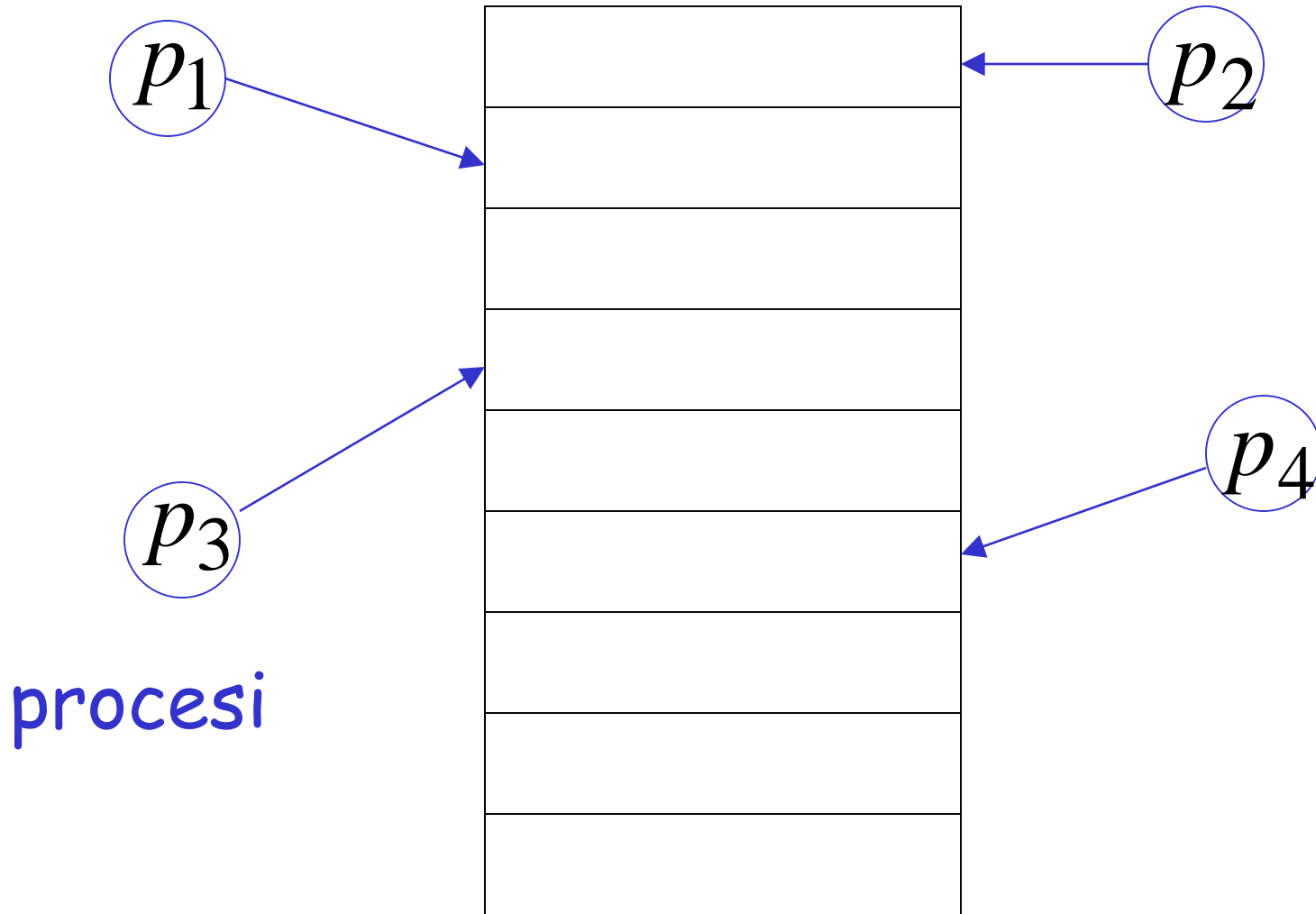


Deljena memorija

Deljena memorija



Tipovi deljenih promenljivih

- Read/Write
- Test & Set
- Read-Modify-Write

Read/Write promenljive

Read(v)

return(v);

Write(v,a)

v = a;

Ili jednostavnije:

x = v; (read v)

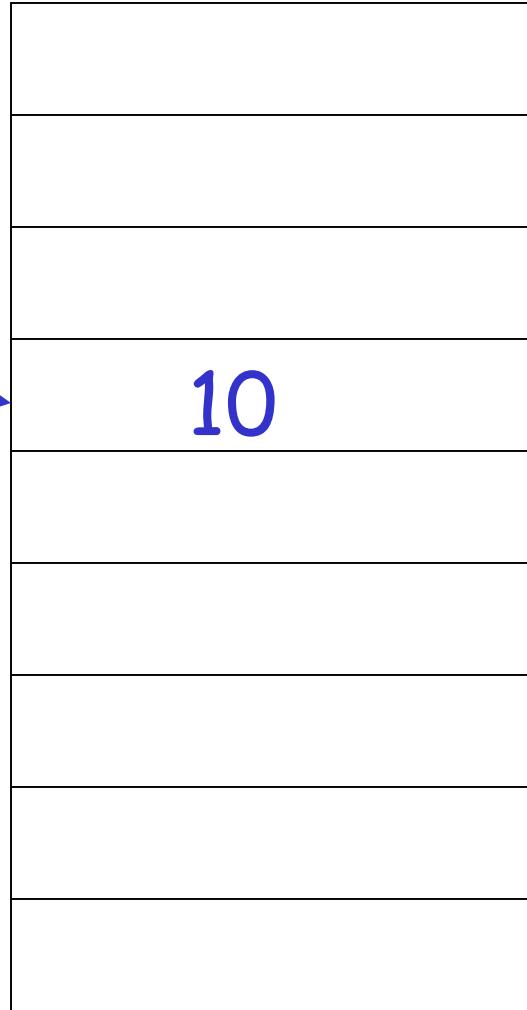
v = x; (write v)

p_1 write 10

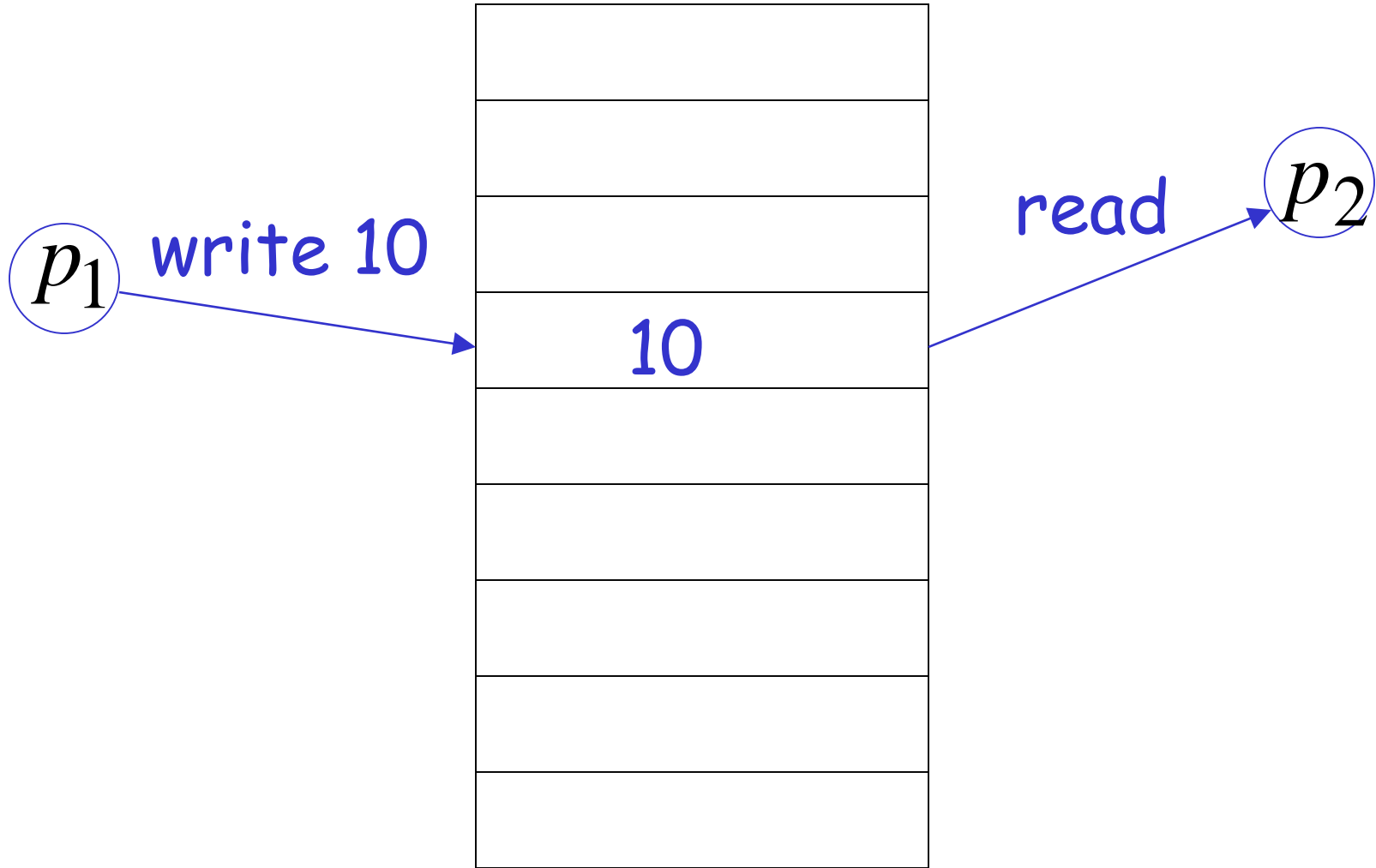


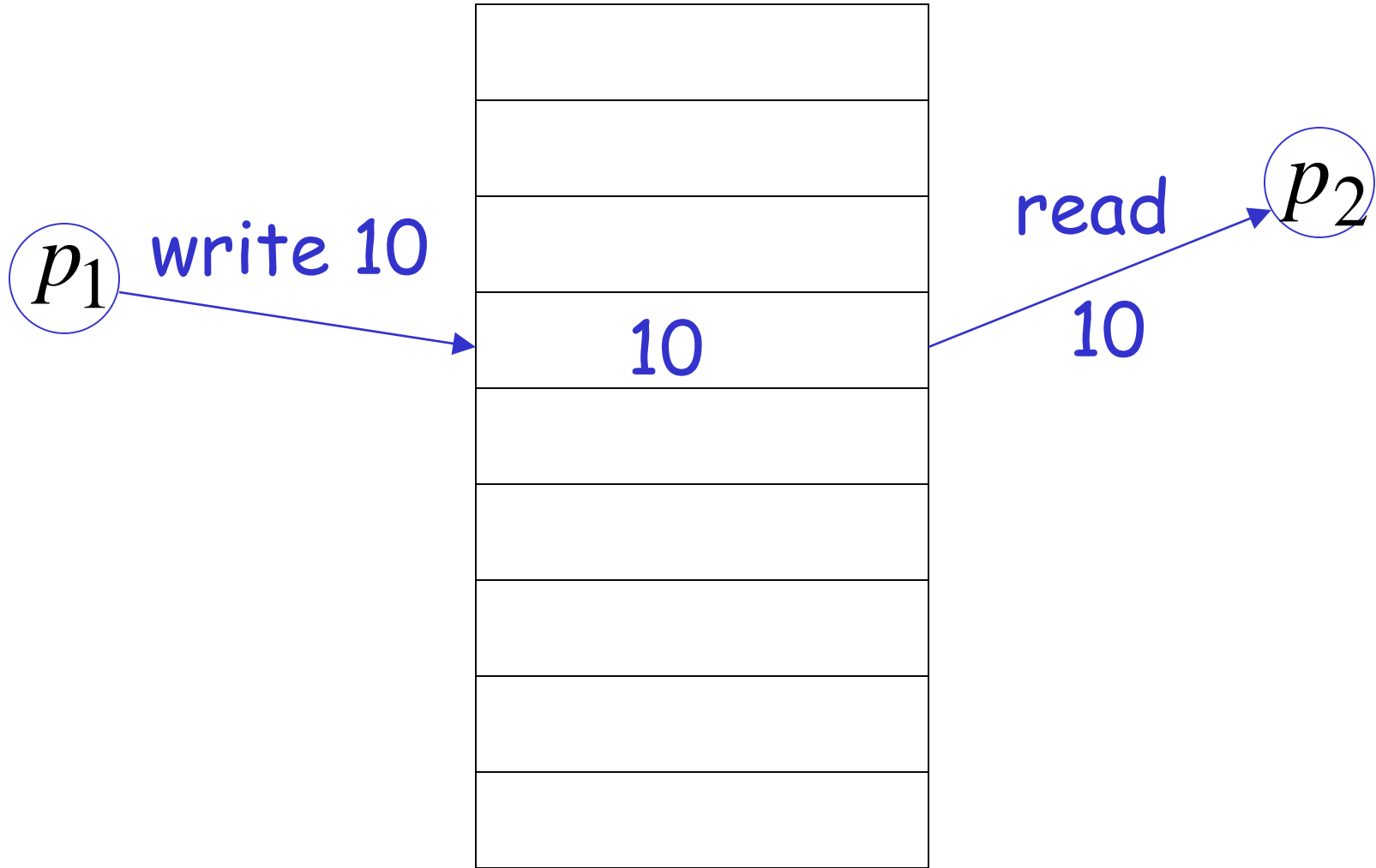
A diagram showing a process p_1 writing to a memory stack. A blue arrow points from the text p_1 to the fourth cell of a vertical stack of ten empty rectangular cells.

