sotto il nome di **consegna.pdf**

## Organizzazione del progetto

In questa sezione verrà presentato il file system del progetto, una breve descrizione di esso ed eventuali strutture utilizzate.



## **common.h e common.c**

In questi due file vengono definite stringhe di testo che compariranno nei vari messaggi stampati dal programma, variabili globali e la funzione **check** che controlla l'output di una funziona e in caso di errore stampa un messaggio e termina l'esecuzione.

## **sock**

Vengono definite funzioni per lo scambio di messaggi attraverso socket.

## **match**

Contiene la struttura **match** che rappresenta una partita di Nim. Essa consiste in due interi che corrispondono agli elementi rimasti nelle due pile e **turn** che indica a quale giocatore spetta il turno in un dato istante (0 per il primo, 1 per il secondo). **match** contiene in **match.c** funzioni d'utilità per gestire e manipolare la struttura omonima.

## **client**

Contiene il programma **nimclient** che implementa la sua parte del protocollo di messaggi con il server.

## **server**

Contiene il programma **nimserver** che accetta due clients. Dopodichè, come da consegna, lancia su un thread separato (l'arbitro) la gestione della partita tra quei due clients e ritorna in attesa di due nuovi clients.

**arbiter.c** contiene appunto la funzione che viene eseguita dal thread arbitro parallelo al server. L'arbitro ha sia la funzione di implementare la parte server del protocollo di messaggi con i clients, sia di aggiornare lo stato corrente della partita di Nim.

## **Protocollo di messaggi tra client e server**

Finchè un match non è terminato:

- All'inizio di ogni nuovo turno l'arbitro invia al client di dovere il messaggio **YOURTURN**.

- Successivamente, l'arbitro invia sempre a quel client un'istanza di **match** che rappresenta la partita in quel dato istante.
- A questo punto il client fa scegliere all'utente da quale delle due pile vuole rimuovere elementi. Raccoglie l'input da **stdin**, verifica che sia consistente e lo spedisce al server.
- In un secondo momento, il client fa la stessa cosa con il numero di elementi che l'utente intende rimuovere.

Quando una partita termina, l'arbitro comunica ai due clients chi è il vincitore e termina.

## Ulteriori informazioni sul progetto

### Come compilare

Il progetto si compila con il comando **make**.

### Sviluppo & Test

Il progetto è stato sviluppato in ambiente MacOS nella distribuzione HighSierra. È stato testato e compilato senza errori o segnalazioni con **clang**.