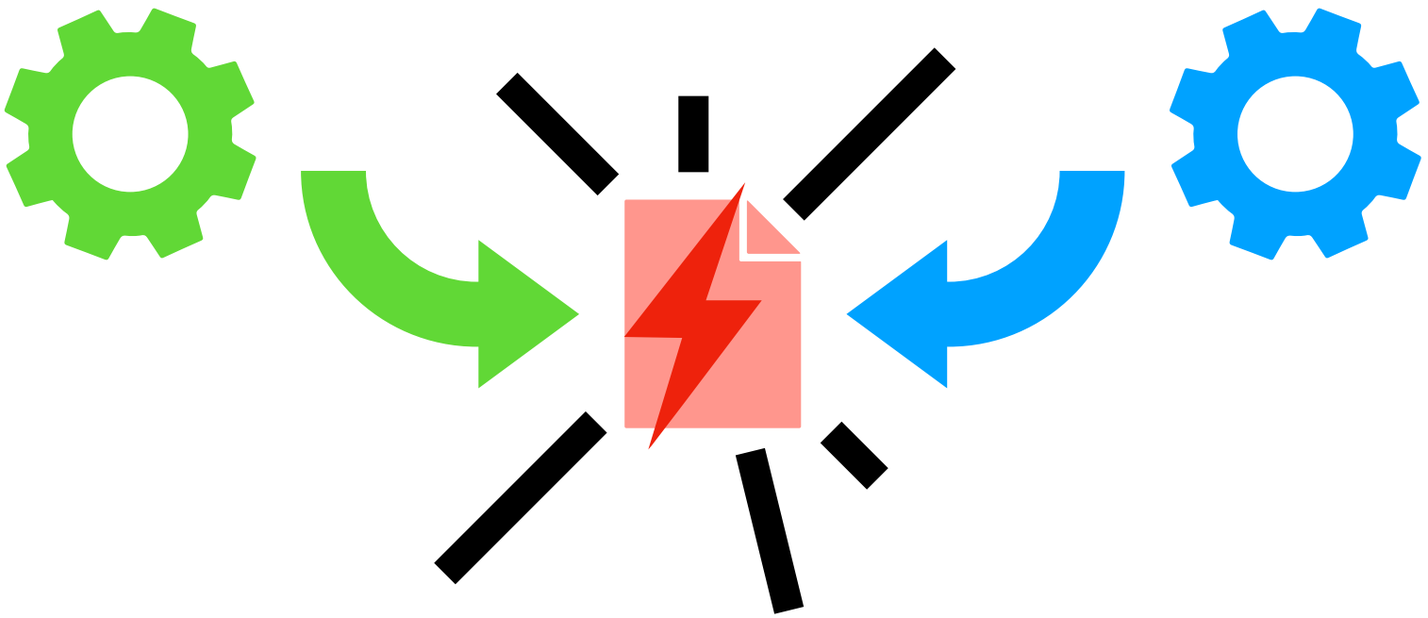
Linguaggi di Programmazione



Roberta Gori
Vincenzo Ciancia

Presentazione

Orario

Lunedi’: 16:00-18:00 Fib P1

Mercoledi’: 9:00-11:00 Fib N

Giovedi’: 14:00-16:00 Fib E1

Venerdi’: 14:00-16 Aula G

I prof

Roberta Gori
Dipartimento di Informatica
roberta.gori@unipi.it

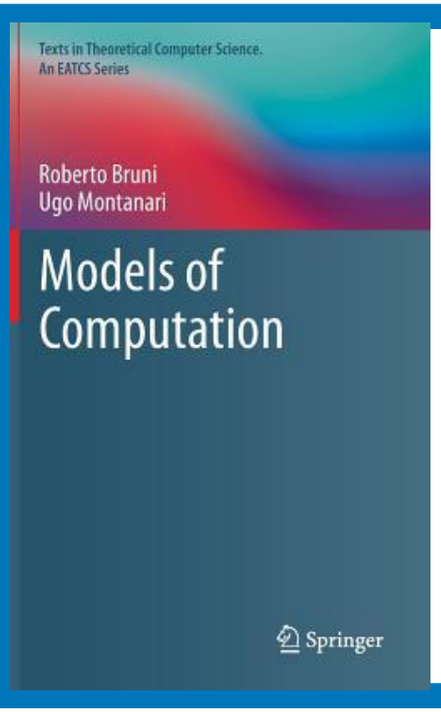
Vincenzo Ciancia
Istituto di Scienza e Tecnologie
dell'Informazione "Alessandro Faedo"
vincenzo.ciancia@isti.cnr.it