p_1 write 10

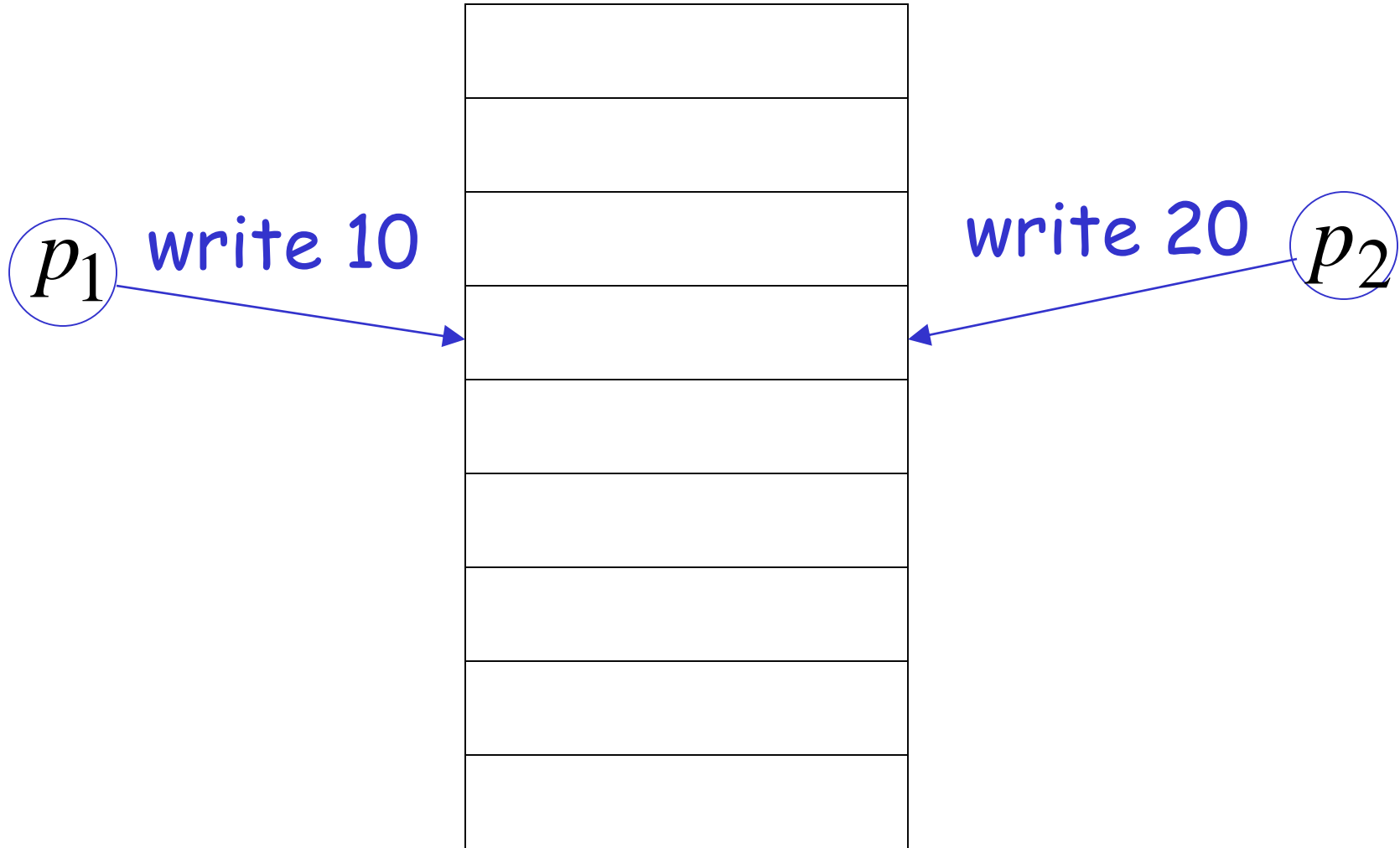


10

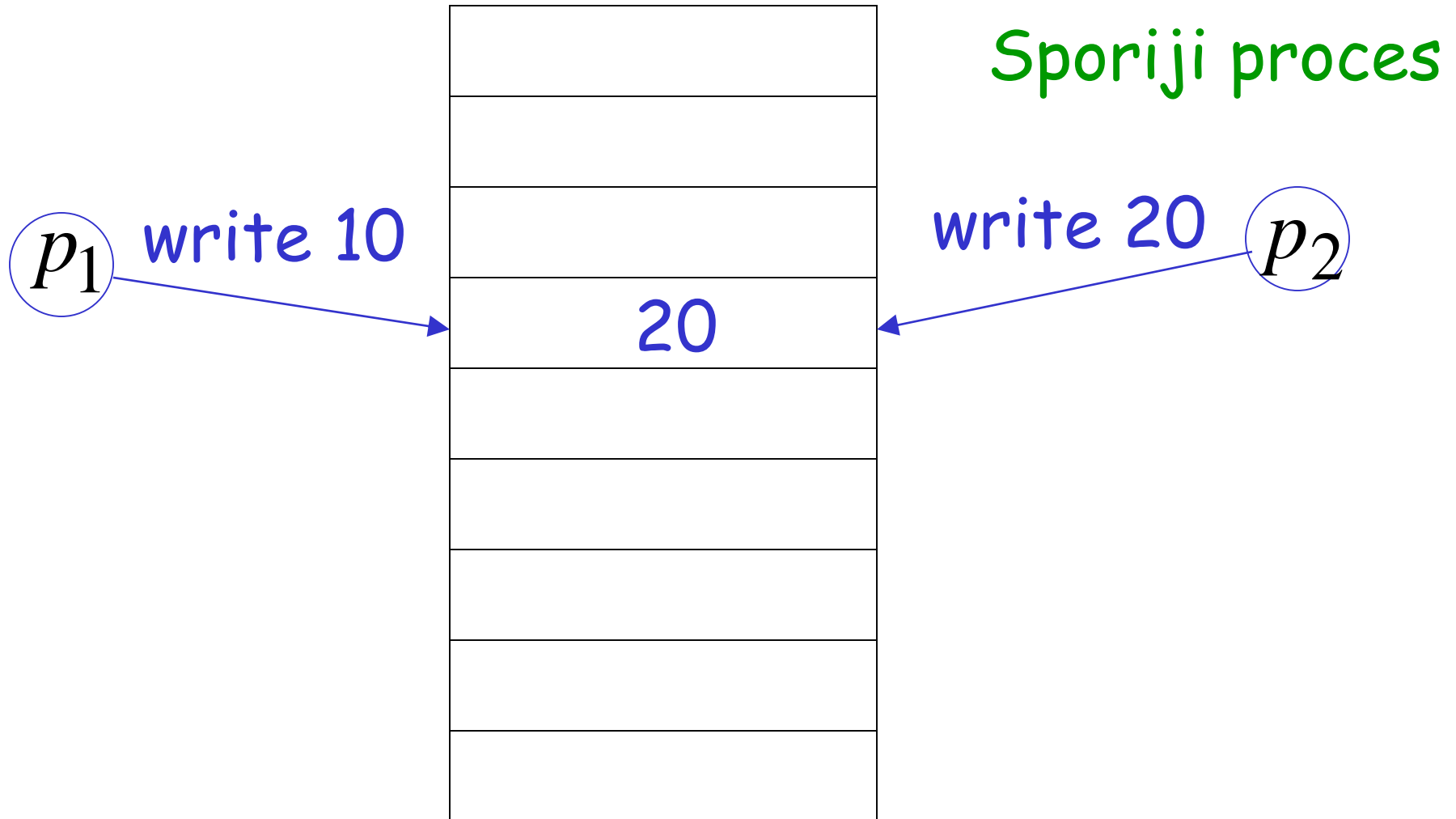




Simultani upisi



Simultani upisi Mogućnost 1

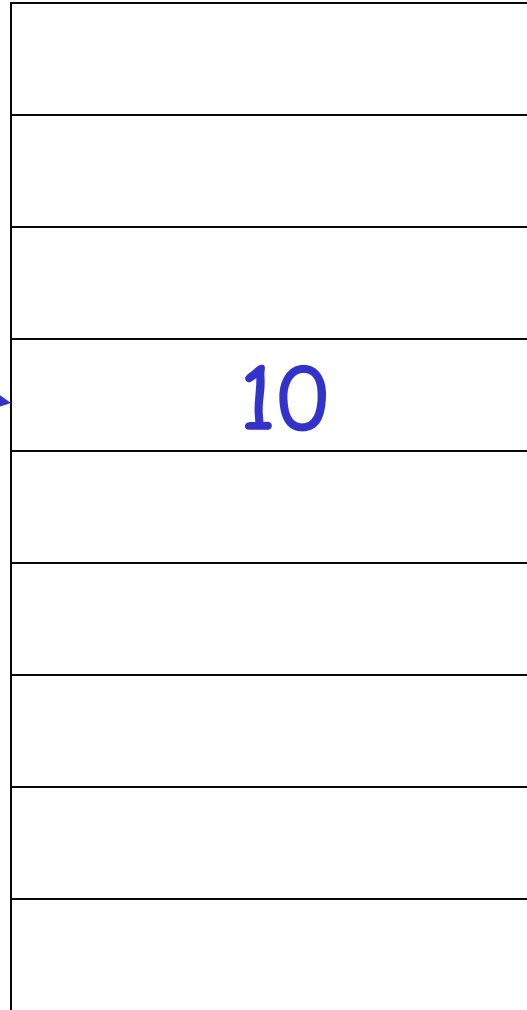


Simultani upisi Mogućnost 2

Sporiji proces

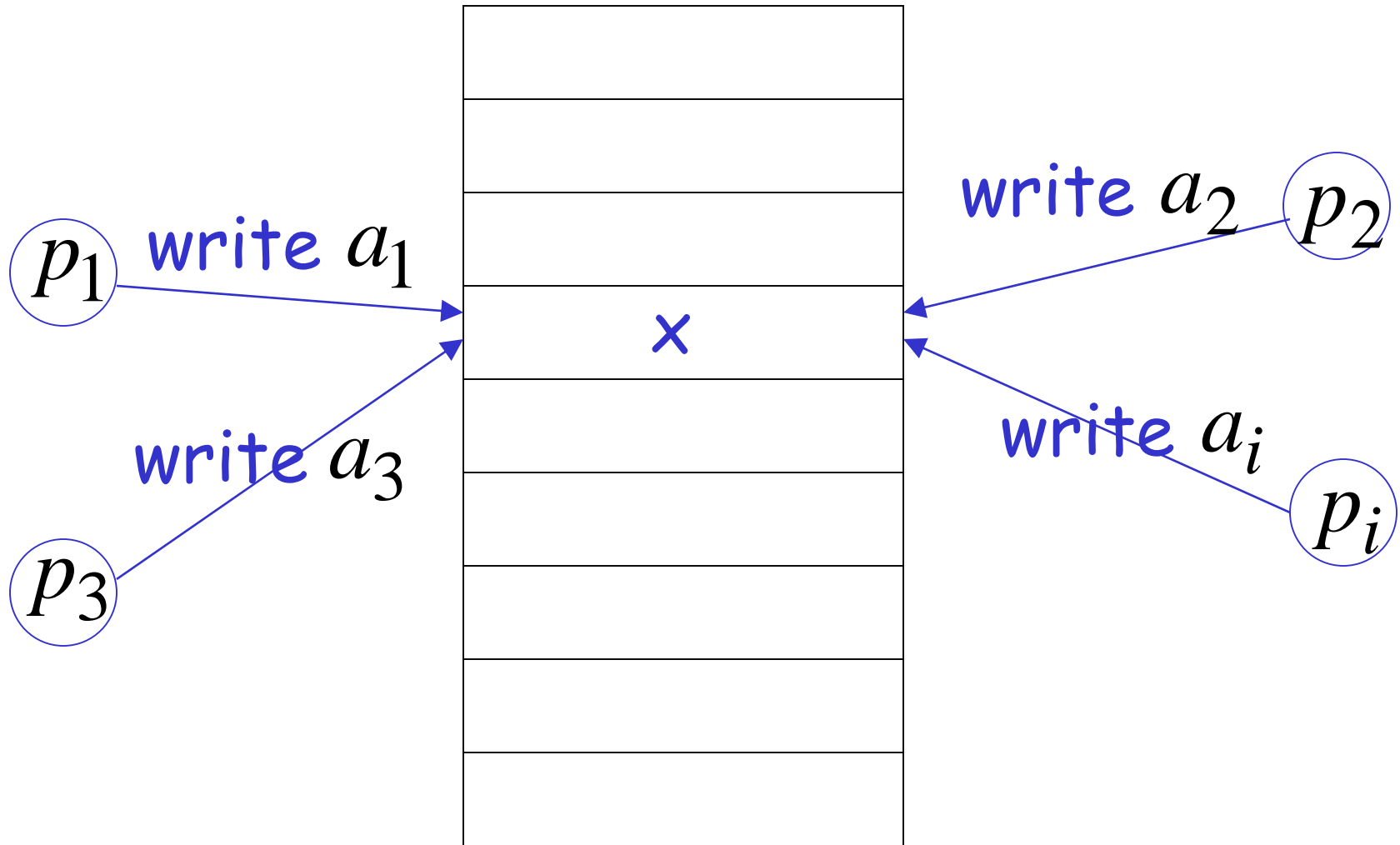
p_1 write 10

write 20 p_2

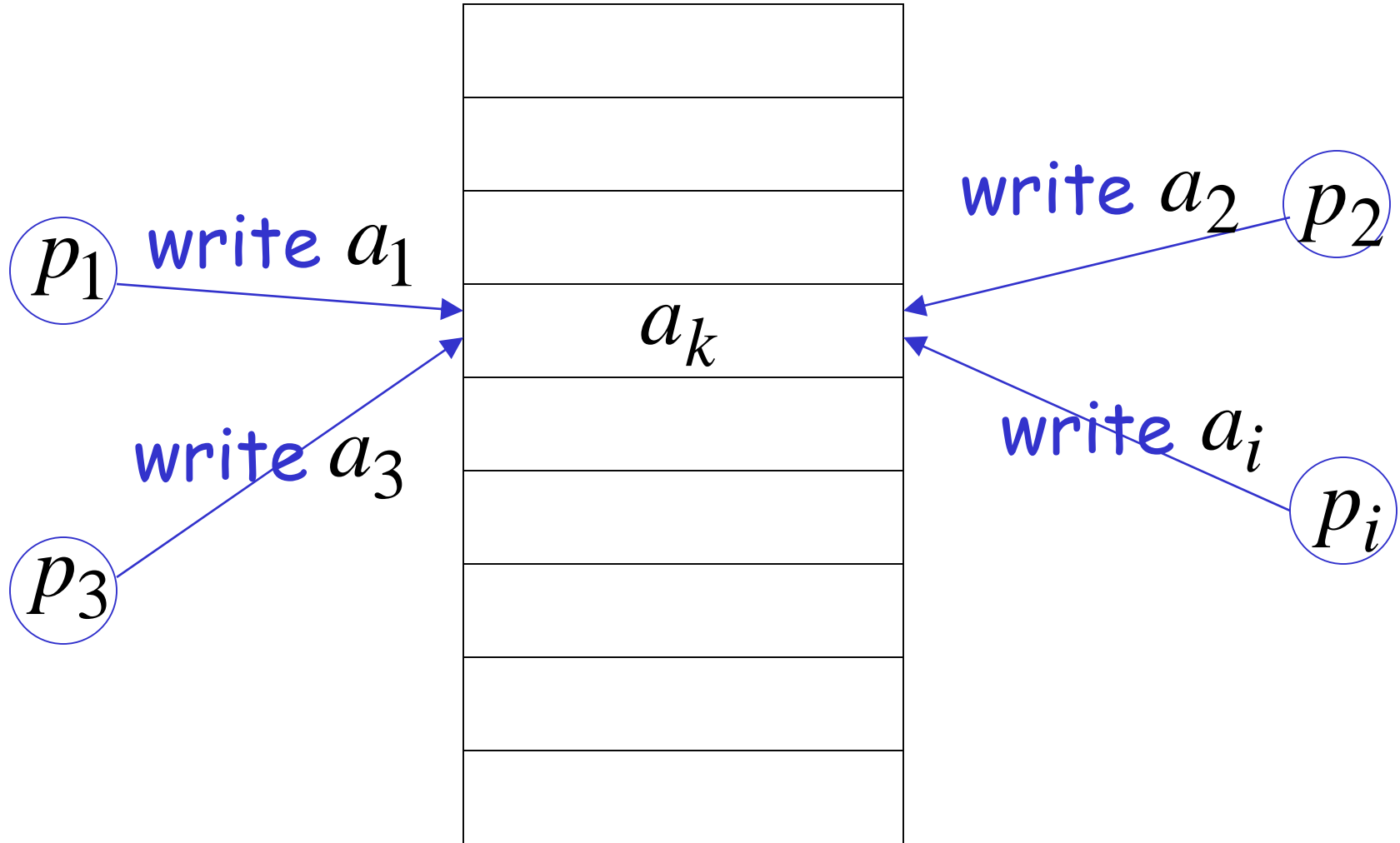


Simultani upisi

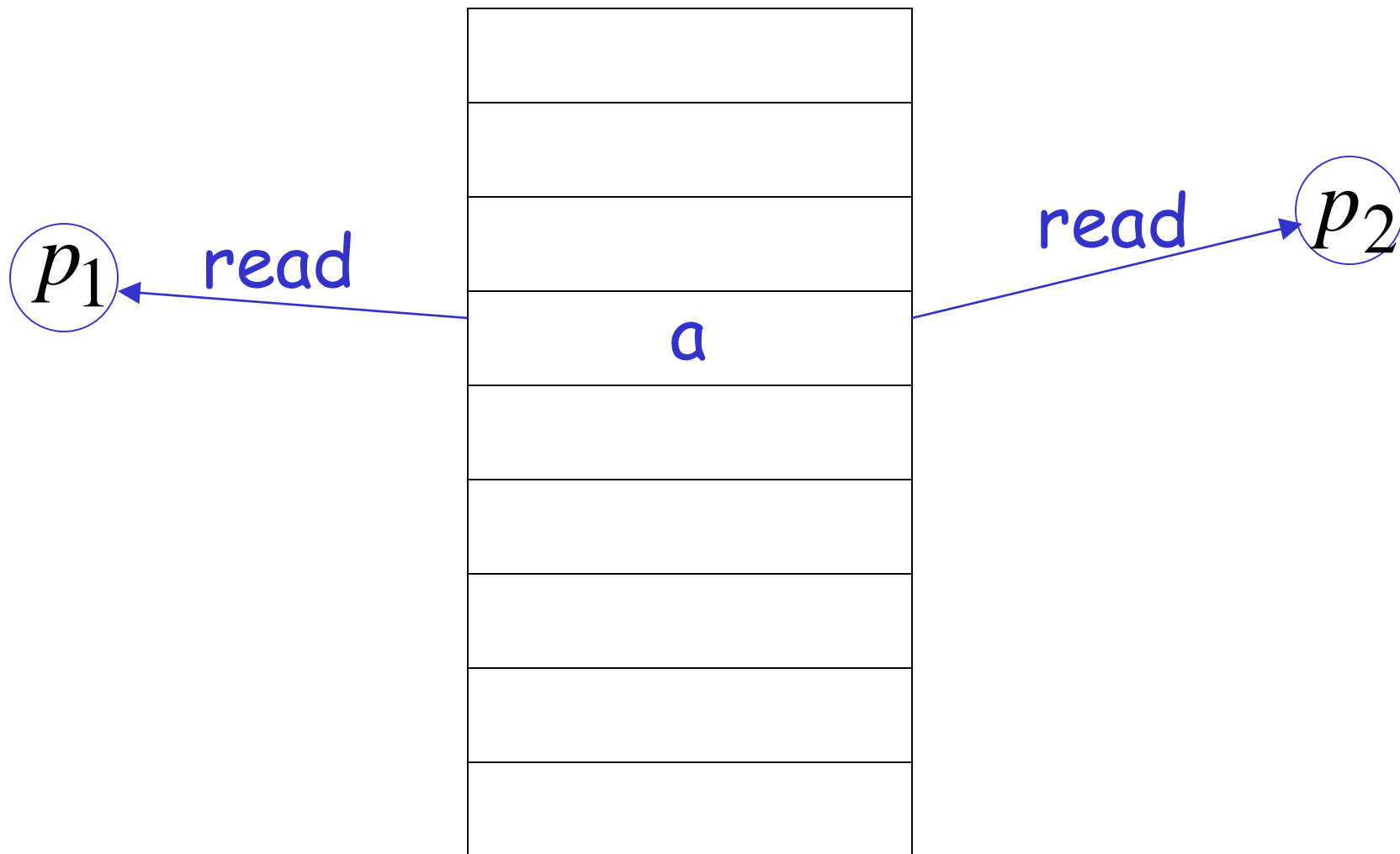
Opšti slučaj:



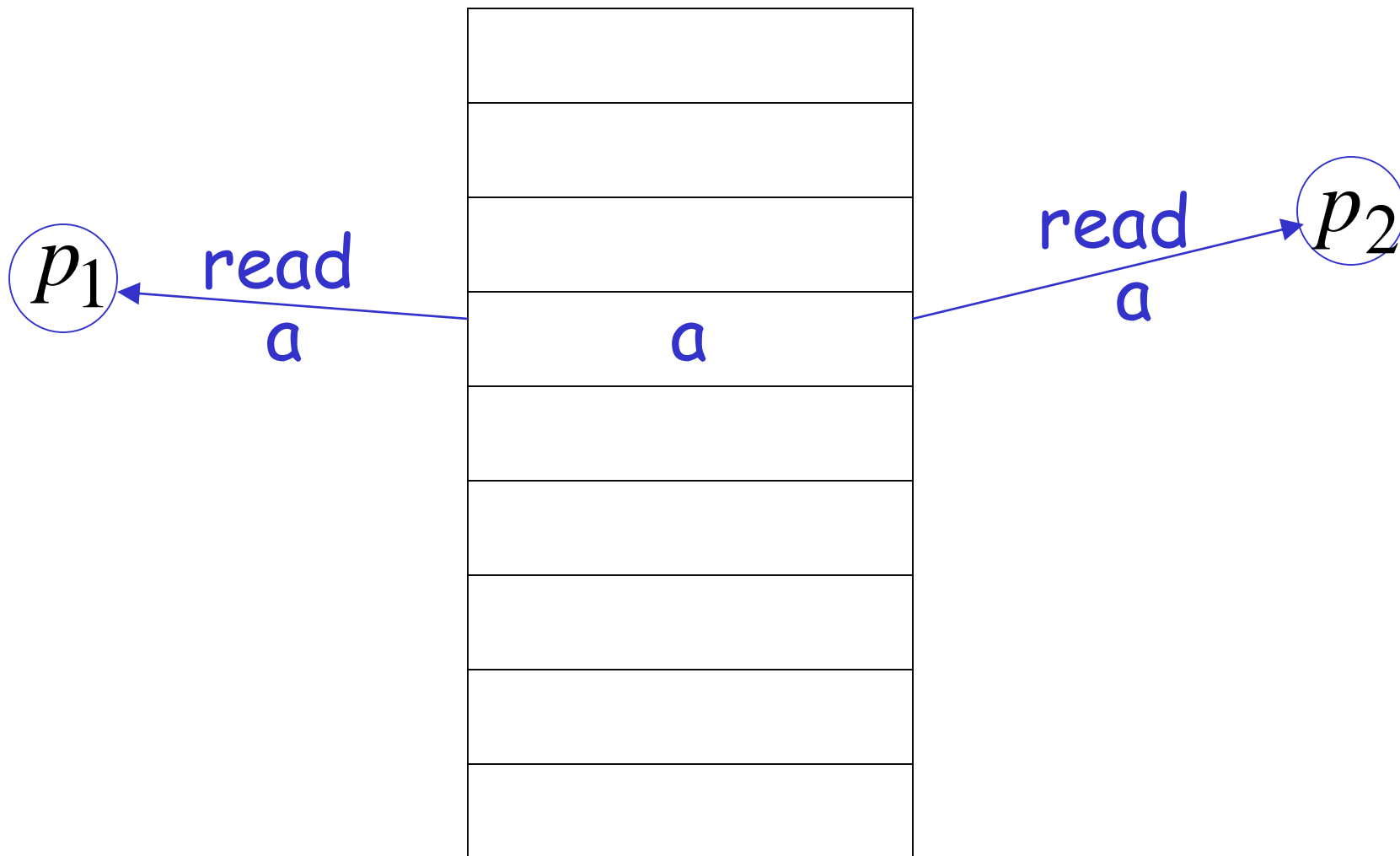
Simultani upisi Sporiji proces: p_k



Simultana čitanja



Simultana čitanja



Svi čitaju istu vrednost

Test&Set promenljive

Test&Set(v)

temp = v;

v = 1;

return (temp);

