# Programmazione e Architetture (Modulo B)

## Lezione 17
## Esercizi sui thread

Luca Manzoni

# Variabili condizione

## Attendere che si verifichi un evento

- Una cosa che dobbiamo fare spesso nella programmazione multithread è di attendere che si verifichi un cambiamento
- Per esempio abbiamo un thread che “processa” dei dati in arrivo…
- …ma non deve fare niente fino a quando non ci sono dati!
- Una possibilità sarebbe quella di controllare continuamente se ci sono nuovi dati
- Un approccio migliore è quello di fermarsi e attendere che qualcuno segnali che sono arrivati nuovi dati

# Variabili condizione

## Attendere che si verifichi un evento

- Un approccio utilizzato è quello delle variabili condizione (*condition variables*)
- Un thread può decidere di attendere ad una variabile condizione
- E.g. “La coda è vuota. Attendo fino a quando qualcuno non mi avvisa che sono stati inseriti nuovi valori”
- Un altro thread può decidere di “svegliare” gli altri thread che attendevano
- E.g. “Ho inserito un nuovo valore nella coda, posso avvisare i thread in attesa di questo cambiamento”

# Variabili condizione

## Coi pthread

- Nella libraria pthread una variabile condizione è di tipo `pthread_cond_t`

- Le variabili condizioni devono essere inizializzate e per quello è possibile usare la macro `PTHREAD_COND_INITIALIZER`

- Per attendere si utilizza
  ```
  pthread_cond_wait(pthread_cond_t *c,
                    pthread_mutex_t *m);
  ```
  dove `c` è la variabile condizione e `m` è un mutex che al momento si detiene

- Per segnalare si utilizza invece
  ```
  pthread_cond_signal(pthread_cond_t *c);
  ```

# Variabili condizione

## Attesa

- Supponiamo di volere segnalare quando un lavoro è stato fatto cambiando il valore della variabile “done” e di avere una variabile condizione `c` e un mutex `m`

- Nell’attesa noi faremo:

```
pthread_mutex_lock(&m);
while (done != 1) {
    pthread_cond_wait(&c, &m);
} // do stuffs here
pthread_mutex_unlock(&m);
```

- Quello che succede è che prendiamo il lock m, se il valore di done non è 1 attendiamo un segnale (su c), rilasciando m

- Quando arriva un segnale riprendiamo il lock m e controlliamo che done sia 1 (qualcuno potrebbe aver cambiato il valore nel frattempo)

# Variabili condizione

## Segnalare

- Per segnalare diventa più semplice:

```
pthread_mutex_lock(&m);
done = 1;
pthread_cond_signal(&c);
pthread_mutex_unlock(&m);
```

- Questo “sveglia” i thread che erano in attesa su quella variabile condizione

- Notate che è importante che quando si attende su una variabile condizione il controllo sia dentro un ciclo while, dato che è possibile che più thread siano in attesa e la variabile sia stata modificata prima che il lock sia ri-acquisito

# Produttore-consumatore

## Una struttura comune