Obiettivo del corso

Iniziare a ragionare formalmente su cosa calcolano i programmi sui loro effetti

- vi presentero' diverse modelli di calcolo:
  - linguaggi imperativi,
  - linguaggi funzionali di ordine superiore,
  - linguaggi concorrenti
- vedremo i loro paradigmi di programmazione,
- e come possiamo descrivere il loro comportamento,
- ci focalizzeremo su alcuni strumenti/tecniche per ragionare sui programmi nei diversi paradigmi.

Il libro che seguiremo



Roberto Bruni and Ugo Montanari
Models of Computation
Texts in Theoretical Computer Science (an EATCS series)
https://www.springer.com/book/9783319428987

Informazioni utili

Canale teams per comunicazioni urgenti : 063AA 24/25 - LINGUAGGI DI PROGRAMMAZIONE CON LABORATORIO [MAT-L] | General | Microsoft Teams

Pagina web per materiale didattico:
http://didawiki.di.unipi.it/doku.php/matematica/lp/start

Modalita' d'esame

- Progetto diviso in due con date di consegna + Compitini

- Progetto intero+ Prova scritta (obbligatoria)

- Prova orale (facoltativa)

Cercate di partecipare!

cerchiamo di trovare insieme le risposte

```
%trovare il piu' piccolo divisore proprio p(1<p<n)di
n>0
p := 0;
x := 2;
while ( x?≤?n/?2 ) do {
    if ( n%x == 0 ) then {
        p := x;
    } else {
        x := x+1;
    }
}
```

Cercate di partecipare!

Correggetemi se sbaglio

```
% trovare l'indice dell'ultima occorrenza di n in
nell'array a
i := length(a)-1;
while ( i>0 && n!=a[i] ) do {
    i := i-1;
}
```

Prerequisiti

Teoria degli insiemi

$\varnothing$ $\quad A \cap B \quad A \cup B \quad A \setminus B \quad \overline{A}$

$a \in A \quad A \subset B \quad A \subseteq B \quad A \times B$

$a \notin A \quad A \not\subseteq B \quad A \cap B = \emptyset$

$\mathbb{N} \quad \mathbb{Z} \quad \mathbb{Q} \quad \mathbb{R} \quad \mathbb{B}$

$N \subseteq \mathbb{N} \quad N \in \wp(\mathbb{N}) \quad S \subseteq \wp(\mathbb{N})$

Prerequisiti

teoria degli insiemi: funzioni, relazioni

$$f : A \to B$$

$$R \subset A \times B$$

funzioni come relazioni

$$R_f \triangleq \{(a, f(a)) \mid a \in A\}$$

insiemi come funzioni
Dato un insieme N

$$f_N : \mathbb{N} \to \mathbb{B}$$

$$f_N(n) \triangleq \begin{cases} 1 & n \in N \\ 0 & \text{otherwise} \end{cases}$$

$$N = \{n \mid f_N(n) = 1\}$$

Prerequisiti

Logica del primo ordine

ff false 0 F tt true 1 T $P \wedge Q$ $P \vee Q$ $\neg P$

$\exists x.\ P(x)$ $\forall x.\ P(x)$ $P \Rightarrow Q$ $P \Leftrightarrow Q$

significato dell'implicazione!

$$P \Rightarrow Q$$

$$Q \vee \neg P$$

$$\neg Q \Rightarrow \neg P$$

importanza dell'ordine dei quantificatori!

