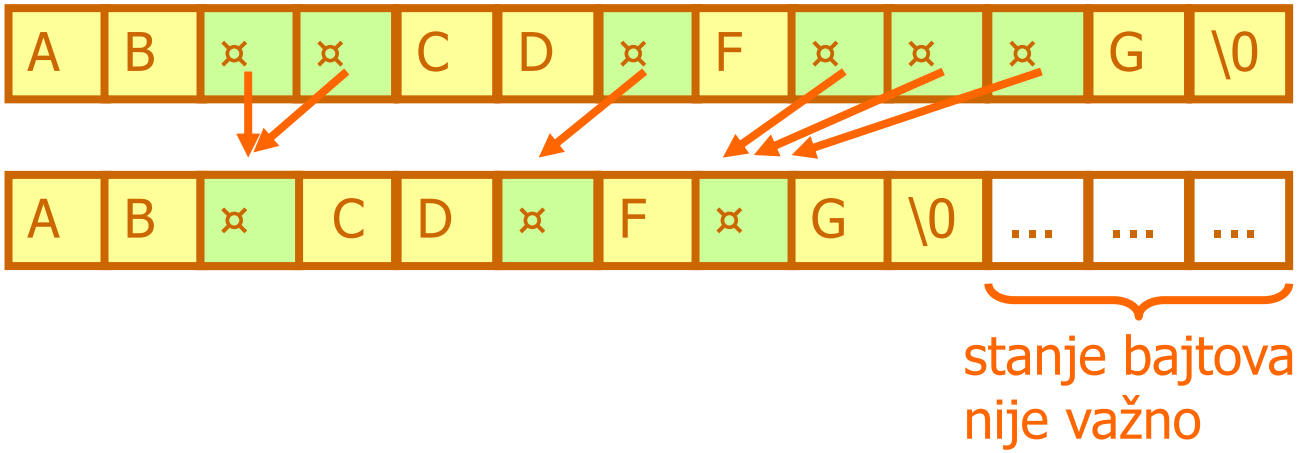
Programiranje procesora FRISC

Ostali primjeri

Ostali primjeri

Primjer:

Napisati potprogram SAZMI kojemu se preko R1 predaje adresa znakovnog niza u memoriji. Znakovi su zapisani ASCII kodom u bajtovima. Potprogram treba u znakovnom nizu sva uzastopna pojavljivanja razmaka sažeti u jednostruke razmake. Niz je zaključen ASCII-znakom NUL.