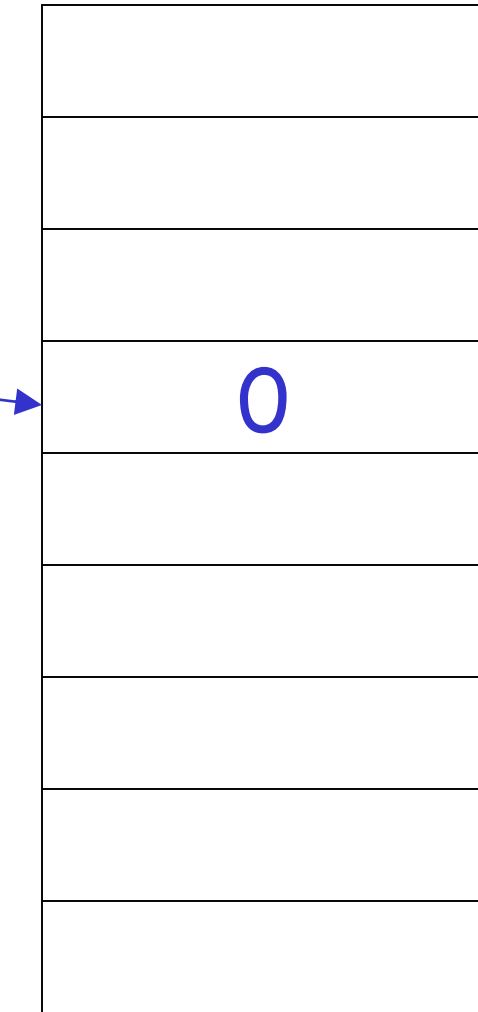
Reset(v)

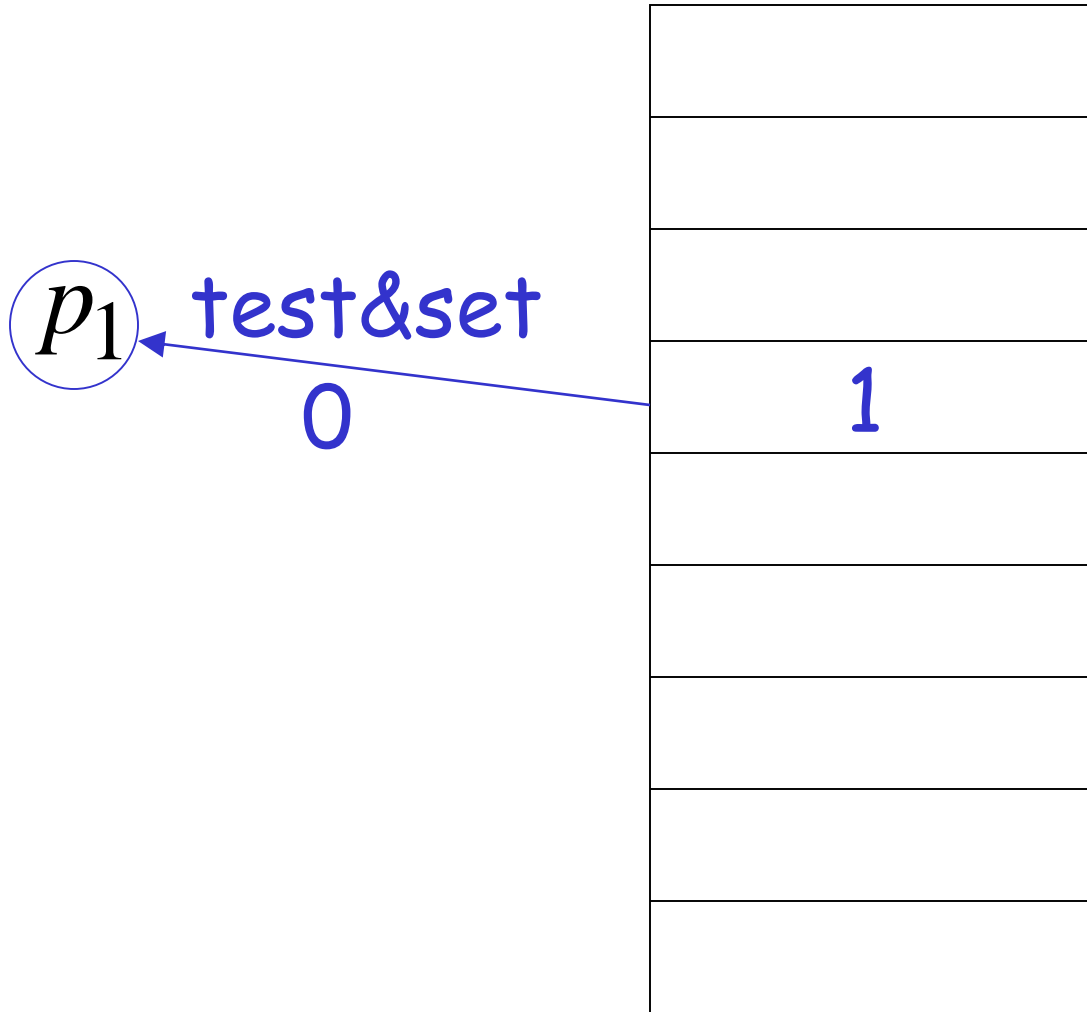
v = 0;