$$\forall n \in \mathbb{N}.\ \exists m \in \mathbb{N}.\ n < m$$

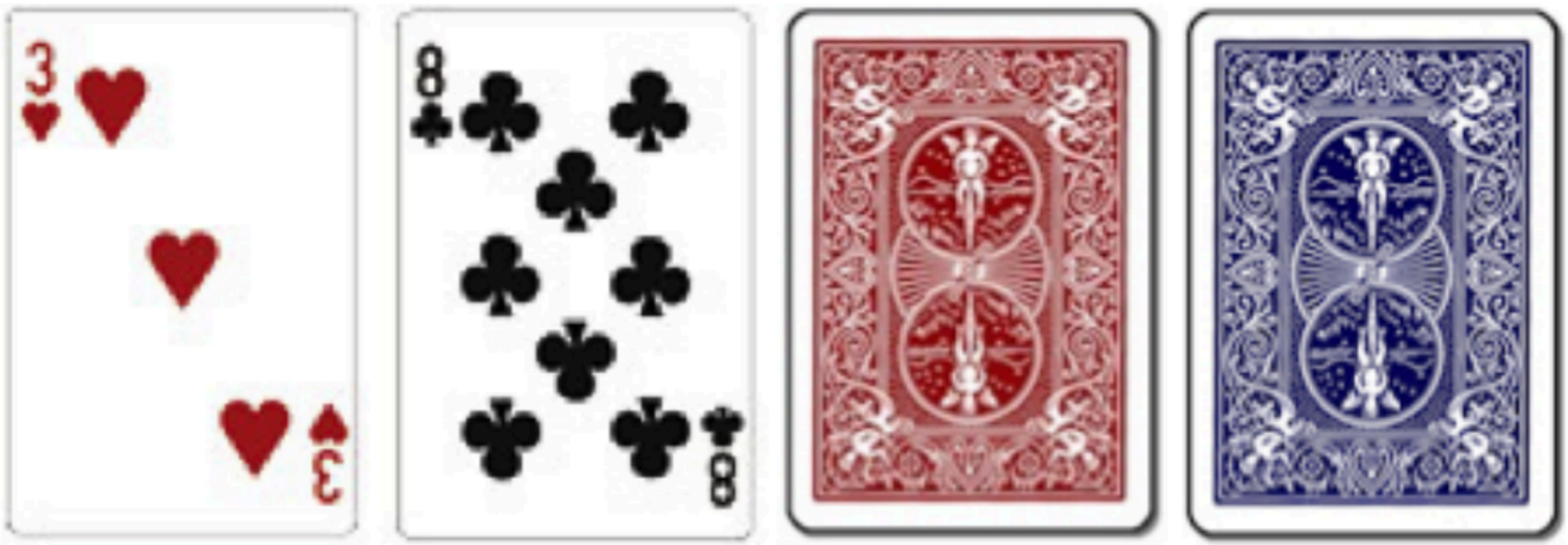
$$\exists m \in \mathbb{N}.\ \forall n \in \mathbb{N}.\ n < m$$

L'implicazione e' un concetto che richiede attenzione

In quali circostanze questa promessa viene infranta?

"Se Rob Bery vince le elezioni, prometto di lasciare il paese."

Wason Selection Task (4 carte)



Quali carte devono essere girate per assicurarsi che la seguente affermazione sia vera?

"Se la faccia anteriore di una carta mostra un numero pari, allora la sua faccia posteriore è rossa."

Wason Selection Task (con birre)

Un poliziotto entra in un locale in Florida dove c'è un cartello che dice:
"Per bere birra devi avere più di 16 anni."

Ci sono quattro clienti:

1. Un ragazzo che beve acqua.
2. Una ragazza che beve birra.
3. Una signora anziana che beve al tavolo.
4. Un ragazzo di 15 anni che beve al bancone.

Chi deve controllare il poliziotto?

Prerequisiti

Definizioni ricorsive e induttive

$$\begin{aligned} 0! &\triangleq 1 \\ (n+1)! &\triangleq n! \cdot (n+1) \end{aligned}$$

$$\begin{aligned} A^0 &\triangleq \{\epsilon\} \\ A^{(n+1)} &\triangleq A \times A^n \end{aligned}$$

$$f(n) \triangleq \begin{cases} 1 & \text{if } n \leq 1 \\ f(n/2) & \text{if } n \neq 0 \wedge n\%2 = 0 \\ f(3n+1) & \text{otherwise} \end{cases}$$