- "Slova" (žuti) se kopiraju
- Razmaci označeni sa ¤ (zeleni) se sažimaju

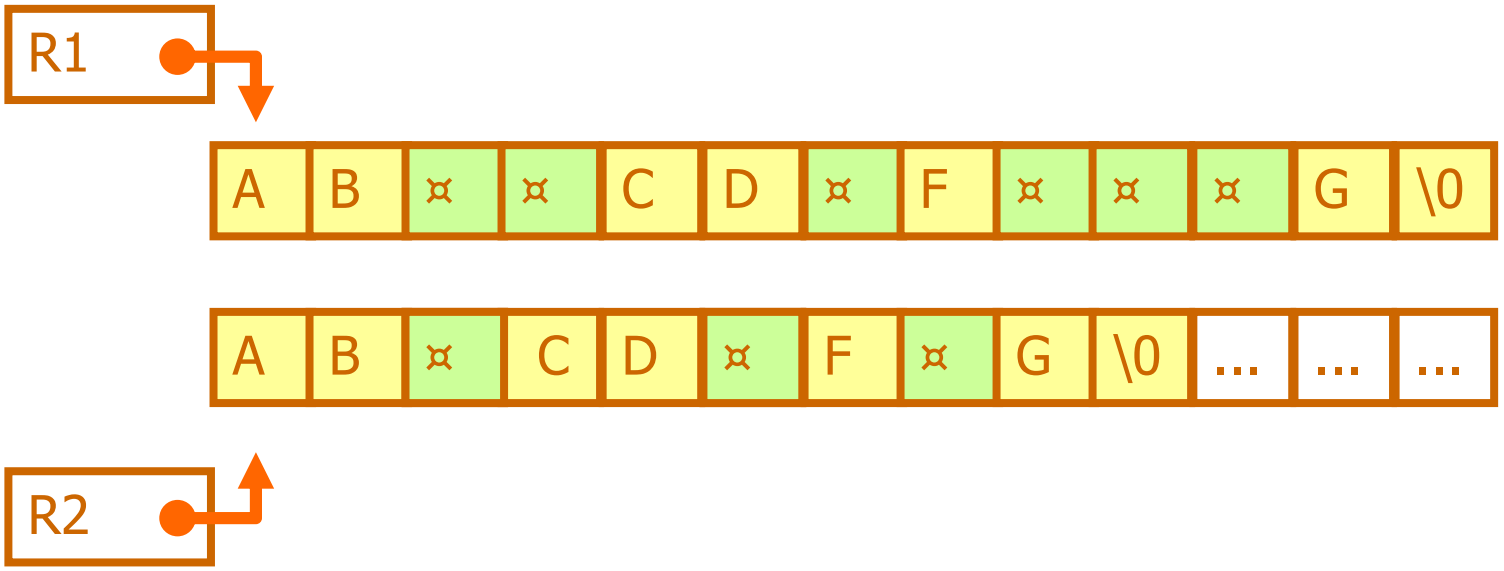
>>>>

Ostali primjeri

Rješenje:

Upotrijebiti dva pokazivača:

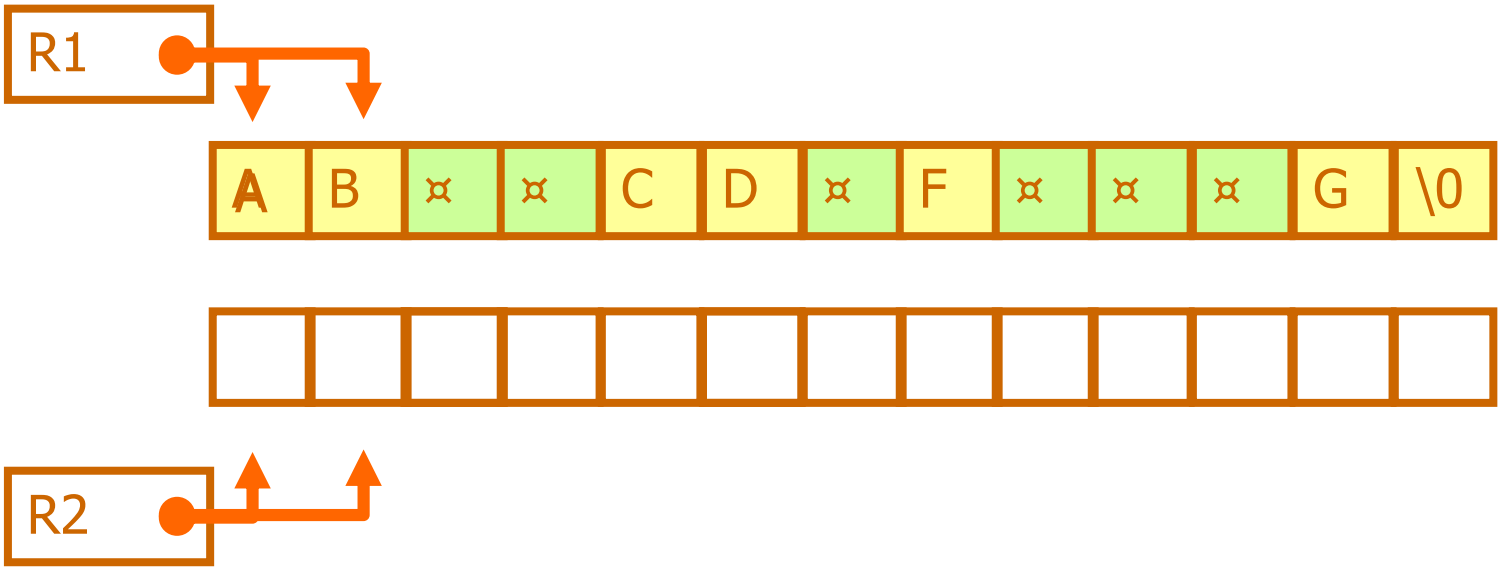
- jedan čita znakove izvornog niza (R1)
- drugi pokazuje mjesto na koji se kopira znak izvornog niza (R2)