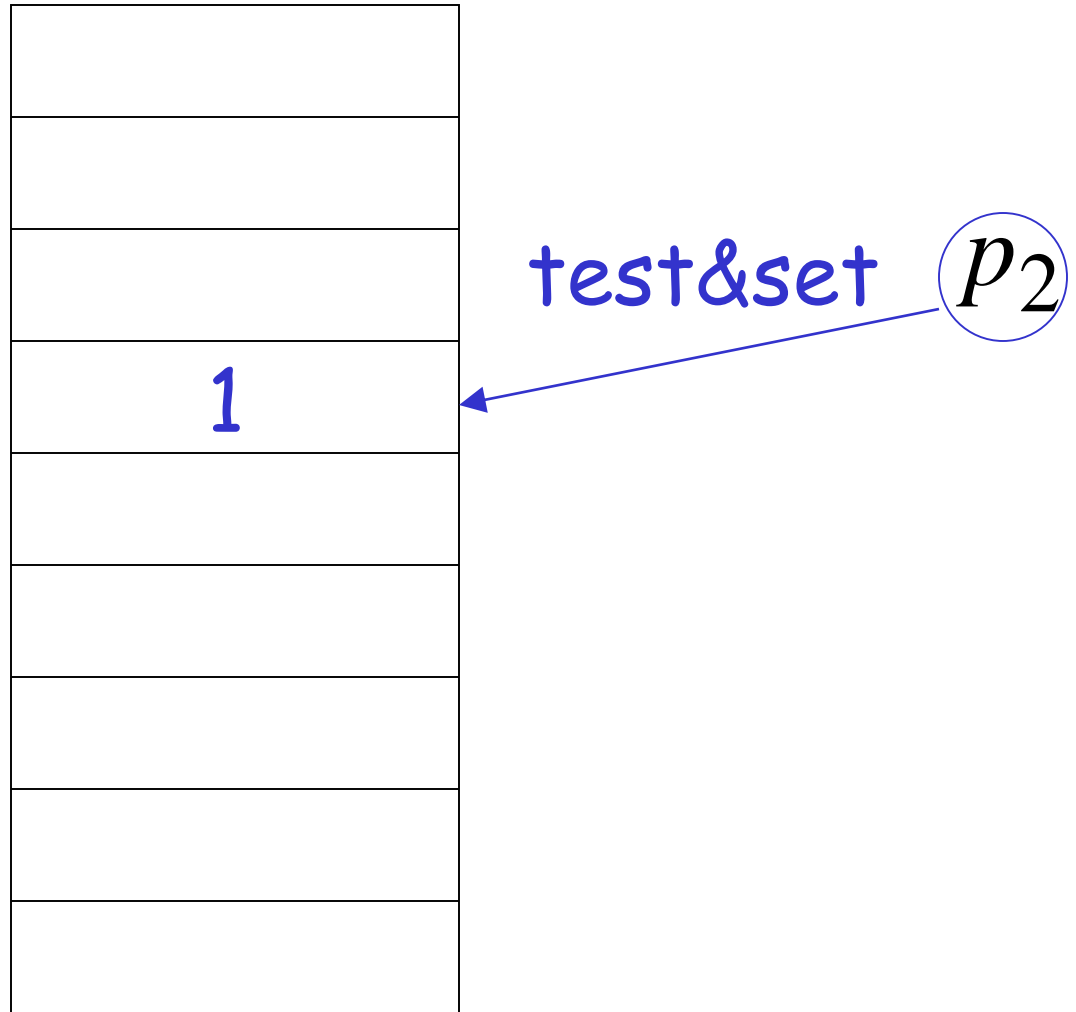
v je binarna promenljiva

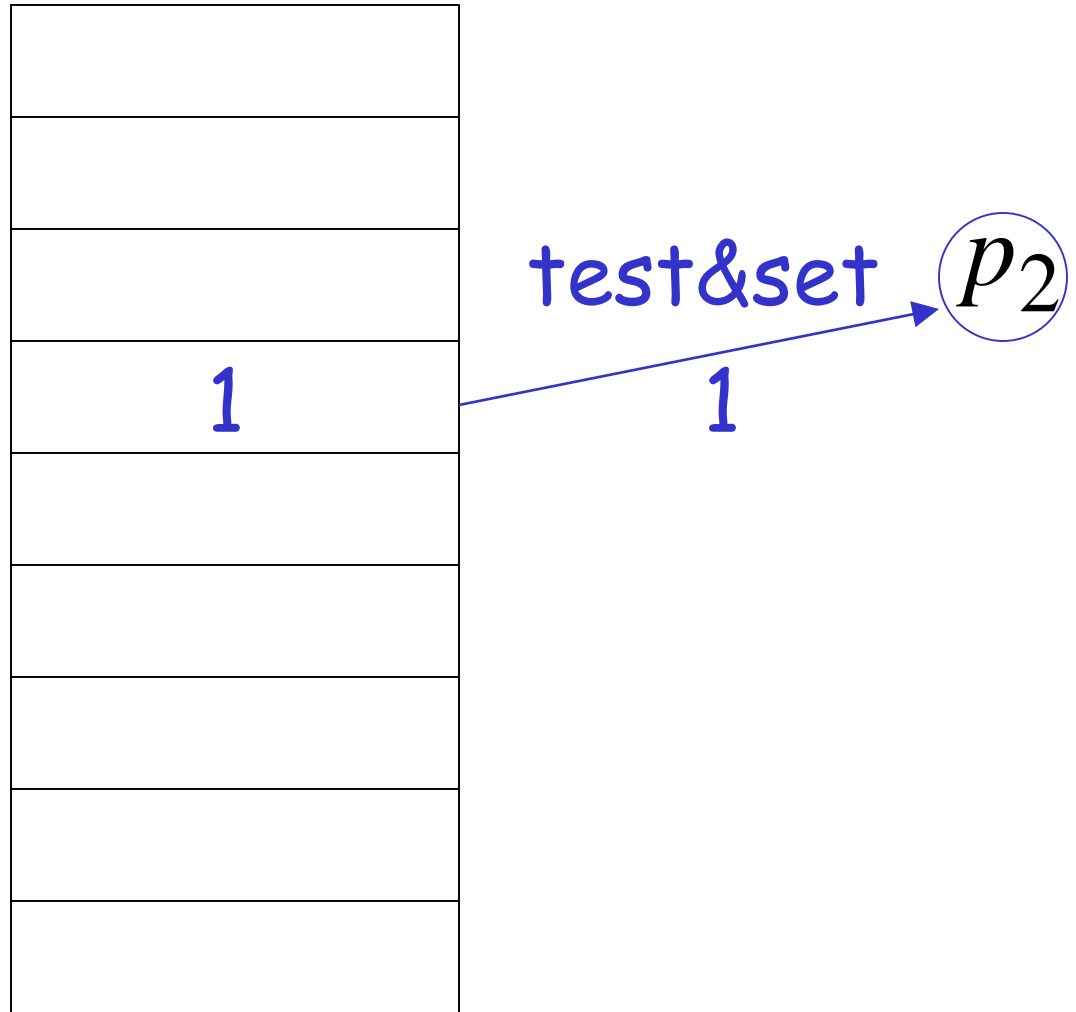
0

p_1 test&set





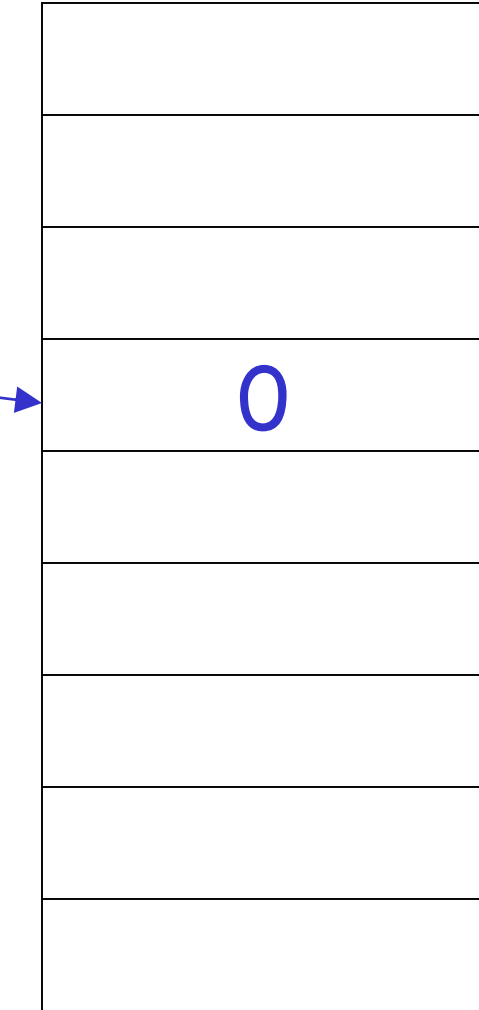




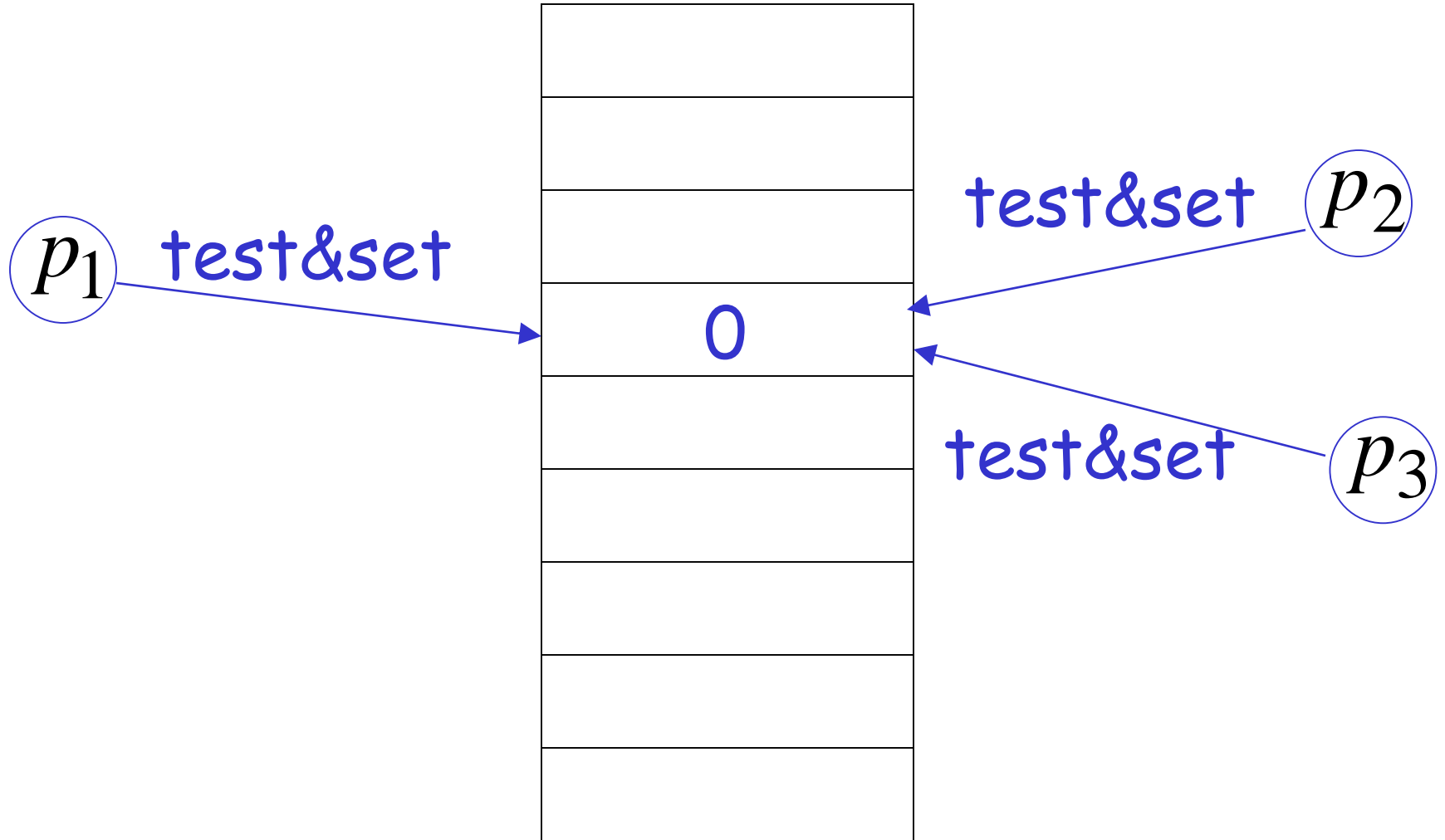
1

p_3

reset

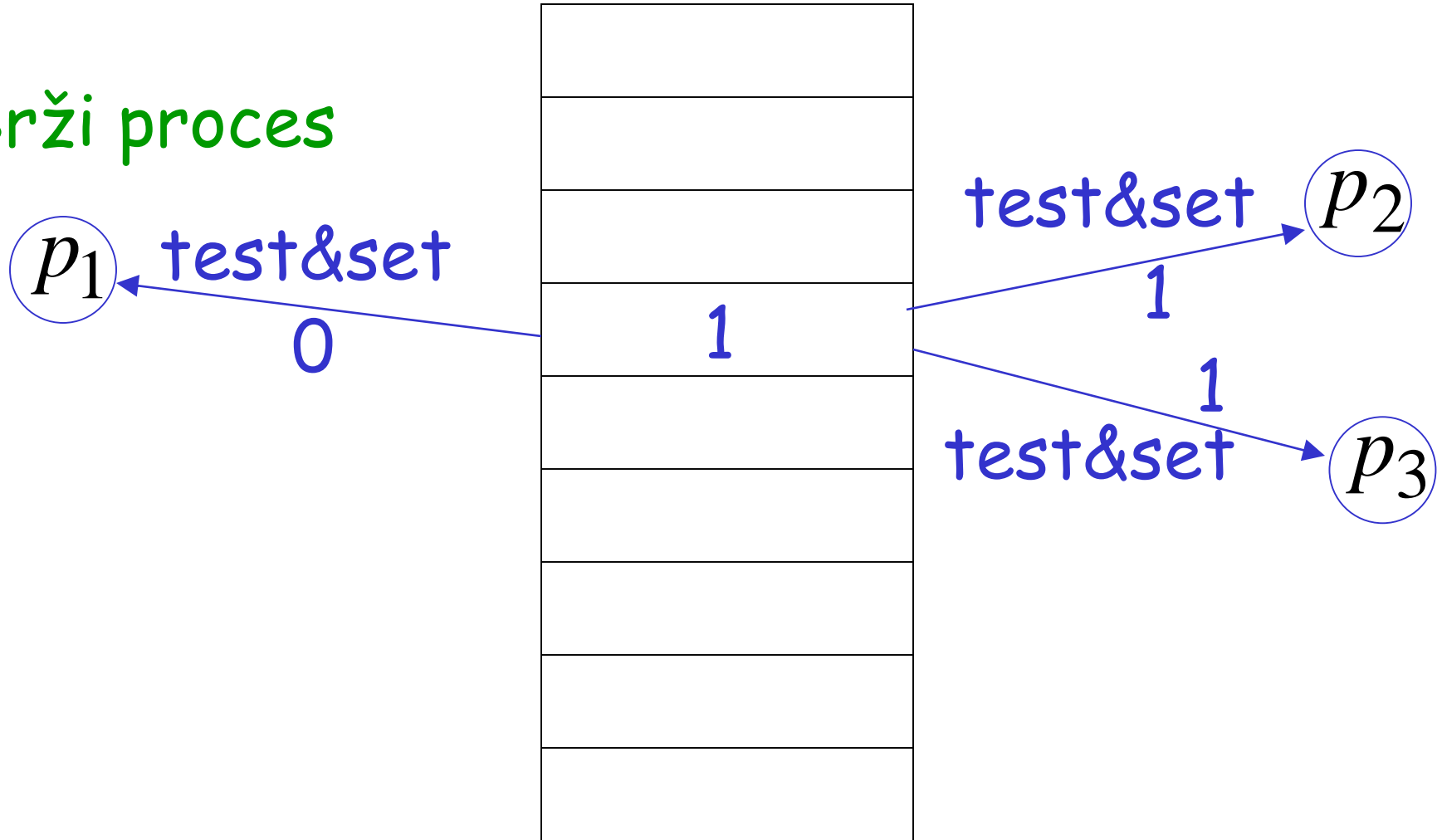


simultani pristupi



simultani pristupi

Brži proces



Read-Modify-Write promenljive

funkcija od v



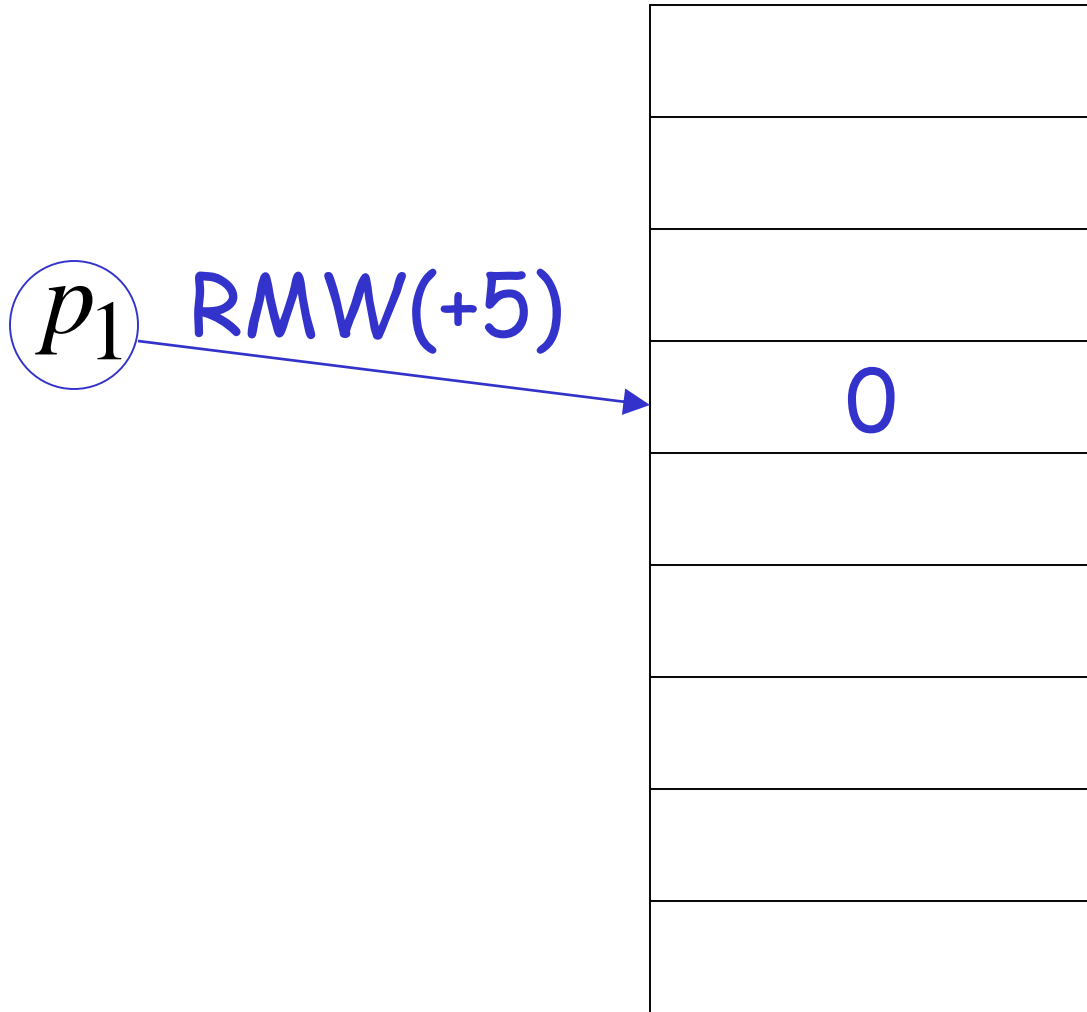
RMW(v, f)

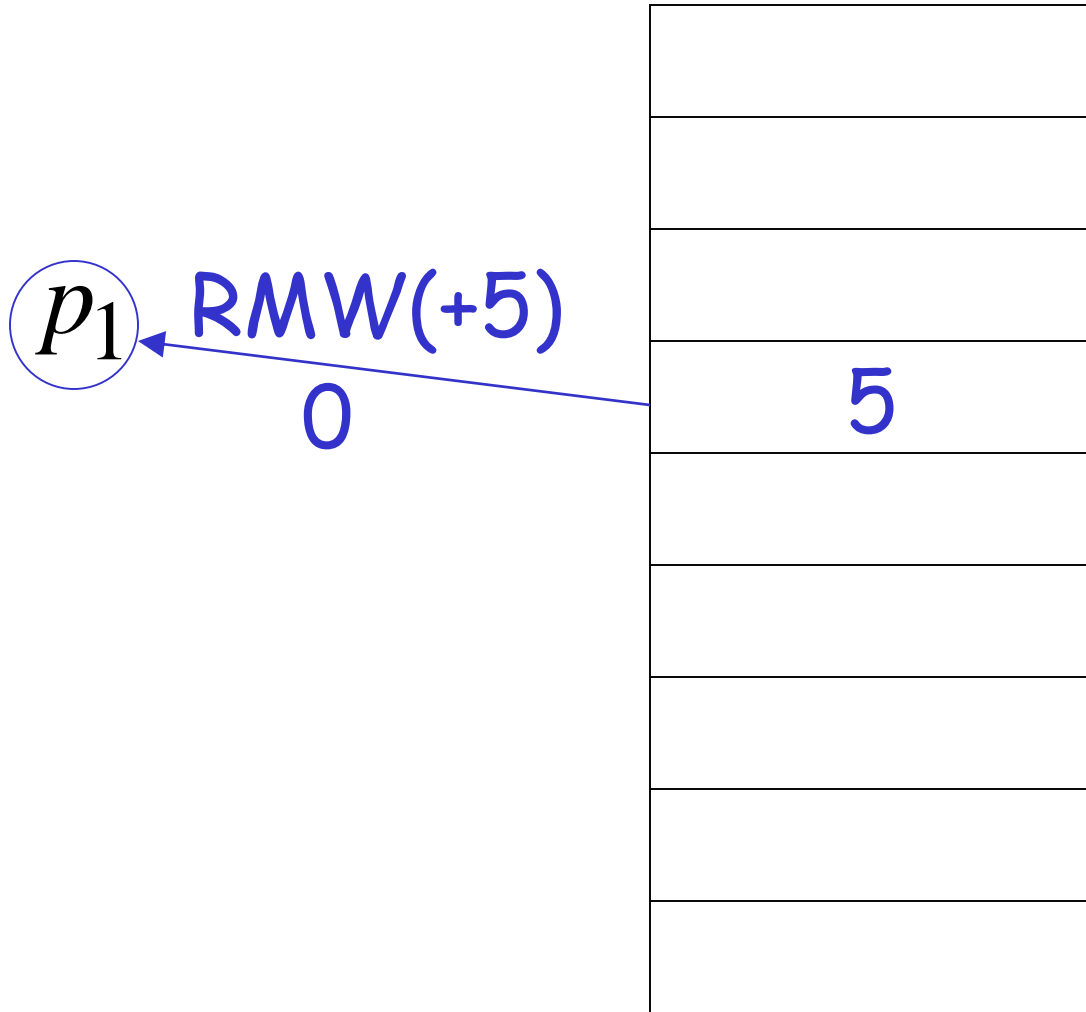
temp = v;

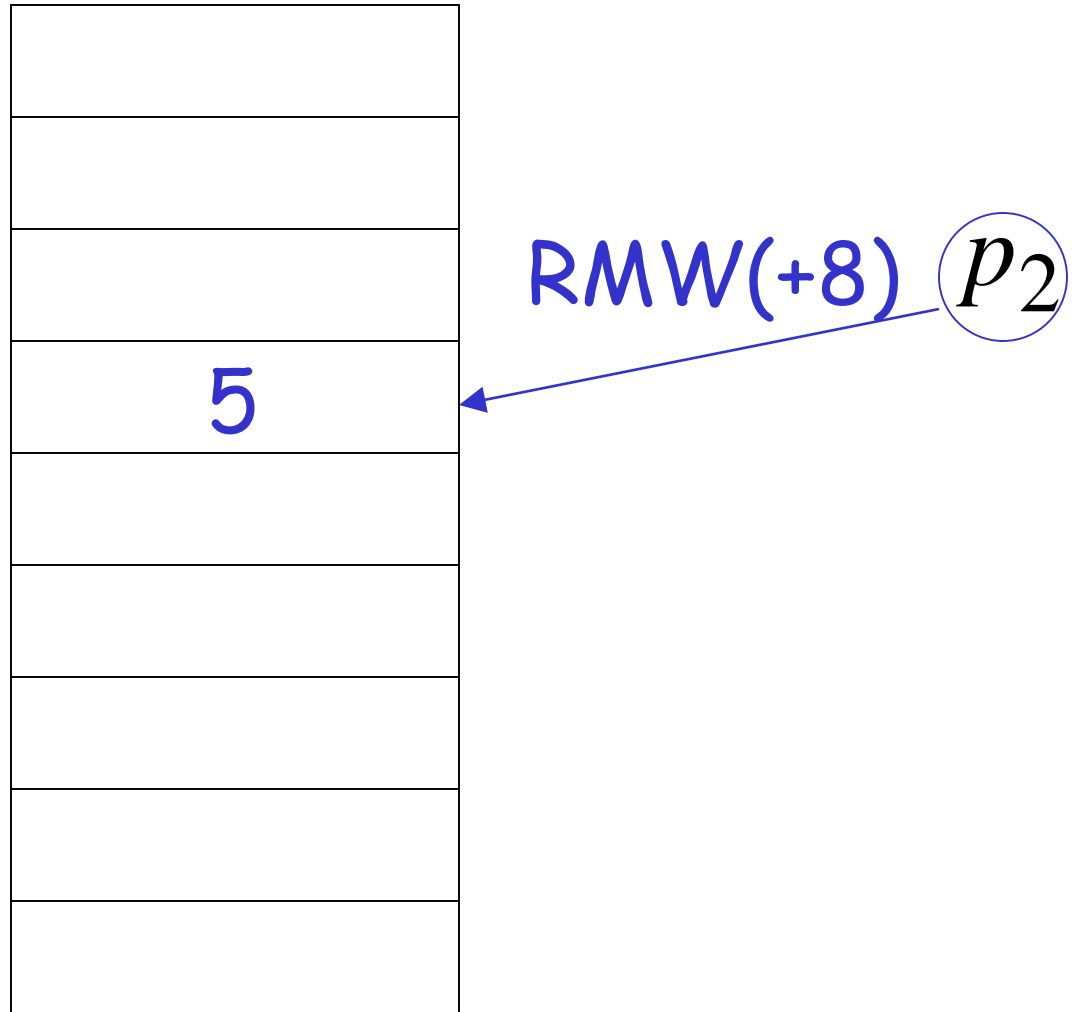
v = f(v);

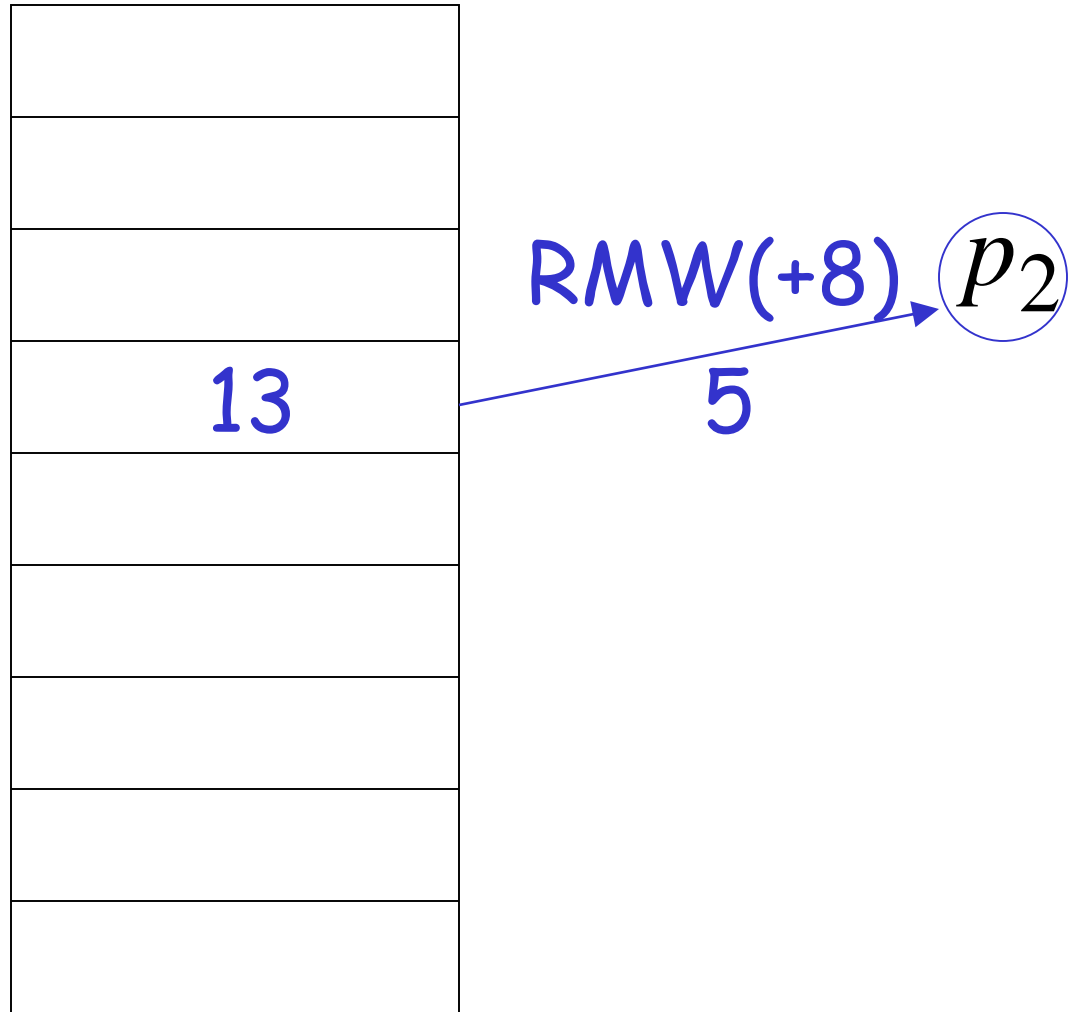
return (temp);