Congettura di Collatz

$$f(12) = f(6) = f(3) = f(10) = f(5) = f(16) = f(8) = f(4) = f(2) = f(1) = 1$$

Prerequisiti

Congetture vs teoremi

un numero naturale p è primo
se non può essere scritto come il prodotto di due numeri più piccoli

n	Is n prime?	$2^n - 1$	Is $2^n - 1$ prime?
2	yes	3	yes
3	yes	7	yes
4	no: $4 = 2 \cdot 2$	15	no: $15 = 3 \cdot 5$
5	yes	31	yes
6	no: $6 = 2 \cdot 3$	63	no: $63 = 7 \cdot 9$
7	yes	127	yes
8	no: $8 = 2 \cdot 4$	255	no: $255 = 15 \cdot 17$
9	no: $9 = 3 \cdot 3$	511	no: $511 = 7 \cdot 73$
10	no: $10 = 2 \cdot 5$	1023	no: $1023 = 31 \cdot 33$

Prerequisiti

Congetture vs teoremi

if p is prime
then $2^p - 1$ is prime

if $n > 1$ is not prime
then $2^n - 1$ is not prime

Usate qualsiasi mezzo per provare o confutare le congetture di cui sopra

Prerequisiti

Stringhe su un alfabeto

Alfabeto A $\quad A^n \triangleq \underbrace{A \times \cdots \times A}_{n} \quad A^* \triangleq \bigcup_{n \in \mathbb{N}} A^n$

$$\mathbb{B} = \{\mathtt{0}, \mathtt{1}\}$$

$$\mathbb{B}^0 = \{\epsilon\}$$
$$\mathbb{B}^1 = \{\mathtt{0}, \mathtt{1}\}$$
$$\mathbb{B}^2 = \{00, 01, 10, 11\}$$

$$\mathbb{B}^* = \{\epsilon, \mathtt{0}, \mathtt{1}, \mathtt{00}, \mathtt{01}, \mathtt{10}, \mathtt{11}, \mathtt{000}, \ldots\}$$

Prerequisiti

Grammatiche

Un linguaggio e' un sottoinsieme di stringhe su un alfabeto.

Usiamo una grammatica per descrivere le stringhe legali nel linguaggio:

$$\text{SheepNoise} \rightarrow \text{SheepNoise}\ \underline{\text{baa}} \mid \underline{\text{baa}}$$

Definisce l'insieme dei rumori che una pecora emette in circostanze normali.

Prerequisiti

Linguaggio

$$\mathbb{B} = \{0, 1\} \qquad \mathbb{B}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$$

$$\begin{aligned} \mathsf{A} &::= \epsilon \mid 0\ \mathsf{A} \mid 1\ \mathsf{B} \\ \mathsf{B} &::= 0\ \mathsf{B} \mid 1\ \mathsf{A} \end{aligned}$$

$$\mathcal{L}(\mathbb{A}) = \{s \in \mathbb{B}^* \mid s \text{ contiene un numero pari di occorrenze di } 1\}$$

$$\mathsf{A} \to 0\ \mathsf{A} \to 0\ 1\ \mathsf{B} \to 0\ 1\ 1\ \mathsf{A} \to 0\ 1\ 1\ \epsilon = 0\ 1\ 1$$

Un antipasto

Cosa vediamo

In questo corso vediamo tre paradigmi diversi:

- Paradigma imperativo esemplificativo di linguaggi come C, C++, C#, Java, PHP, Python e Ruby

- Paradigma funzionale esemplificativo di linguaggi come ML, Miranda, OCAML,Haskel e F#

- Paradigma concorrente esemplificativo di linguaggi come Erlang, Scala, Go e Rust

Quali proprieta' valgono per questa funzione?

```
int function(int x)
  {k := 1;
     while k > 0 do
          { if r > 100 then
            r := r – 10;
            k := k – 1
          else
            r := r + 11;
            k := k + 1 }
return r}
```