>>>>

Ostali primjeri

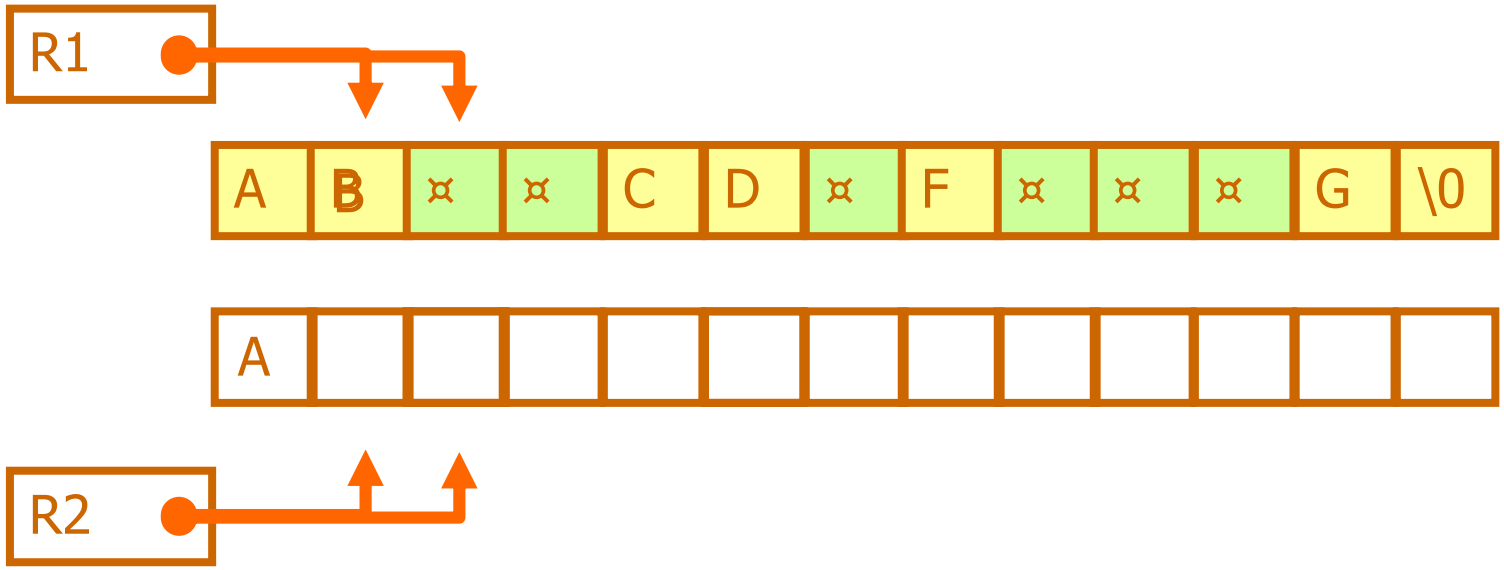
- Ako je "ispod" R1 slovo, onda
  - slovo se kopira tamo gdje pokazuje R2
  - R1 i R2 se pomiču za jedno mjesto



>>>>

Ostali primjeri

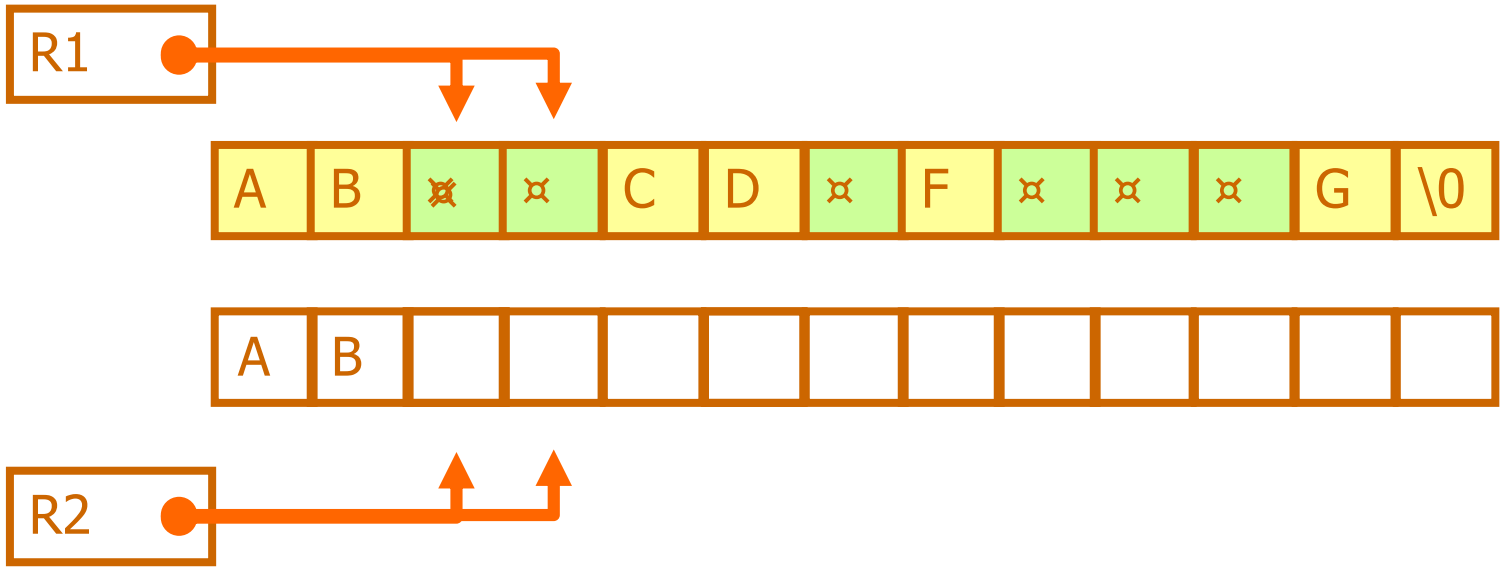
- Ako je "ispod" R1 slovo, onda
  - slovo se kopira tamo gdje pokazuje R2
  - R1 i R2 se pomiču za jedno mjesto