0

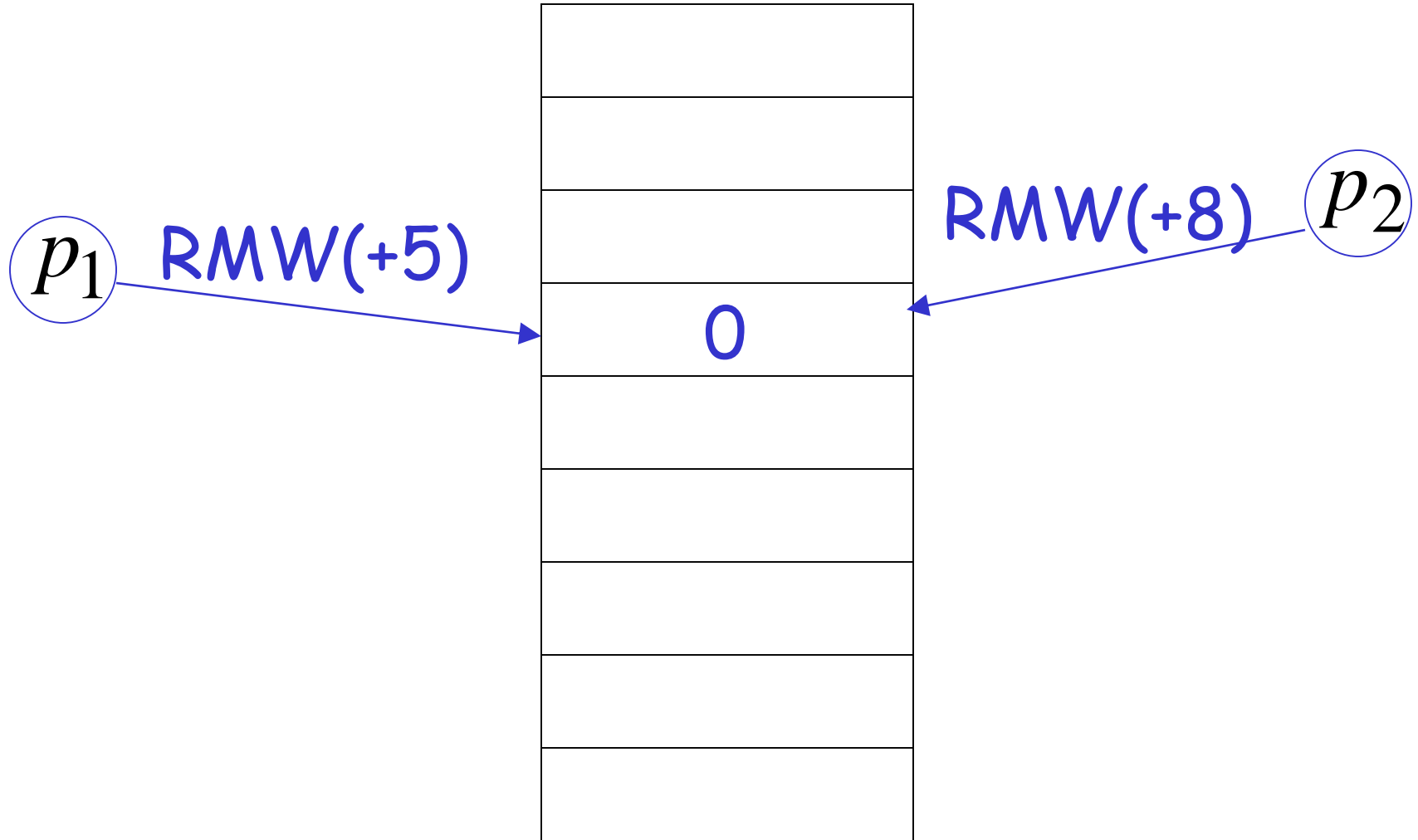




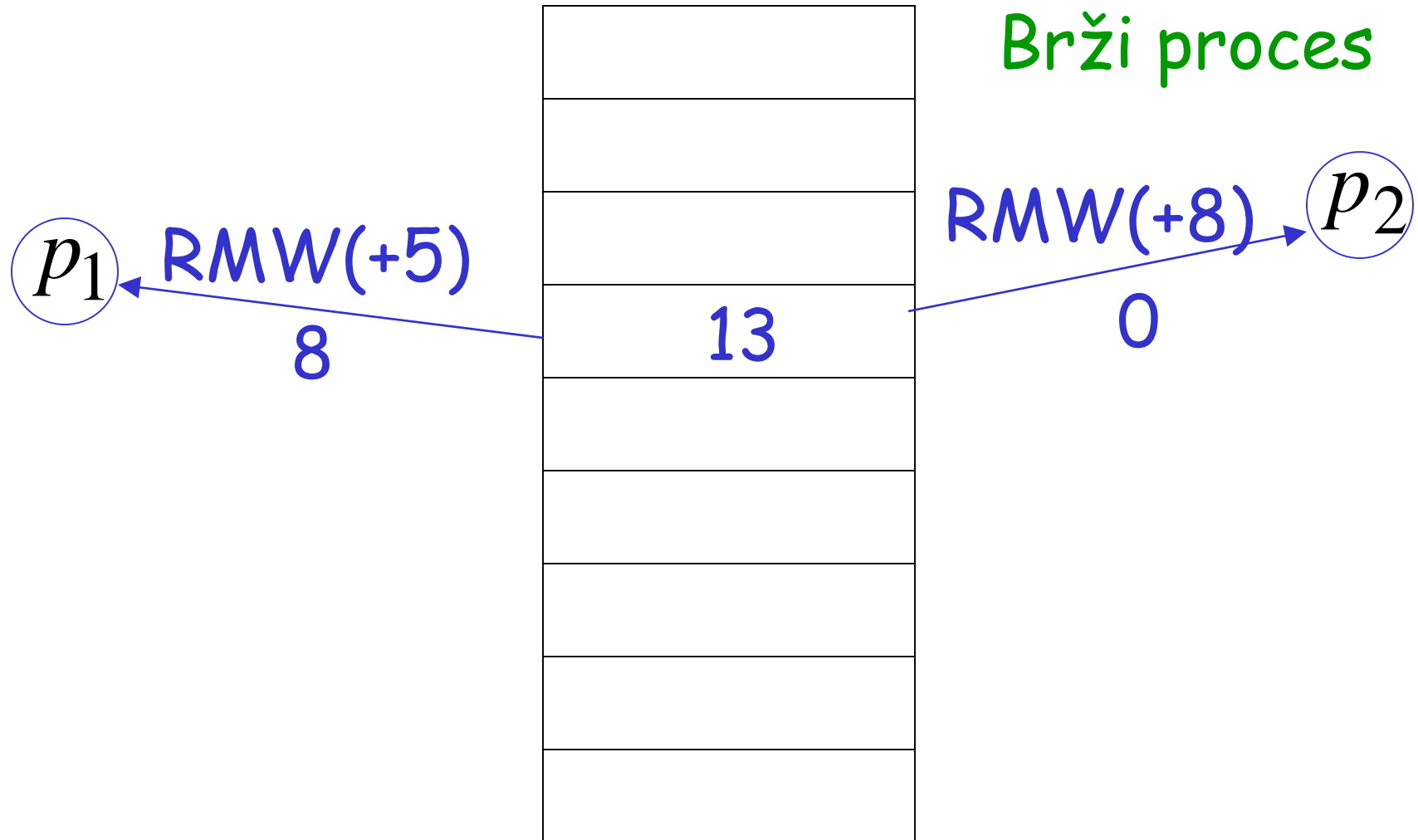




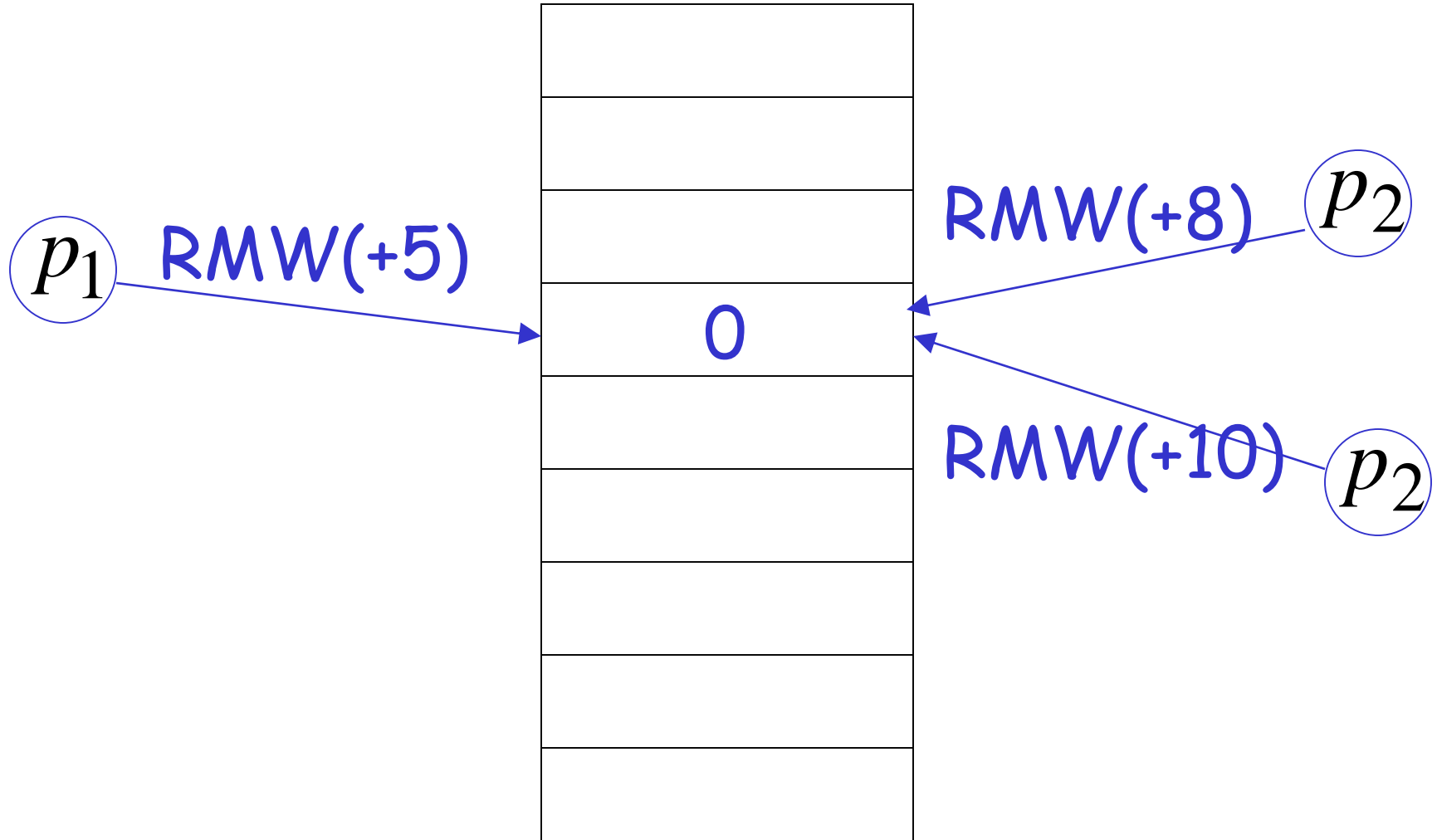
simultani pristupi



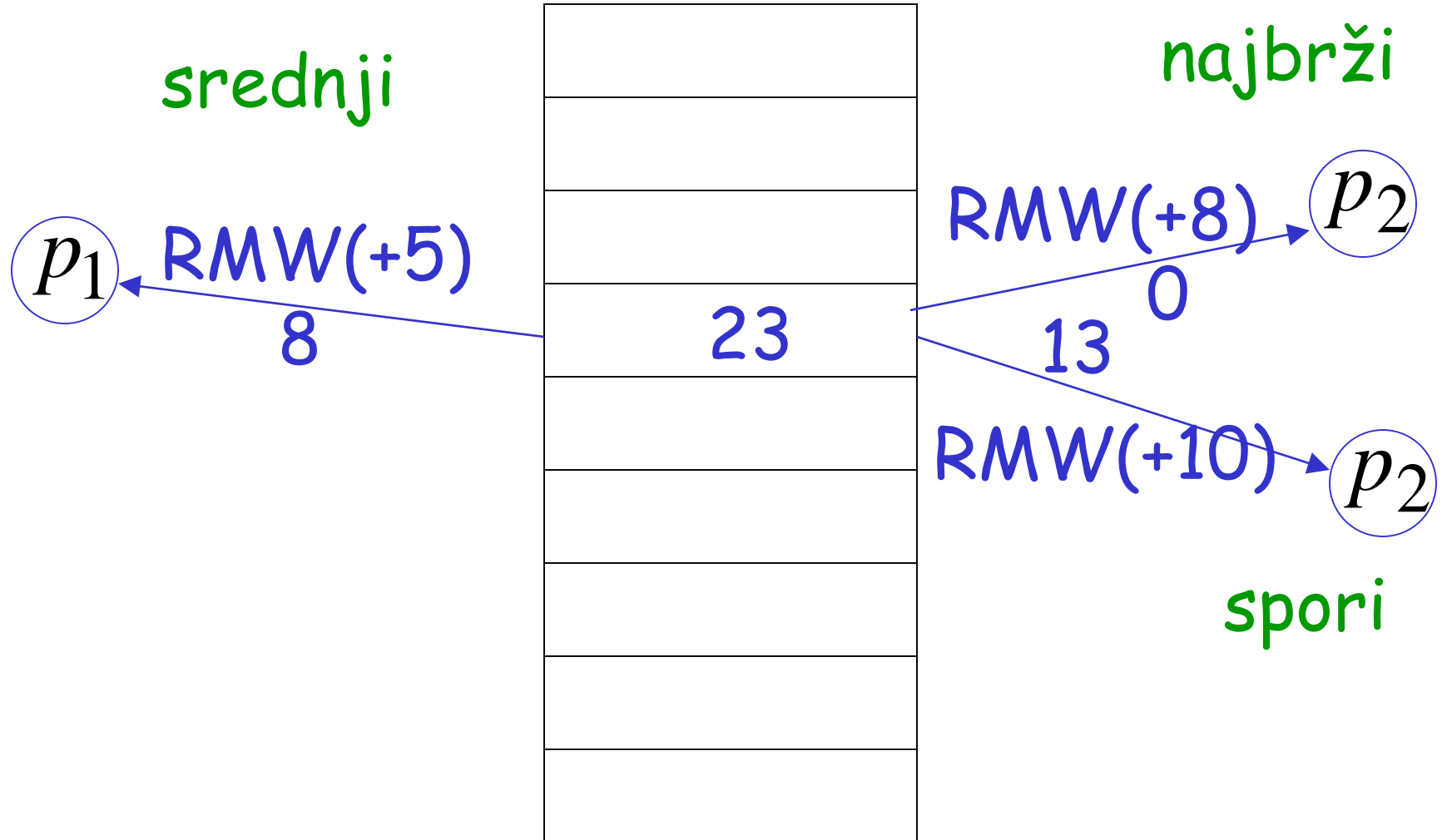
simultani pristupi



simultani pristupi



simultani pristupi



RMW simulacija Test&Set

Test&Set(v)

```
temp = v;  
v = 1;  
return (temp);
```

Reset(v)

```
v = 0;
```

RMW(v, f)

```
temp = v;  
v = f(v); (f(v) = 1)  
return (temp);
```

RMW(v, f)

```
temp = v;  
v = f(v); (f(v) = 0)  
return (temp);
```

Međusobno isključivanje

Proces 1 - program

```
V = 1;  
For (x = 10; x < y; x++) {  
    cout << "hello";  
}  
  
If (t > 10) then  
    while (x < 100)  
        m = 20;  
  
.....
```

Kritična sekcija

Proces 1 - program

Ostatak koda

Kritična sekcija

Ostatak koda

Proces 2 - program

Ostatak koda

Kritična sekcija

Ostatak koda

Samo 1 proces može da uđe u kritičnu sekciju

Kritična sekcija

Proces 1 - program

Ostatak koda

Kritična sekcija
 $v = v + 10;$

Ostatak koda

Proces 2 - program

Ostatak koda

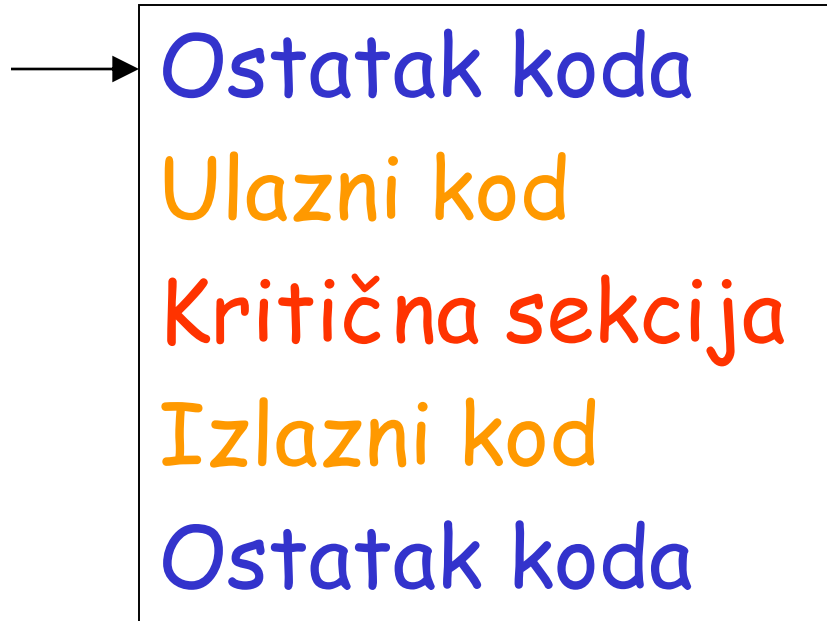
Kritična sekcija
 $v = v + 10;$

Ostatak koda

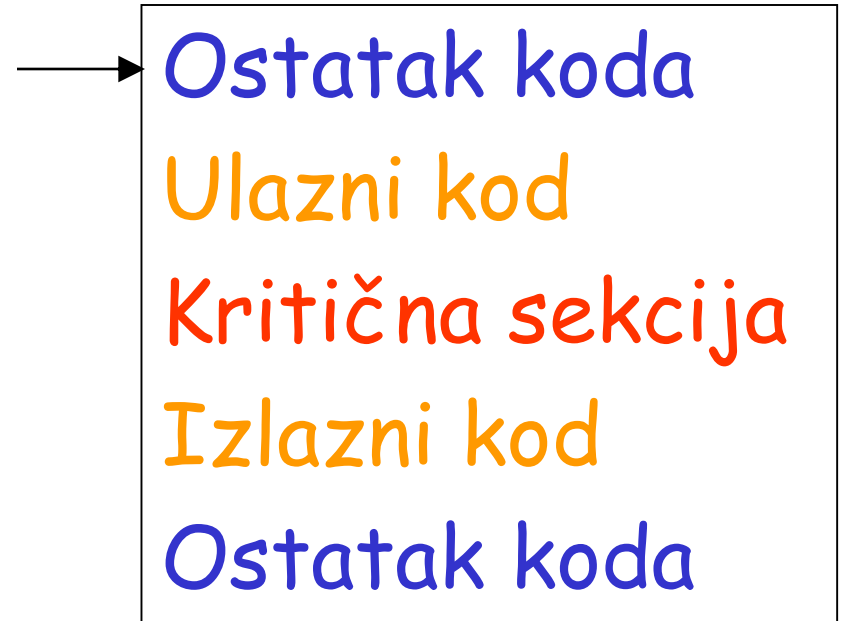
Procesi mogu ažurirati iste promenljive
u kritičnoj sekciji