$\forall x, function(x)\ termina??$

$\forall x, function(x) > 0??$

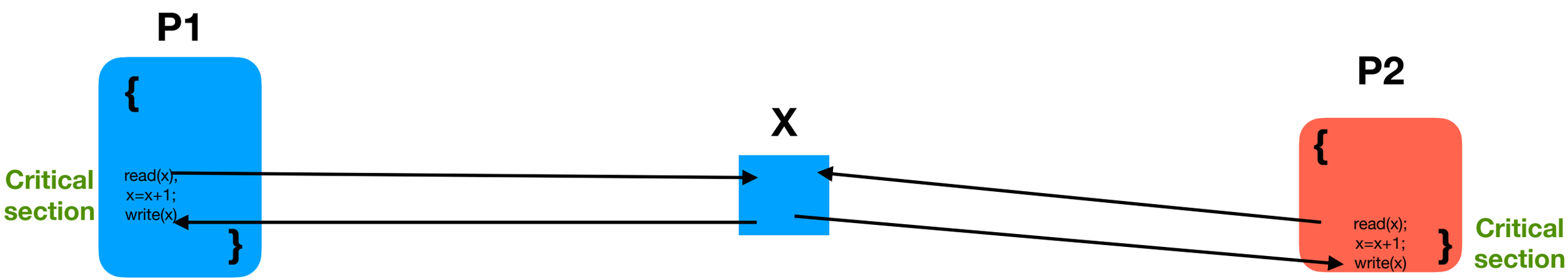
$\forall x, k > 0?$

$\forall x, k \geq 0?$

$\forall x, function(x) > 50??$

Come facciamo a provare queste proprietà??

Esempio: accesso in mutua esclusione



Due processi concorrenti condividono una risorsa che possono usare solo uno alla volta

Possono comunicare usando la memoria condivisa

Vogliamo garantire che non ci siano conflitti quando i processi accedono alla risorsa

Non vogliamo imporre una rigida alternanza di turni senza motivo

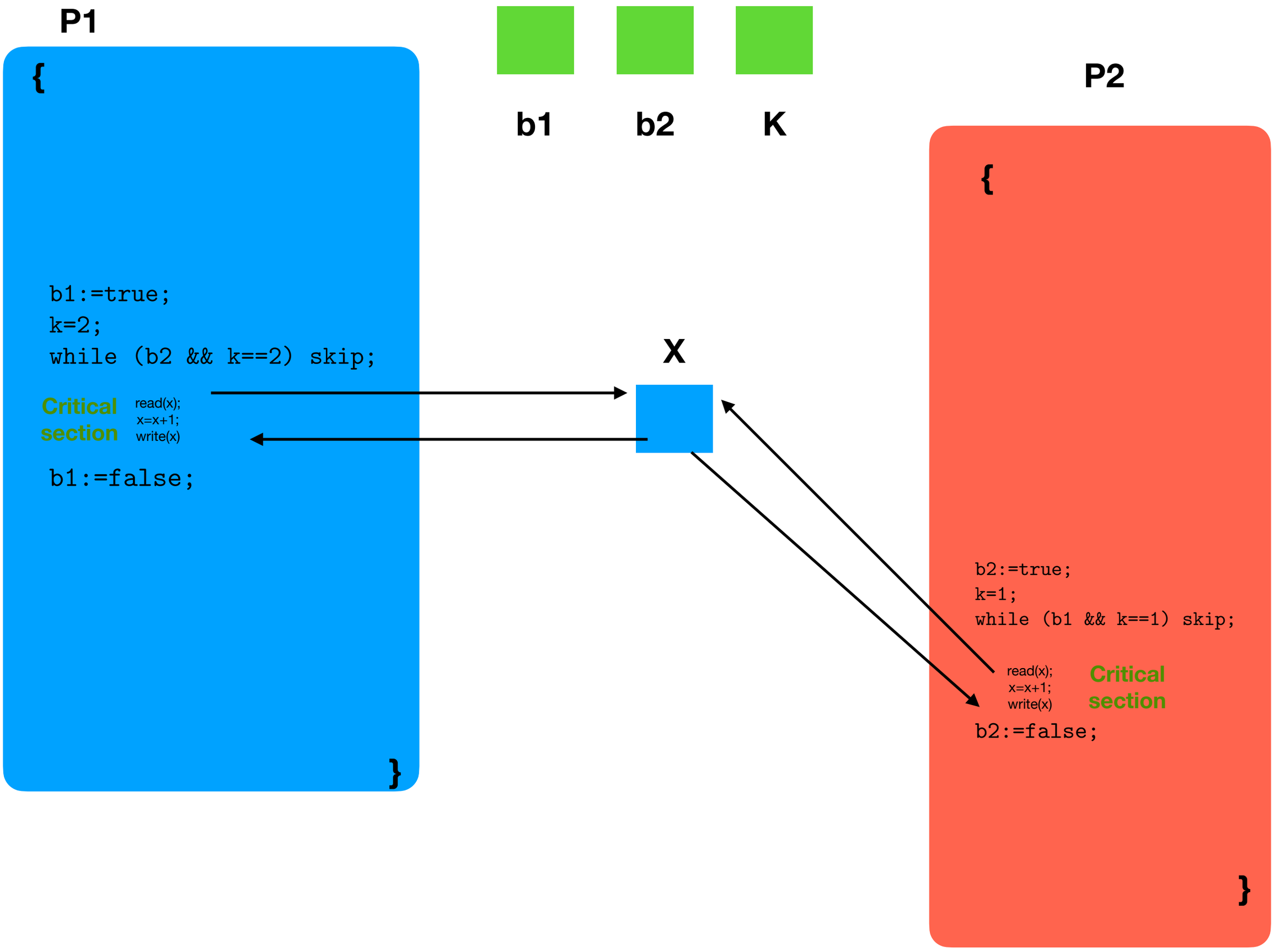
Algoritmo di mutua esclusione di Peterson (1981)

```
% Due processi P1, P2
% Due variabili booleane b1, b2 (valori true o false, all'inizio false)
% quando il processo Pi vuole accedere alla risorsa setta la variabile
% bi a true
% Una variabile k che avra' valori in {1,2}(inizialmente di valore arbitrario)
% ci dice che  Pk ha priorita' sull'altro processo
% Processo P1 in pseudocodice

 {
    …
    ...                          % fa altre cose prima
    b1 := true ;                 % P1 vuole accedere alla risorsa
    k := 2 ;                     % P1 da la priority all'altro processo
    while (b2 && k==2) skip ;    % P1 aspetta il suo turno
    <critical section>           % P1 entra in possesso della risorsa
    b1 := false                  % P1 rilascia la risorsa
       ...
       ...

 }

% P2 e' analogo: i ruoli di b1 e b2 sono scambiati, dove l'algoritmo setta b1
settera' invece  b2 e dove controlla b2 controllera' invece b1
 e  gli assegnamenti e controlli della variabile k coinvolgeranno il valore 1
invece del valore 2
```


