Una variabile intera `n`, inizializzata a 0, è condivisa tra 2 thread `tO`, `tE`.
Il thread `tE`, ciclicamente:

1. attende 200 ms (N.B.: la chiamata `usleep(t)` attende per `t` **microsecondi**)
2. genera un `int` casuale pari e lo somma alla variabile condivisa `n`
3. se ha eseguito almeno 10 cicli e `n` è pari termina
4. altrimenti ricomincia dal passo (1), a meno che abbia già compiuto 1000 iterazioni, nel qual caso termina.

Il thread `tO`, ciclicamente:

1. attende 200 ms (N.B.: la chiamata `usleep(n)` attende per `n` **microsecondi**)
2. genera un `int` casuale dispari e lo somma alla variabile condivisa `n`
3. se ha eseguito almeno 10 cicli e `n` è dispari termina
4. altrimenti ricomincia dal passo (1), a meno che abbia già compiuto 1000 iterazioni, nel qual caso termina.

(Non ricorrere a un array di 2 thread per l'implementazione!)

Il programma termina quando tutti i thread hanno terminato la propria esecuzione. I thread scriveranno di essere terminati. Possono anche visualizzare, a ogni ciclo, il valore trovato in `n`.
Nel codice, proteggere opportunamente la variabile `n` dagli accessi concorrenti.

Tempo a disposizione: 30 minuti.