Međusobno isključivanje

Proces 1



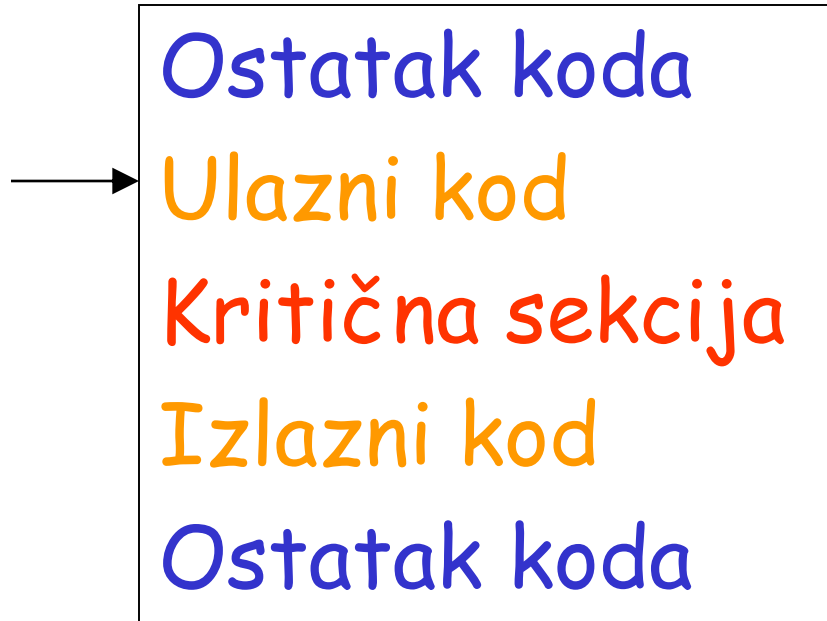
Proces 2



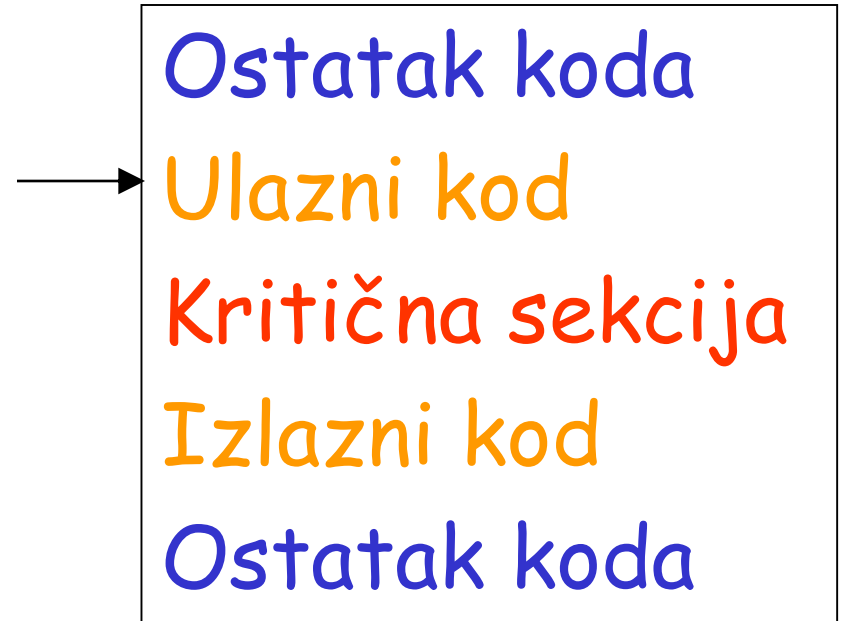
Ulazni i Izlazni kod garantuju da je samo 1 proces u kritičnoj sekciji

Međusobno isključivanje

Proces 1

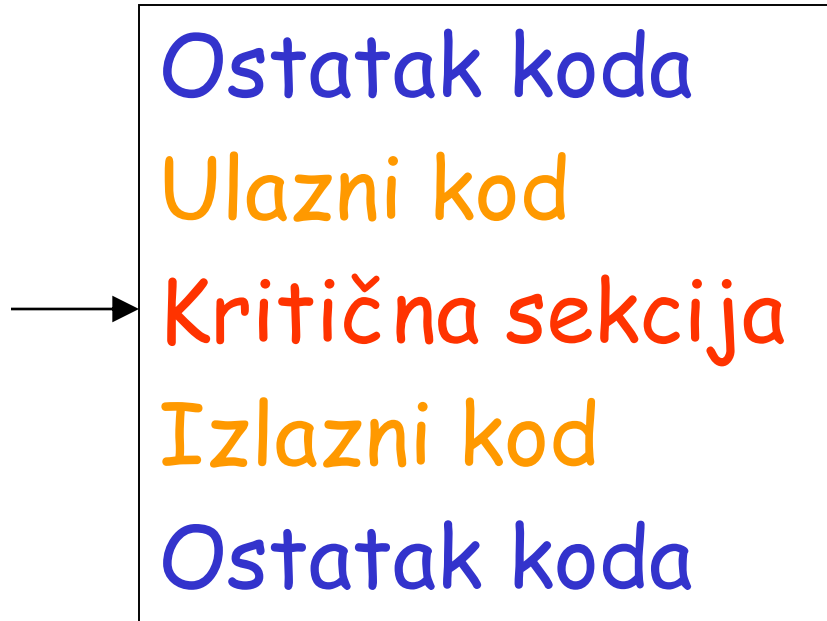


Proces 2

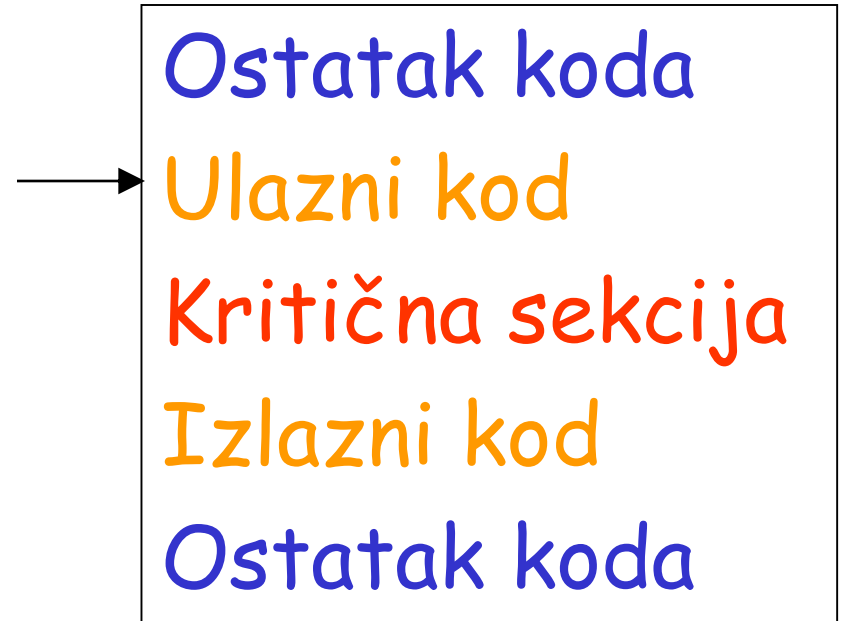


Međusobno isključivanje

Proces 1

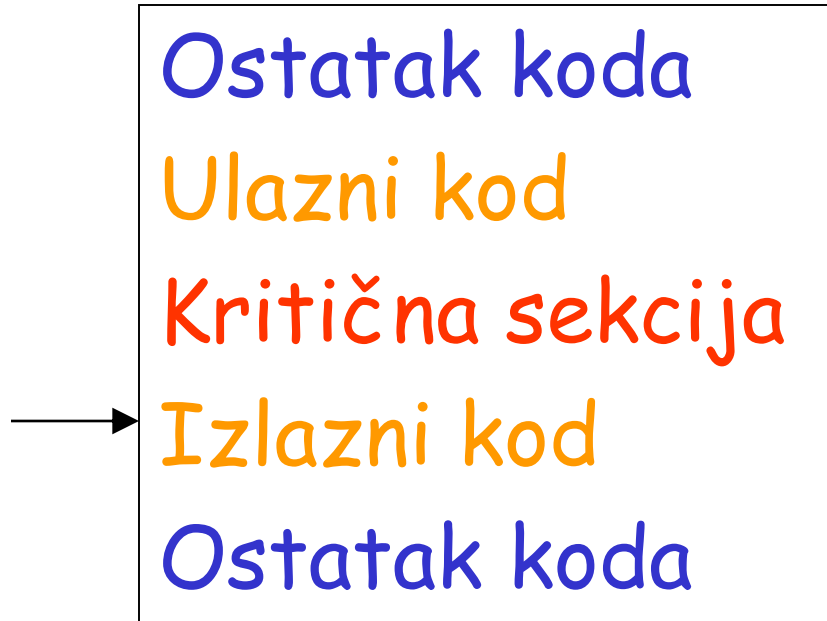


Proces 2

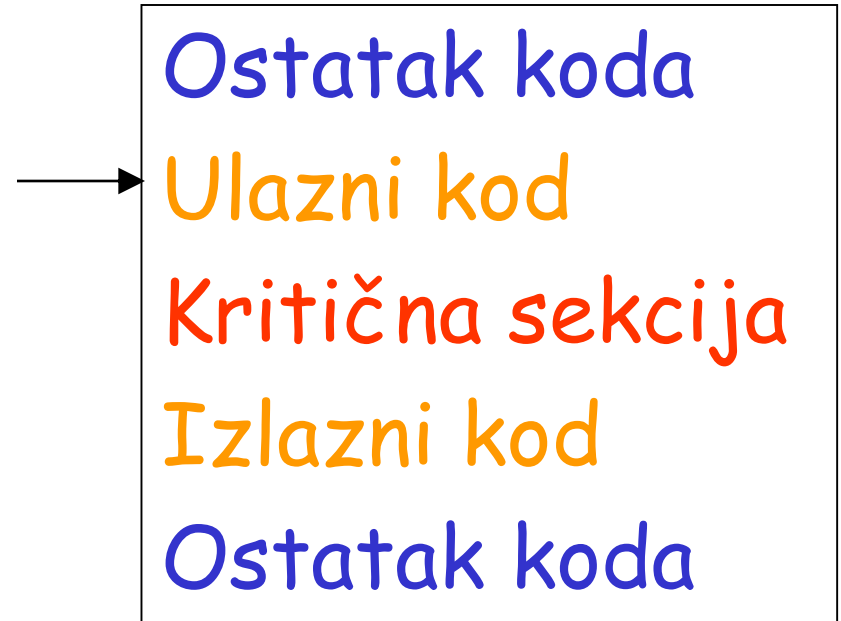


Međusobno isključivanje

Proces 1

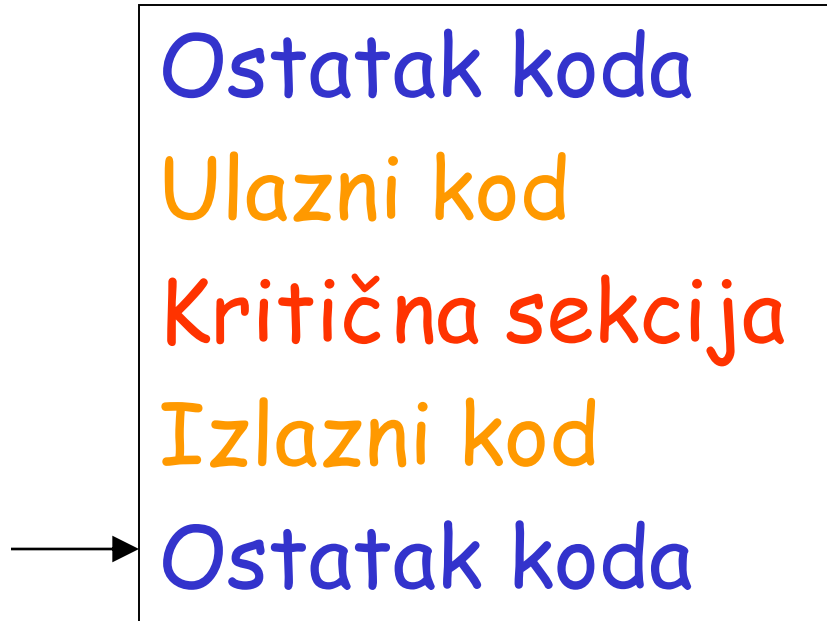


Proces 2

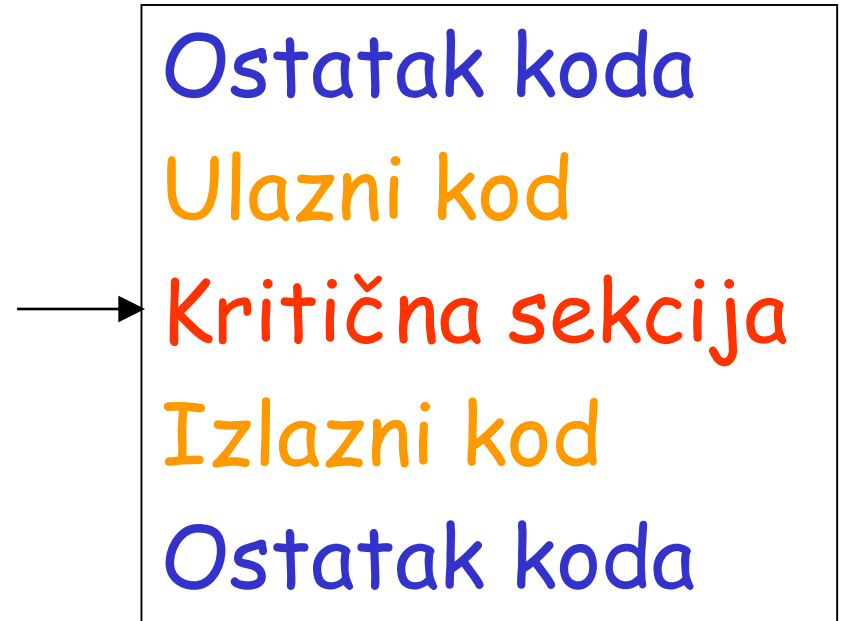


Međusobno isključivanje

Proces 1



Proces 2



Međusobno isključivanje

Proces 1



A vertical sequence of five text elements: 'Ostatak koda' (blue), 'Ulazni kod' (orange), 'Kritična sekcija' (red), 'Izlazni kod' (orange), and 'Ostatak koda' (blue). A black arrow points from the left to the first 'Ostatak koda' element.

Ostatak koda
Ulazni kod
Kritična sekcija
Izlazni kod
Ostatak koda

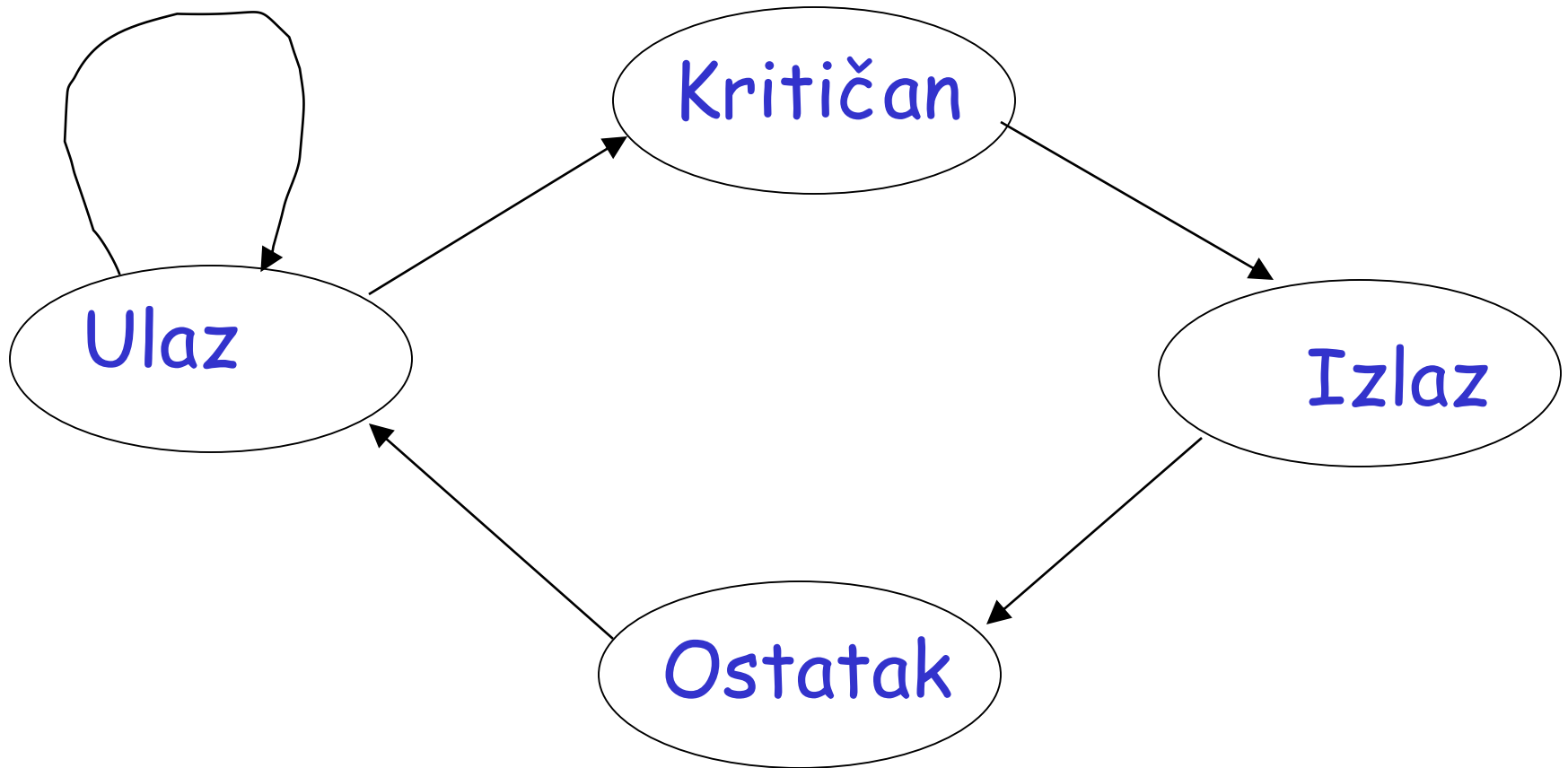
Proces 2



A vertical sequence of five text elements: 'Ostatak koda' (blue), 'Ulazni kod' (orange), 'Kritična sekcija' (red), 'Izlazni kod' (orange), and 'Ostatak koda' (blue). A black arrow points from the left to the 'Izlazni kod' element.