>>>>

Ostali primjeri

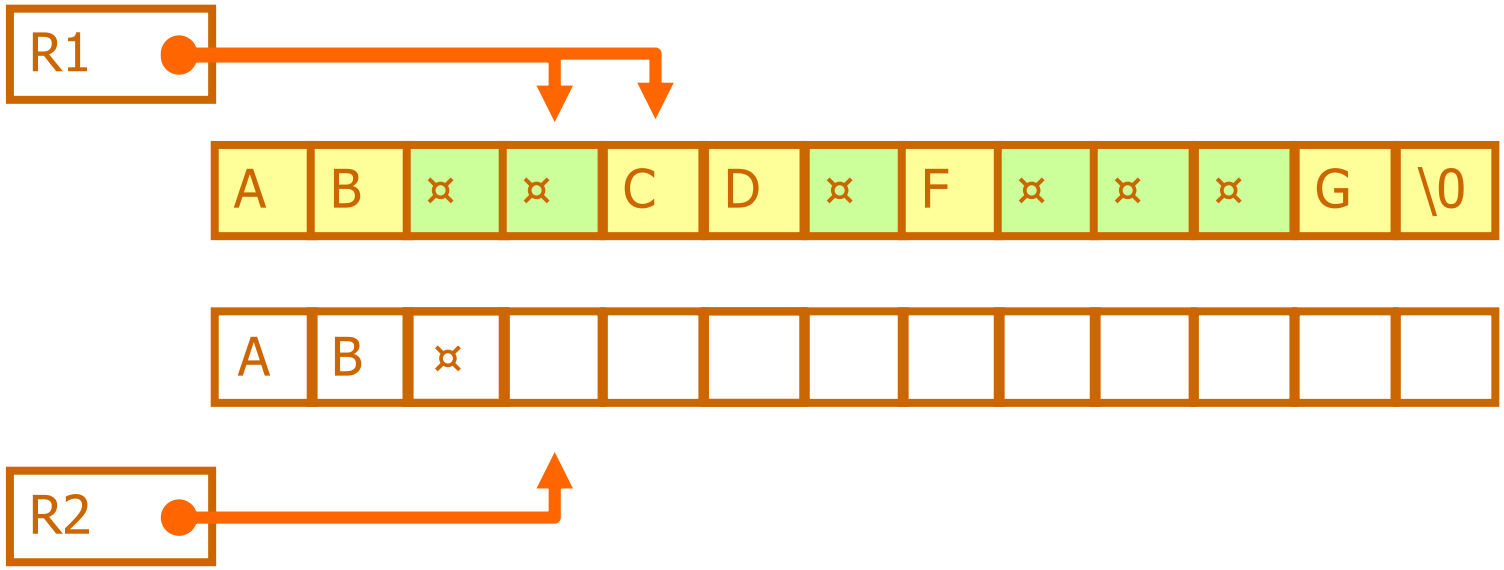
- Ako je "ispod" R1 razmak, onda
  - ako je prethodilo slovo, onda
    - trenutačni razmak se kopira
    - R1 i R2 se pomiču za jedno mjesto



