

**Sistemi Operativi**  
prova di laboratorio  
– 20 febbraio 2024 –

Creare un programma **bingo-game.c** in linguaggio C che accetti invocazioni sulla riga di comando del tipo:

**bingo-game <n> <m>**

Il programma dovrà gestire una partita al gioco del Bingo tra  $n$  giocatori, ognuno in possesso di  $m$  card. Il gioco del Bingo prevede un dealer che estrae i numeri (da 1 a 75) e card sono composte da 3 righe di 5 numeri (tutti distinti all'interno della card). I premi in palio sono solo due: la cinquina (una riga da 5 numeri estratti) e il Bingo (una card completa).

Il programma una volta avviato, creerà  $1+n$  thread: un thread dealer **D** e  $n$  thread giocatori **P1**, **P2**, ... , **Pn**. Tutti i thread si dovranno coordinare usando semafori generici e, se necessario, dei mutex.

L'unica struttura dati condivisa tra i thread dovrà contenere, oltre agli strumenti di coordinamento (semafori, mutex), una insieme di valori in quantità non dipendente da  $n$  o  $m$ , quindi costante  $O(1)$ . Ad esempio: UNA scheda di  $3 \times 5$  numeri, ultimo numero estratto, ... L'uso di variabili globali è fortemente scoraggiato.

In una prima fase il dealer dovrà generare casualmente, una alla volta, le  $(n \times m)$  card dei giocatori: ogni card generata verrà passata al giocatore assegnatario usando la struttura condivisa. Terminata la distribuzione delle card si passerà all'estrazione dei numeri. Per ogni numero estratto (senza duplicati rispetto alle precedenti estrazioni) questo sarà notificato a tutti i giocatori attraverso la struttura dati condivisa e gli strumenti di sincronizzazione. Se un giocatore rileva di aver raggiunto uno dei due premi (cinquina o Bingo) ancora disponibili, lo notifica al dealer attraverso la struttura condivisa e gli strumenti di sincronizzazione, inviandogli copia completa della scheda. La cinquina può essere vinta solo dal primo che la ottiene con una data estrazione e, vinto il Bingo, la partita termina.

Per semplicità, non ci sono "ex aequo": in caso di vincitori multipli sulla stessa estrazione, il dealer considererà vincitore solo il primo di cui avrà notifica.

Terminato il gioco, i thread dovranno terminare correttamente e spontaneamente.

Suggerimenti:

- usare `rand()` per campionare un numero pseudo-casuale e, una tantum, `srand(time(NULL))` da `time.h` per ottenere sequenze di numeri di volta in volta diversi;
- il dealer, dopo aver notificato ai giocatori una estrazione, prima di procedere con la prossima estrazione dovrebbe aspettare una "conferma" da parte di ogni giocatore in cui questo conferma di aver controllato le proprie card e, eventualmente, di aver vinto qualche premio.

**Tempo:** 2 ore e 45 minuti

L'output tipo di una esecuzione dovrebbe essere il seguente:

```
$ ./bingo-game 3 2

D: ci saranno 3 giocatori con 2 card ciascuno
D: genero e distribuisco la card n.1: (49,23,6,3,74) / (45,2,9,44,1) / (4,64,17,33,55)
P1: ricevuta card (49,23,6,3,74) / (45,2,9,44,1) / (4,64,17,33,55)
D: genero e distribuisco la card n.2: (12,53,28,9,18) / (4,45,17,1,2) / (75,71,44,63,5)
P2: ricevuta card (12,53,28,9,18) / (4,45,17,1,2) / (75,71,44,63,5)
...
D: fine della distribuzione delle card e inizio di estrazione dei numeri
D: estrazione del prossimo numero: 12
D: estrazione del prossimo numero: 74
D: estrazione del prossimo numero: 28
...
D: estrazione del prossimo numero: 44
P2: card con cinquina: (12,53,28,9,18) / (4,45,17,1,2) / (75,71,44,63,5)
D: il giocatore n.2 ha vinto la cinquina con la scheda (12,53,28,9,18) /
(4,45,17,1,2) / (75,71,44,63,5)
...
D: estrazione del prossimo numero: 34
P3: card con Bingo: (23,12,43,34,19) / (4,13,1,75,74) / (41,32,70,17,66)
D: il giocatore n.3 ha vinto il Bingo con la scheda (23,12,43,34,19) / (4,13,1,75,74) /
(41,32,70,17,66)
D: fine del gioco
```