Corso di Sistemi Operativi e Reti

Prova scritta telematica 30 LUGLIO 2020

ESERCIZIO 1 - PROGRAMMAZIONE MULTITHREAD

ISTRUZIONI

1. **Questo file contiene il testo che ti è stato dato ieri, incluso il codice;**
2. **Mantieni a tutto schermo** questo file per tutta la durata della prova; puoi scorrere liberamente tra le sue pagine, ma non puoi cambiare applicazione;
3. **Firma** preliminarmente il foglio che userai per la consegna con nome cognome e matricola;
4. **Svolgi** il compito; puoi usare solo carta, penna e il tuo cervello;
5. **Alla scadenza** termina *immediatamente* di scrivere, e attendi di essere chiamato, pena l'esclusione dalla prova;
6. **Quando è il tuo turno** mostra il foglio ben visibile in webcam, e poi metti una foto dello stesso foglio in una chat privata Microsoft Teams con il prof.

Corso di Sistemi Operativi e Reti

Prova scritta telematica 30 LUGLIO 2020

ESERCIZIO 1, TURNO 1 - QUESITO DA RISOLVERE

Si illustri come modificare il metodo `puntaNumero` in maniera tale da fornire informazioni sull'esito della partita. La nuova versione di `puntaNumero` deve fare la puntata regolarmente. Quindi, anzichè terminare subito, si deve attendere l'esito della partita; infine va restituito un valore booleano che indichi se il thread che ha appena invocato `puntaNumero` risulta essere quello vincente.

Si scriva su carta il codice del metodo, e si indichi, aiutandosi con i numeri di riga, quali modifiche andrebbero apportate al codice pre-esistente.

MATERIALE DIDATTICO

Il codice fornito implementa una versione del gioco “BassoNumero”, che consente la presenza simultanea di N thread giocatori in parallelo.

*Nel gioco del “BassoNumero”, ogni giocatore punta segretamente (cioè senza essere a conoscenza delle puntate altrui) un **qualsiasi** numero intero a partire da 1. Una volta che tutti i giocatori hanno effettuato le proprie puntate, queste vengono esaminate. Vince il giocatore che ha puntato il numero più basso che non sia stato puntato da nessun altro giocatore. Ad esempio, supponiamo di avere una partita in cui i giocatori G1, G2, G3, G4 e G5 effettuano rispettivamente le puntate:*

G1 : 1
G2: 1
G3: 2
G4: 3
G5: 13

In questo caso i numeri puntati da un solo giocatore sono 2, 3 e 13, mentre il numero 1 è stato puntato da due giocatori. Il giocatore vincente è G3, poichè “2” è la puntata unica più bassa.

La classe **`NumeroBasso`** è preposta allo scopo di gestire una o più partite. La classe contiene il metodo **`puntaNumero(num : int)`** che consente a un certo thread/giocatore di puntare il numero voluto, e del metodo **`gioca(int N) -> Thread,`** che avvia e gestisce una partita tra N giocatori simulati, restituendo infine il Thread ID (TID) del thread vincitore.

```
01: from threading import Thread, RLock, Condition, current_thread
02: from random import randint
03:
04: class Player(Thread):
05:
06:     def __init__(self,nb):
07:         super().__init__()
08:         self.nb = nb
09:
10:     def run(self):
11:         self.nb.puntaNumero(randint(1,10))
12:
13: class NumeroBasso:
14:
15:
16:     def __init__(self):
17:         self.giocate = []
18:         self.lock = RLock()
19:         self.threadGioca = Condition(self.lock)
20:         self.partitaInCorso = False
21:         self.nGiocate = 0
22:         self.barrier = None
23:
24:     def gioca(self,N : int) -> int:
25:         with self.lock:
26:             self.giocate = {}
27:             self.nGiocate = 0
28:             self.partitaInCorso = True
29:             for _ in range(0,N):
30:                 Player(self).start()
31:             while(self.nGiocate < N):
32:                 self.threadGioca.wait()
33:             self.partitaInCorso = False
34:             for k in sorted(self.giocate):
35:                 if len(self.giocate[k]) == 1:
36:                     print(f"Il vincitore è il thread {self.giocate[k][0]} che ha puntato il numero {k}")
37:                     return self.giocate[k][0]
```

```
38:             print("Non ci sono vincitori")
39:             return 0
40:
41:     def puntaNumero(self,n : int):
42:         with self.lock:
43:             self.giocate.setdefault(n,[]).append(current_thread().ident)
44:             self.nGiocate += 1
45:             self.threadGioca.notify()
46:
47: if __name__ == '__main__':
48:     gameManager = NumeroBasso()
49:     v = 1
50:     while v != 0:
51:         v = gameManager.gioca(10)
```