P1
{
b1:=true;
k=2;
while (b2 && k==2) skip;
Critical section
read(x);
x=x+1;
write(x)
b1:=false;
}
b1
b2
K
X
P2
{
b2:=true;
k=1;
while (b1 && k==1) skip;
read(x);
x=x+1;
write(x)
Critical section
b2:=false;
}

Domande

L'algoritmo di Peterson funziona?

Cosa significa "funziona"? Cosa ci aspettiamo?

(Progresso)

Se la risorsa è disponibile, nessun processo è costretto ad aspettare

(Attesa limitata)

Nessun processo aspetterà per sempre la risorsa
(*altrimenti la soluzione più semplice è che nessuno entri*)

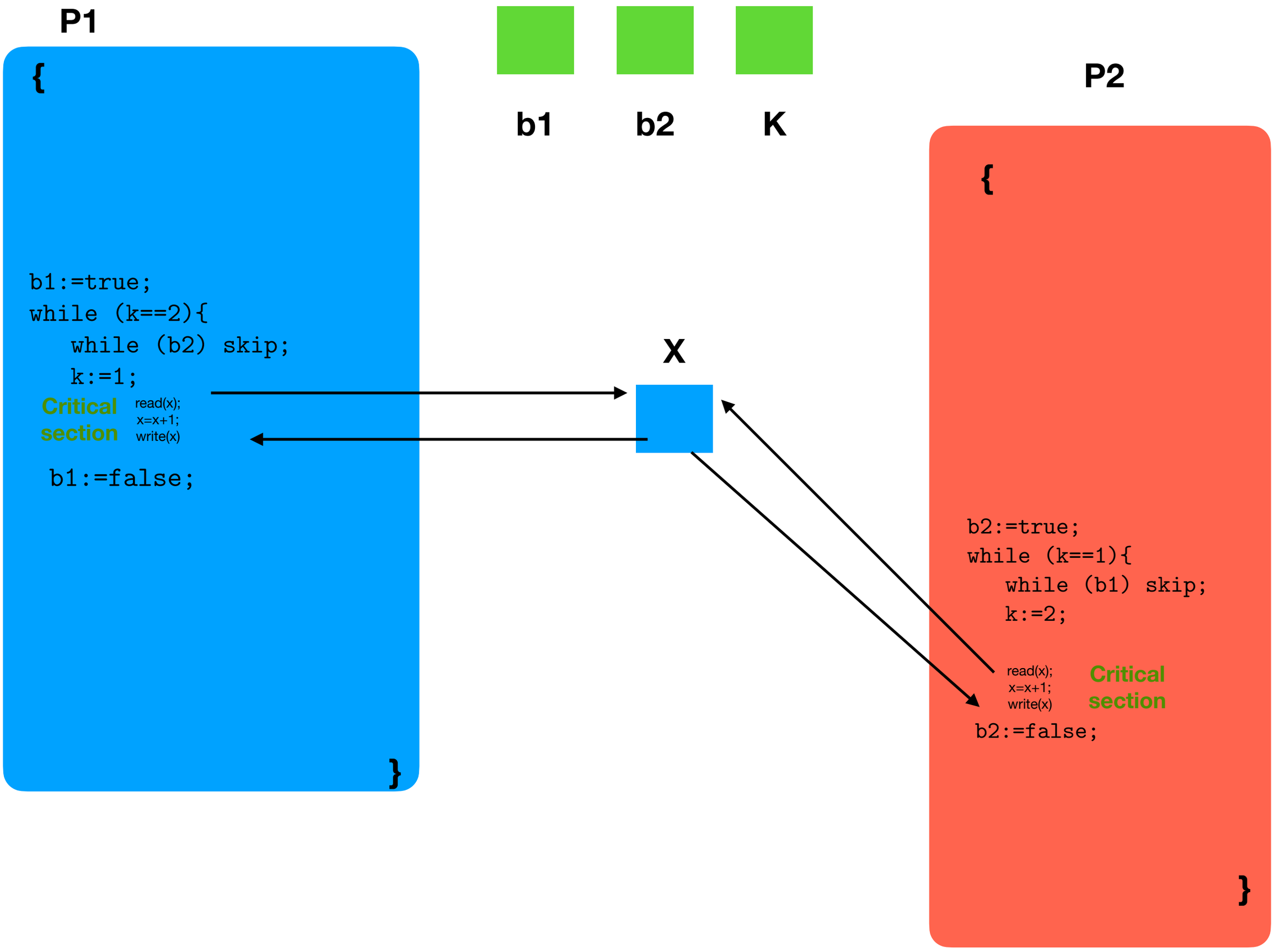
(Mutua Esclusione)

P1 e P2 non accedono alla risorsa allo stesso tempo

Algoritmo di mutua esclusione di Hyman (1966)

```
% Due processi H1, H2
% Due variabili booleane b1, b2 (valori true o false, all'inizio false)
% quando il processo Hi vuole accedere alla risorsa setta la variabile
% bi a true
% il processoHk ha priorita' sull'altro processo
%
% Processo H1 in pseudocodice
{
    ...                          % fa altre cose prima
    b1 := true ;                 % H1 vuole accedere alla risorsa
    while (k==2) {               % finche' H2 ha la priorita'
        while (b2) skip ;        % H1 aspetta
        k := 1;                  % H1 setta la priorita' per se stesso
   }
    <critical section>           % H1 accede alla risorsa
    b1 := false                  % H1 rilascia la risorsa
}

% H2 e' analogo: i ruoli di b1 e b2 sono scambiati, dove l'algoritmo
setta b1 settera' invece b2 e dove controlla b2 controllera' invece b1 e
analogamente il ruolo di 1 e 2 sono scambiati negli assegnamenti e
controlli della variabile k
```


P1
{
b1:=true;
while (k==2){
while (b2) skip;
k:=1;
Critical section
read(x);
x=x+1;
write(x)
b1:=false;
}
b1
b2
K
X
P2
{
b2:=true;
while (k==1){
while (b1) skip;
k:=2;
read(x);
x=x+1;
write(x)
Critical section
b2:=false;
}

La domanda

L'algoritmo di Peterson soddisfa la mutua esclusione?

L'algoritmo di Hyman soddisfa la mutua esclusione?

Per le risposte siate pazienti e aspettate le lezioni di fine corso