Ostatak koda
Ulazni kod
Kritična sekcija
Izlazni kod
Ostatak koda

Međusobno isključivanje



Atributi algoritama međusobnog isključivanja

Nema međusobnog blokiranja (deadlock):
ako je neki proces u ulaznoj sekciji onda
će neki proces ući u kritičnu sekciju

Nema trajnog zaključavanja (lockout):
ako je neki proces u ulaznoj sekciji onda
će isti proces ući u kritičnu sekciju

Međusobno isključivanje

test&set promenljive

Ulaz: While (test&set(v) = 1)

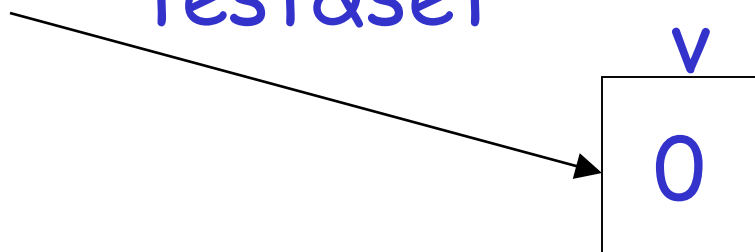
Kritična sekcija

Izlaz: Reset (v)

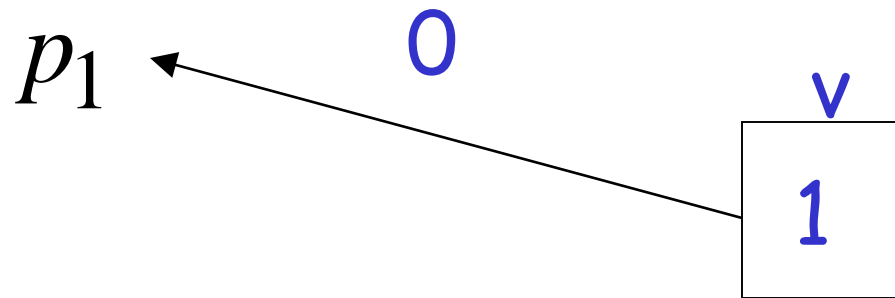
ulaz

p_1

test&set



kritična sekcija



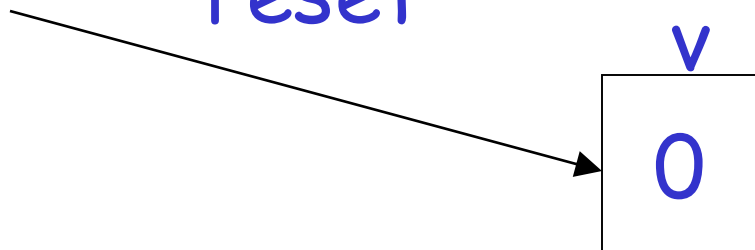
izlaz

p_1

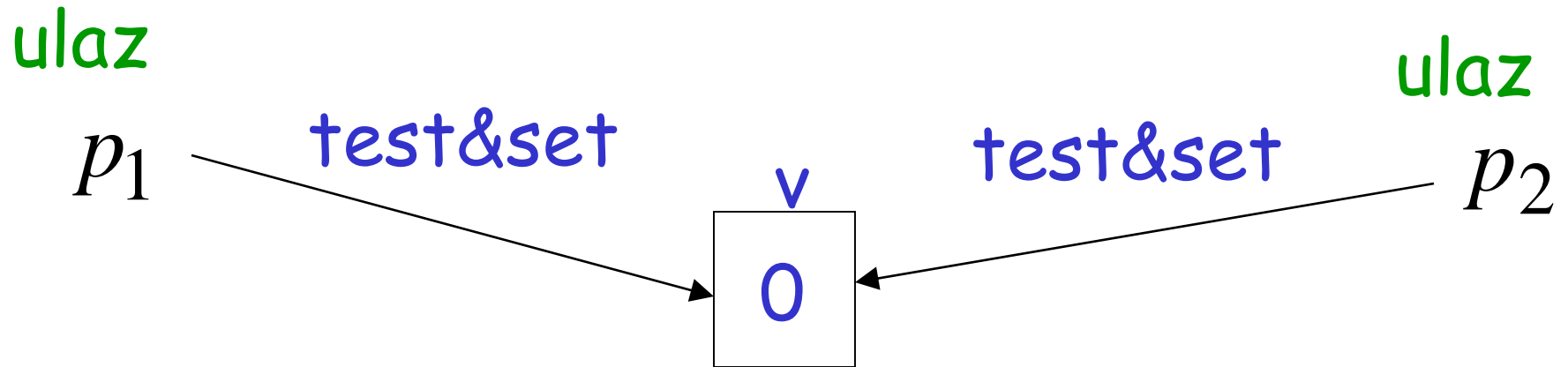
reset

v

0



Dva procesora



ulaz

p_1

1

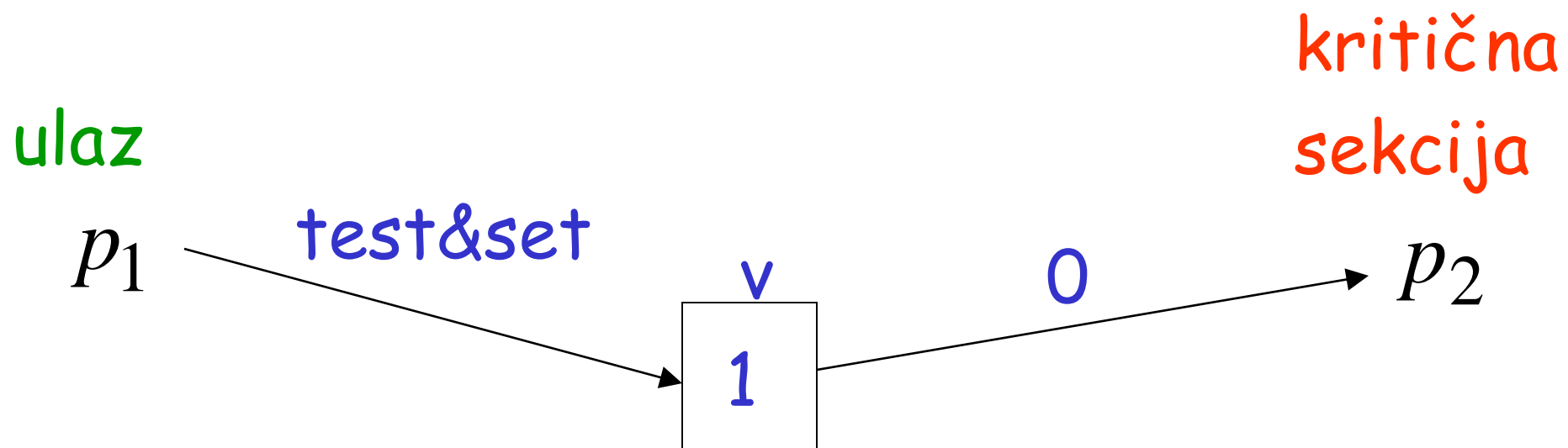
v

1

0

kritična
sekcija

p_2



ulaz

p_1

1

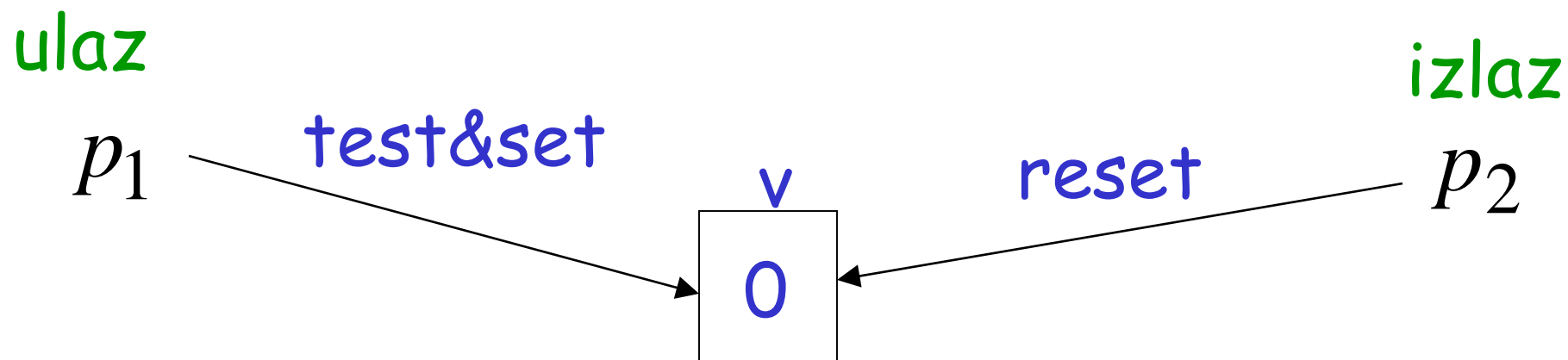
v

1

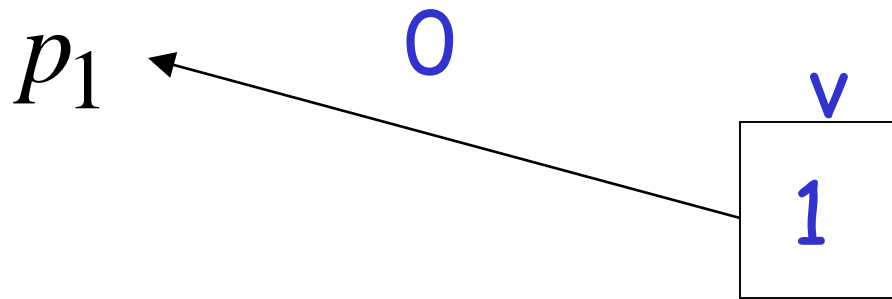
0

kritična
sekcija

p_2



kritična
sekcija



izlaz

p_1

reset

v

0

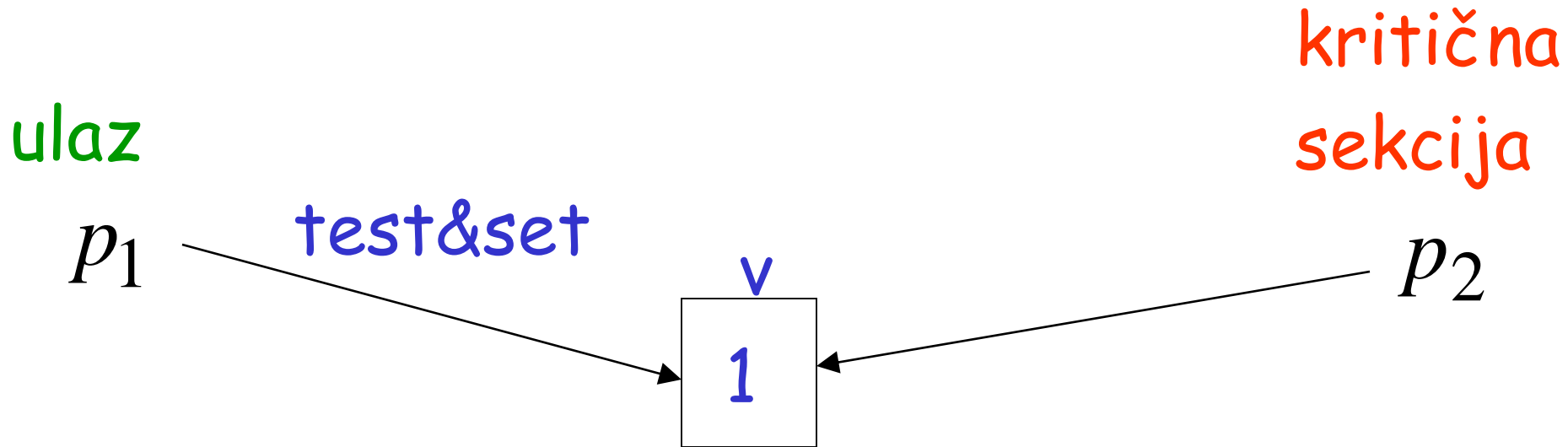


The diagram illustrates a reset operation. A black arrow originates from the label p_1 and points to a square box. Above the box is the label v , and inside the box is the number 0. The word 'reset' is written in blue above the arrow.

Problem: ovaj algoritam ne garantuje da nema trajnog zaključavanja, proces može „gladovati“

Primer:

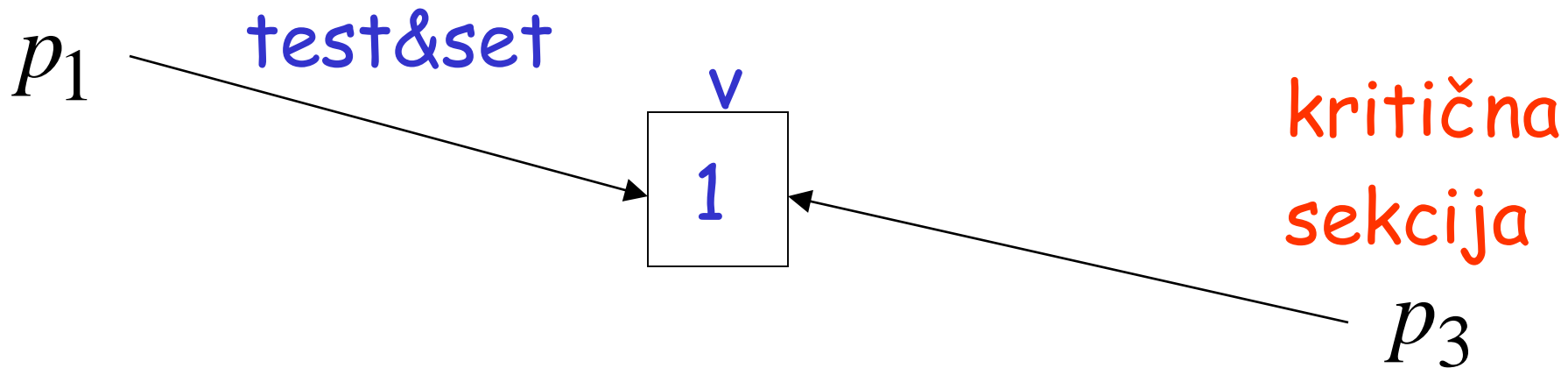
p_1 može da nikad ne uđe u kritičnu sekciju



Primer:

p_1 može da nikad ne uđe u kritičnu sekciju

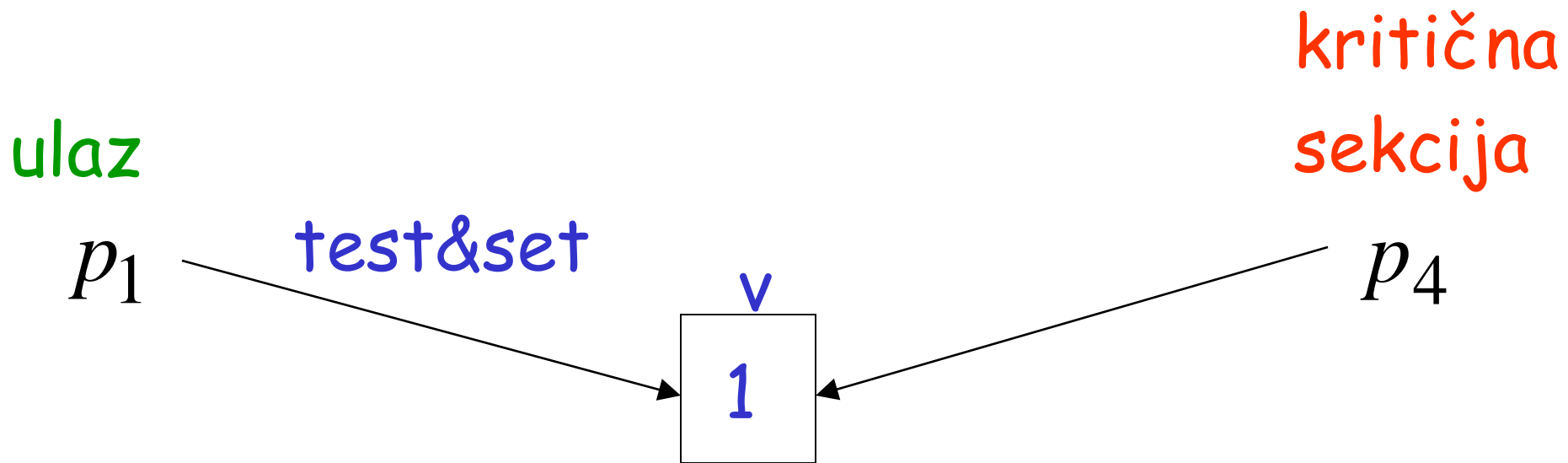
ulaz



p_2, p_3, p_4 : brži od p_1

Primer:

p_1 može da nikad ne uđe u kritičnu sekciju



Međusobno isključivanje

read-modify-write promenljive

Deljena promenljiva v

v.first Ulaznica tekućeg procesa
u kritičnoj sekciji

v.last Ulaznica poslednjeg procesa
koji čeka u ulaznoj sekciji

Ulaz: $p = \text{RMW}(v, (v.\text{first}, v.\text{last} + 1))$

Repeat

$q = \text{RMW}(v, v)$

Until $q.\text{first} = p.\text{last}$

Kritična sekcija

Izlaz: $\text{RMV}(v, (v.\text{first}+1, v.\text{last}))$

(p i q su lokalne promenljive)

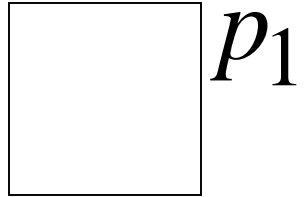
v.first

1

v.last

1

p.last



v.first

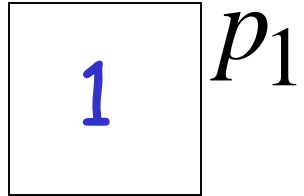
1

v.last

1

ulaz

p.last



v.first

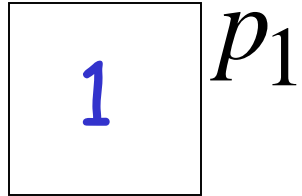
1

v.last

2

kritična
sekcija

p.last



v.first

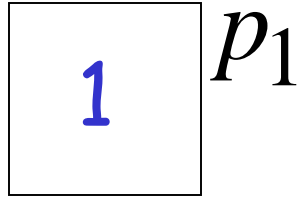
1

v.last

2

izlaz

p.last



v.first

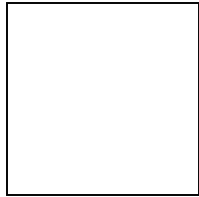
2

v.last

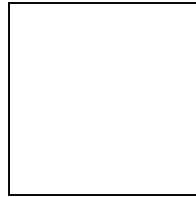
2

Četiri procesa

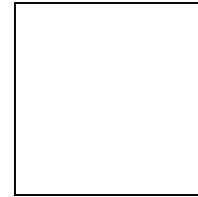
p.last



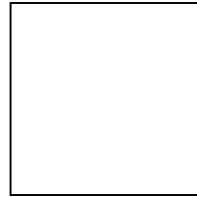
p_1



p_2



p_3



p_4

v.first

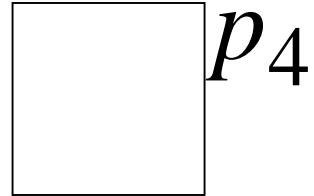
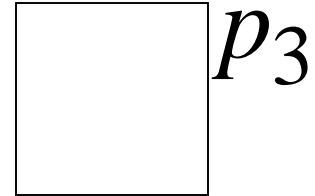
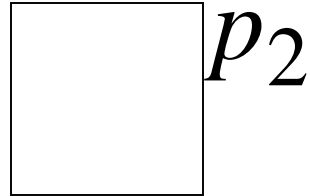
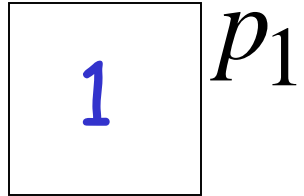
1

v.last

1

ulaz

p.last



v.first

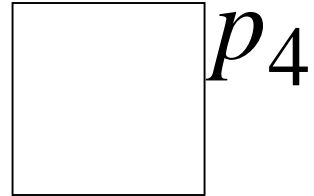
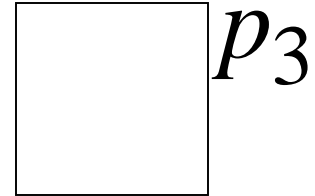
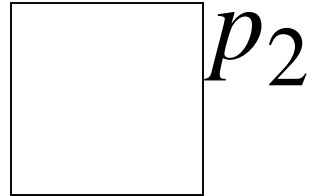
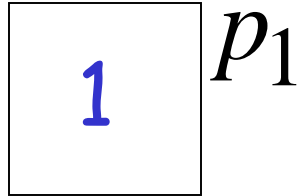
1

