



Sistemi Operativi

Modulo di Laboratorio 18E



Esercizio 01: regole di gioco

- Scrivere in C con l'utilizzo delle ncurses e pthread un **programma multithread** che implementi il gioco “**guardia, ladro e avvocato**”
- Il gioco si compone di **tre oggetti sullo schermo**:
 - La **guardia** (carattere **#**), controllata dall'**utente** tramite le **frecce direzionali**, inizia la partita dall'**angolo inferiore destro** dello schermo
 - Il **ladro** (carattere **\$**), controllato dalla **logica di gioco** e che si muove in **maniera casuale**, inizia la partita dall'**angolo superiore sinistro** dello schermo
 - L'**avvocato** (carattere **A**), controllato dalla **logica di gioco** e che si muove anch'esso in **maniera casuale**, inizia la partita dall'**angolo superiore destro** dello schermo
- Il **gioco termina in due condizioni** distinte e dagli esiti diversi:
 - Con **vittoria** del giocatore se la **guardia #** “**cattura**” il **ladro \$** prima dell'avvocato A
 - Con **sconfitta** del giocatore se l'**avvocato A** “**cattura**” il **ladro \$** prima della guardia #

Esercizio 01: dettagli implementativi

- Il programma richiesto deve far uso di **4 thread distinti**:
 - Il thread principale **riceve le comunicazioni** sulle nuove posizioni degli oggetti di gioco, **stampa gli oggetti** stessi, **verifica le collisioni** e **controlla la terminazione**
 - A fine gioco stampa un messaggio adeguato per il giocatore (“vittoria”/“sconfitta”)
 - Un primo thread creato **acquisisce l’input utente e gestisce la posizione della guardia**, comunicandola al padre ogni volta che la aggiorna
 - Un secondo thread creato **gestisce la posizione del ladro, generando uno spostamento casuale** e comunicando la posizione aggiornata al padre
 - Un terzo thread creato **gestisce la posizione dell’avvocato, generando uno spostamento casuale** e comunicando la posizione aggiornata al padre
- Definire delle **funzioni dedicate per la logica di tutti i thread**
- Utilizzare un **buffer con semaforo per le comunicazioni** verso il thread principale
- Gli eventi di gioco devono essere adeguatamente **temporizzati**

Esercizio 01: pseudocodice

```
#include <stdio.h>
#include <curses.h>
#include <stdlib.h>
#include <unistd.h>
#include <pthread.h>

int main() {
    Inizializzo_buffer_e_semafori();
    Creo_thread(Guardia);
    Creo_thread(Ladro);
    Creo_thread(Avvocato);
    Funzione_Controllo();
    Termino_thread();
    Stampo_messaggio(TipoCollisione);
    Exit();
}

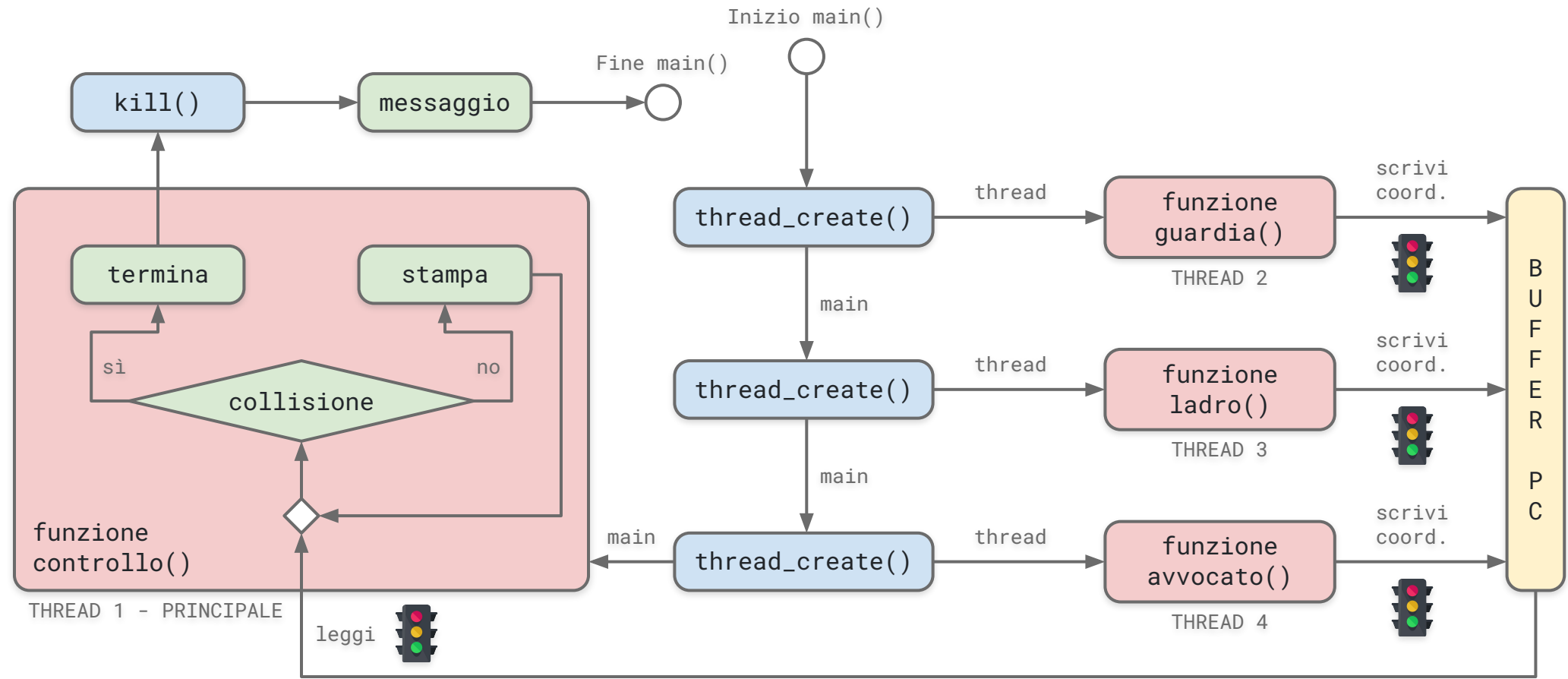
Funzione_Controllo() {
    while(!Collisione) {
        Leggo_coordinate_dal_buffer();
        Rilevo_tipo_oggetto();
        Pulisco_posizione_precedente_oggetto();
        Stampo_nuova_posizione_oggetto();
        if(Collisione)
            TipoCollisione = Definisci_collisione();
    }
}
```

```
Guardia() {
    while(1) {
        Leggo_tasti_cursore();
        Aggiorno_coordinate_oggetto();
        Comunico_coordinate_con_buffer();
    }
}

Ladro() {
    while(1) {
        Genero_movimento_casuale();
        Aggiorno_coordinate_oggetto();
        Comunico_coordinate_con_buffer();
        Sleep(Temporizzazione);
    }
}

Avvocato() {
    while(1) {
        Genero_movimento_casuale();
        Aggiorno_coordinate_oggetto();
        Comunico_coordinate_con_buffer();
        Sleep(Temporizzazione);
    }
}
```

Esercizio 01: diagramma di flusso



Fine Modulo 18E