v.last

2

ulaz

p.last

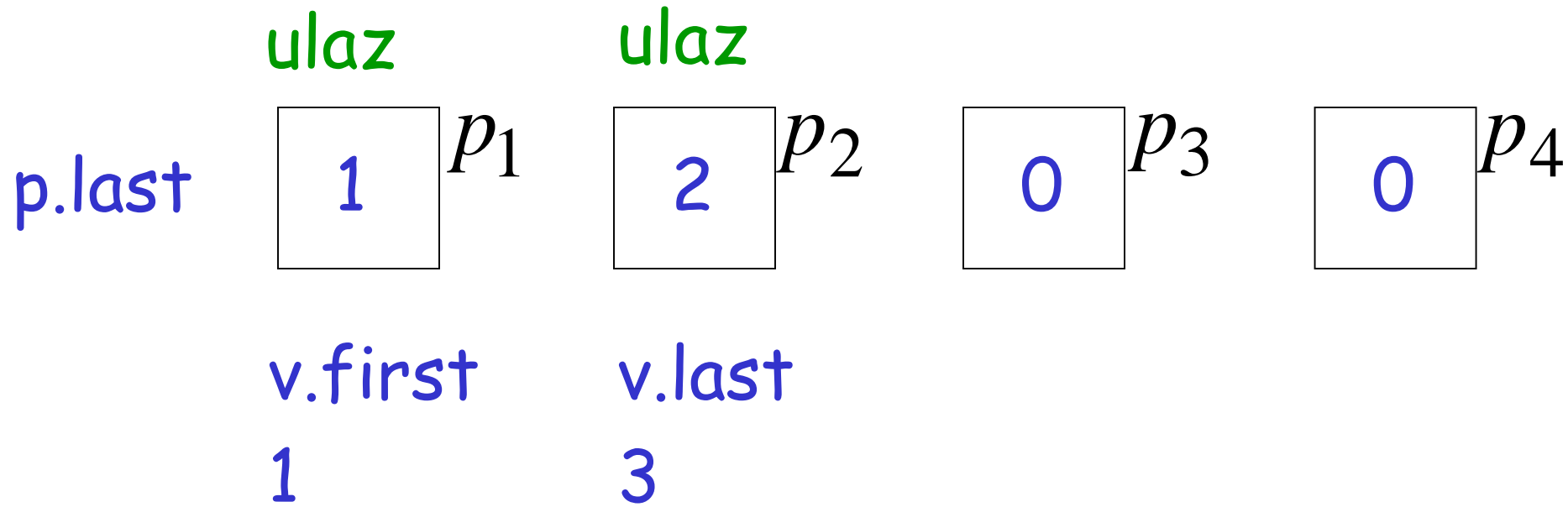


v.first

1

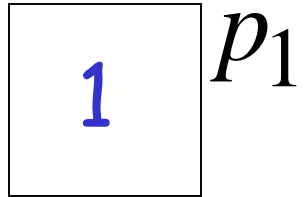
v.last

2

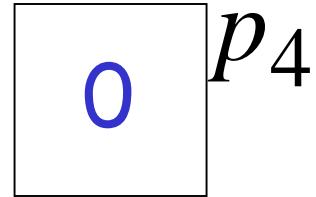
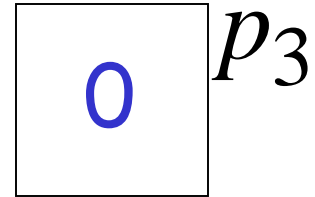
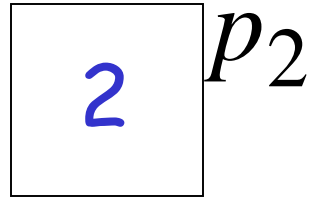


p.last

ulaz



ulaz

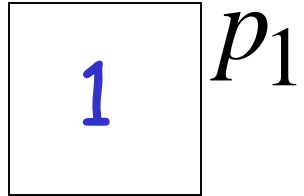


v.first
1

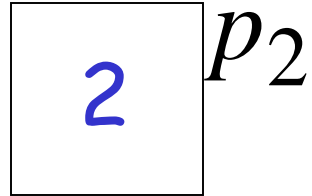
v.last
3

p.last

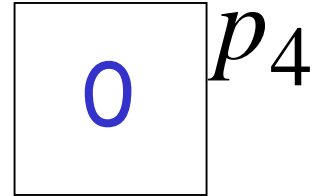
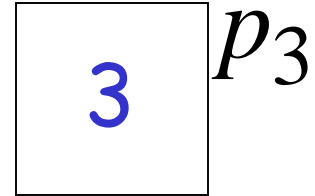
ulaz



ulaz



ulaz



v.first

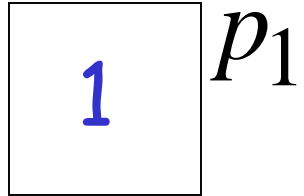
1

v.last

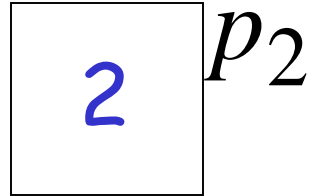
4

p.last

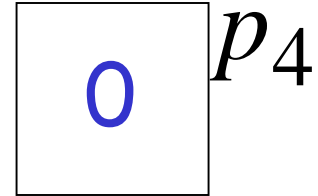
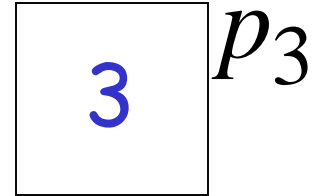
ulaz



ulaz



ulaz

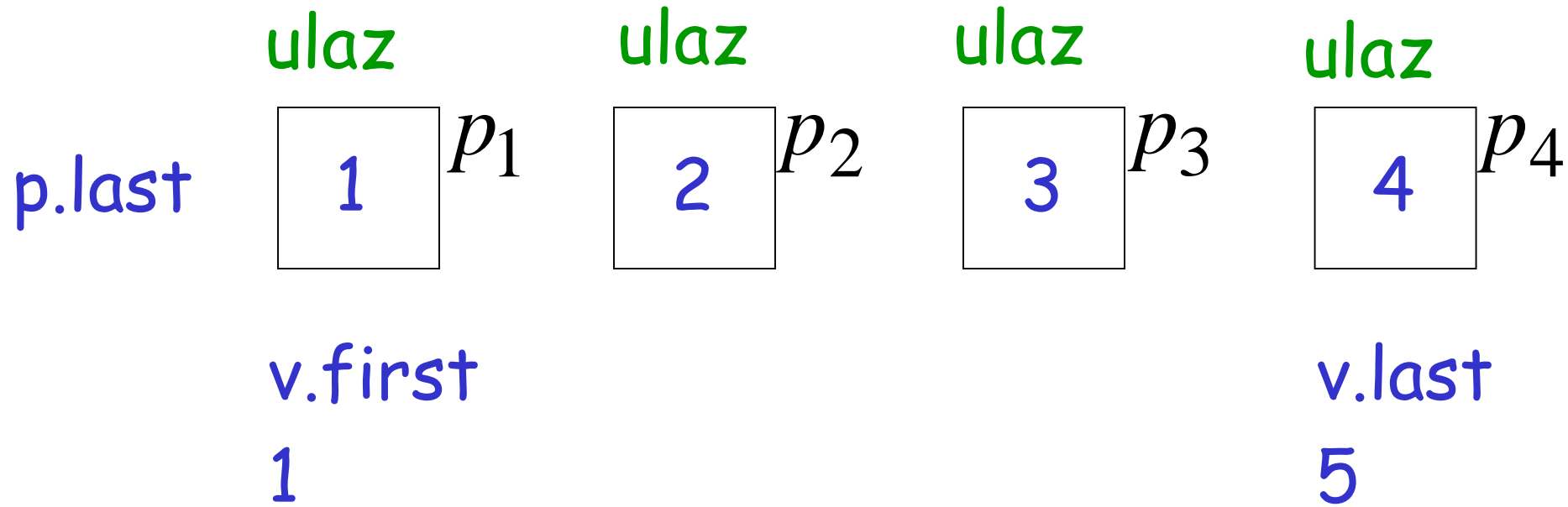


v.first

1

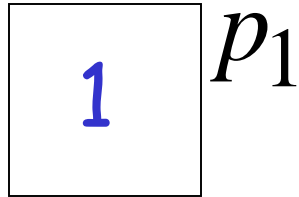
v.last

4

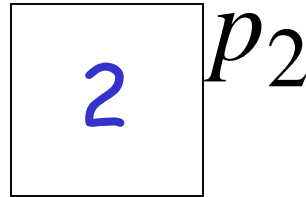


kritična
sekcija

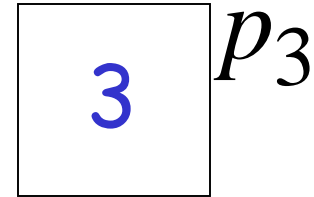
p.last



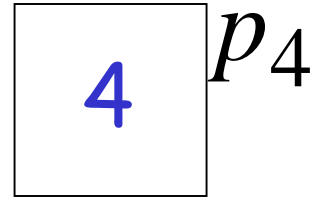
ulaz



ulaz



ulaz



v.first

1

v.last

5

p.last

izlaz

1

 p_1

ulaz

2

 p_2

ulaz

3

 p_3

ulaz

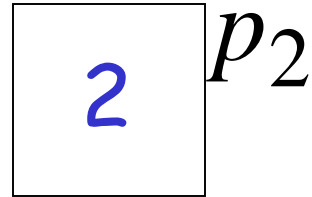
4

 p_4

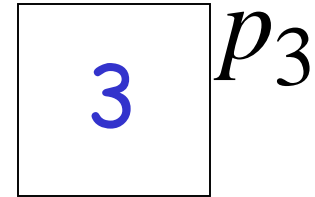
v.first
2

v.last
5

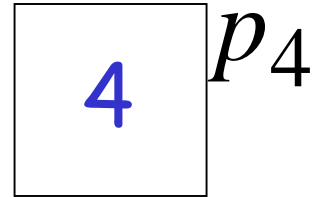
kritična
sekcija



ulaz



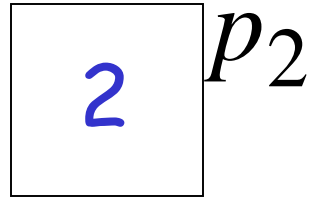
ulaz



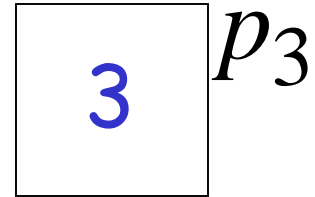
v.first
2

v.last
5

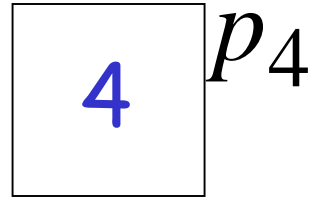
izlaz



ulaz



ulaz



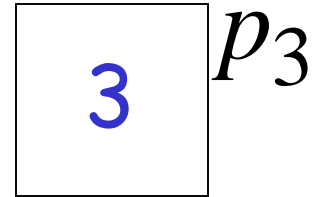
v.first

3

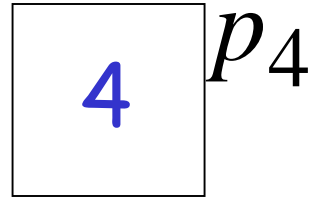
v.last

5

ulaz



ulaz



v.first

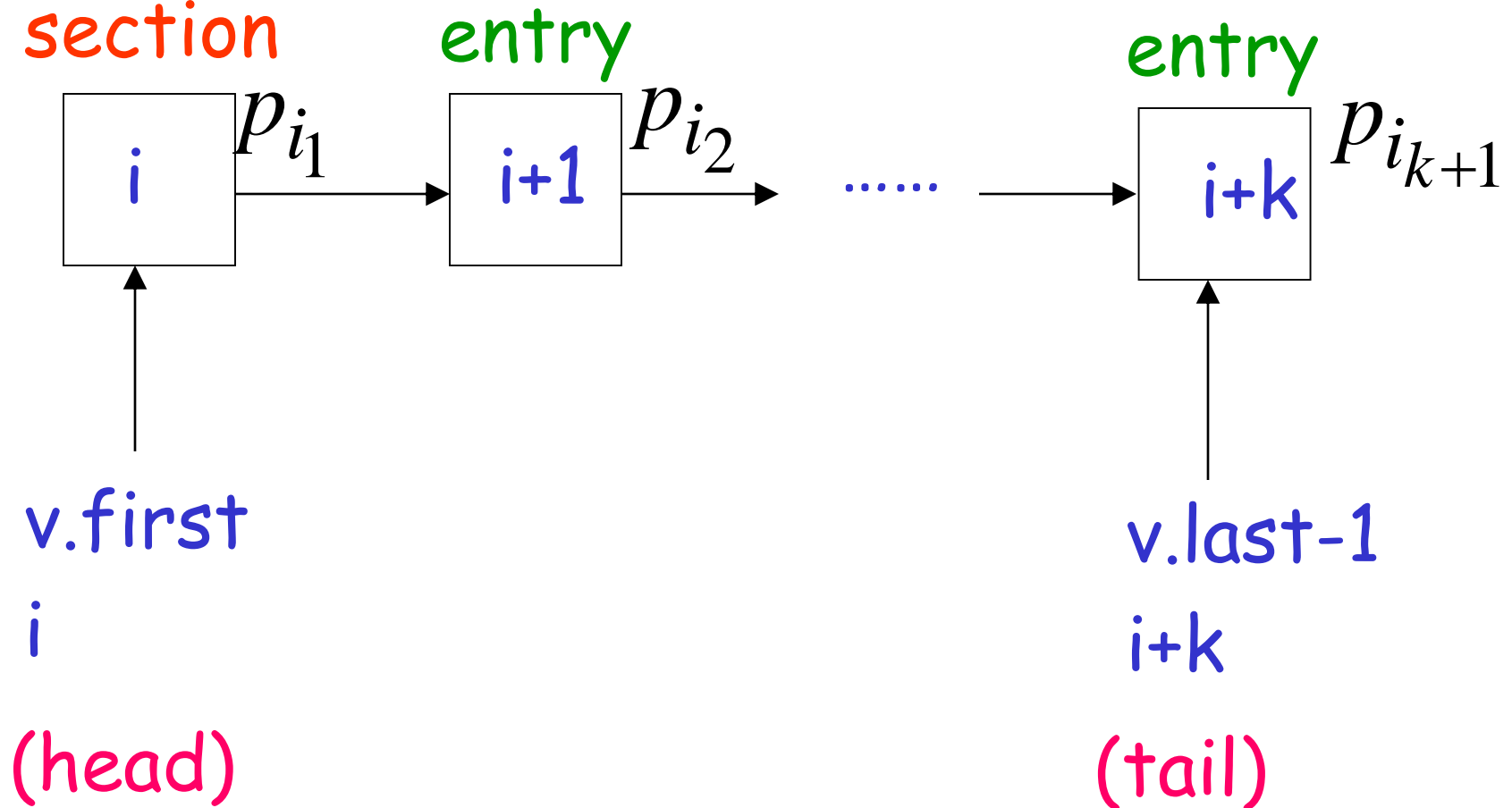
3

v.last

5

Ovo ponašanje je slično sa redom čekanja

critical
section



Dobre osobine ovog algoritma:

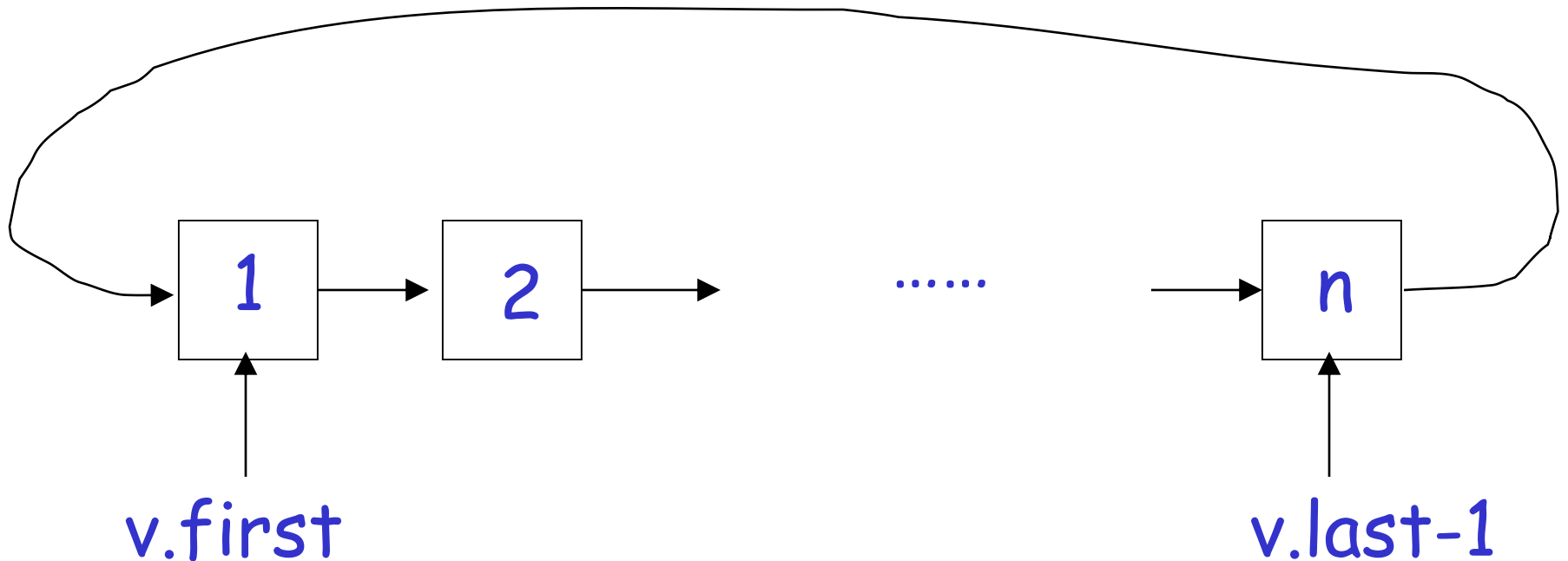
- Garantuje da nema trajnog zaključavanja (svaki proces će konačno ući u kritičnu sekciju)
- Koristi samo jednu deljenu prom. (v)

Jedan problem:

vrednosti mogu da rastu neograničeno

$i \quad i+1 \quad \dots \quad i+k \quad \dots \quad + \infty$

Rešenje: kružni red



Potrebno je samo n različitih vrednosti

(za n procesa)