>>>>

Ostali primjeri

- Ako je "ispod" R1 razmak, onda
  - ako je prethodio razmak, onda
    - trenutačni razmak se zanemaruje
    - R1 se pomiče za jedno mjesto

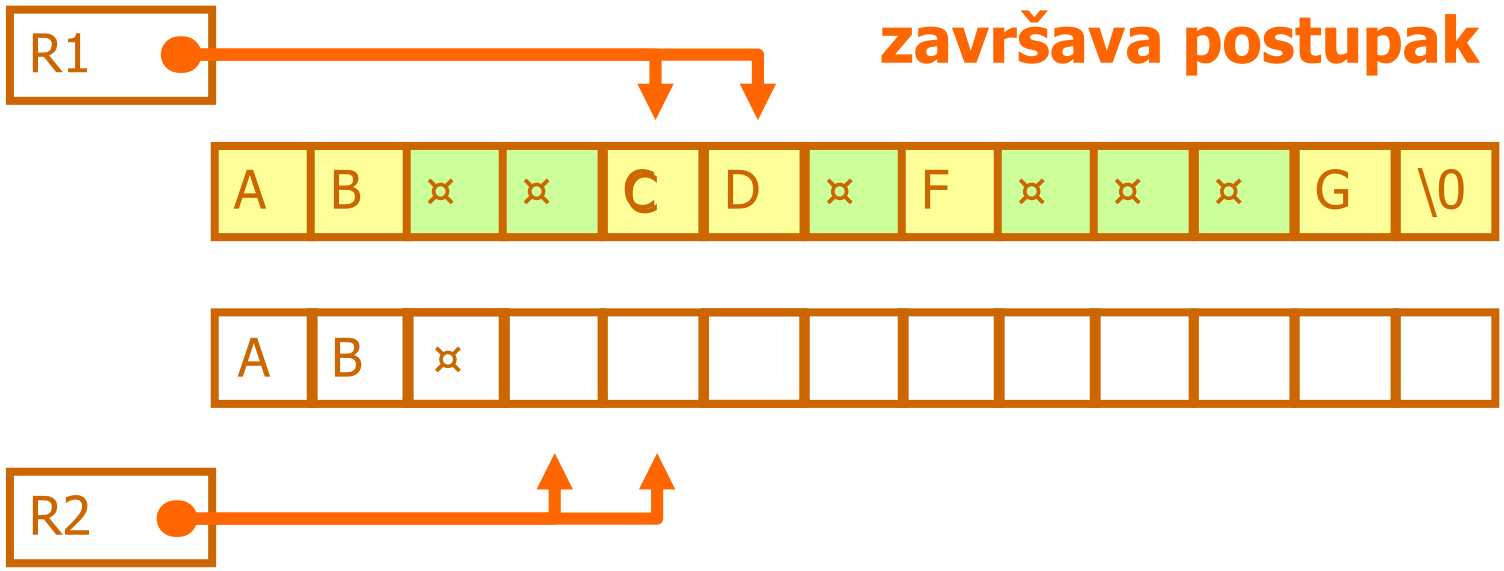


>>>>

Ostali primjeri

- Ako je "ispod" R1 slovo, onda
  - trenutačno slovo se kopira
  - R1 i R2 se pomiču za jedno mjesto

itd... do znaka NUL koji se zadnji kopira čime završava postupak



>>>>

Ostali primjeri

- U registru R0 pamtit ćemo prethodno stanje, tj. koji je bio prethodni znak:
  - SLO će značiti slovo
  - RAZ će značiti razmak

} SLO i RAZ će biti dvije labele koje ćemo definirati sa EQU

- Prilikom prelaska iz slova u razmak, mijenjat ćemo stanje u RAZ
- Prilikom prelaska iz razmaka u slovo, mijenjat ćemo stanje u SLO

- Početno stanje bit će SLO

- Registar R3 služit će samo za učitavanje trenutačnog znaka i njegovo ispitivanje

>>>>

```
<<<<     ; parametar R1 = adresa niza

SAZMI    PUSH   R0        ; Spremi registre
         PUSH   R1
         PUSH   R2
         PUSH   R3

         MOVE   SLO, R0 ; Početno: stanje = slovo
         MOVE   R1, R2  ; Inicijalizacija R2

PETLJA   LOADB  R3, (R1)   ; Učitaj znak ispod R1

         CMP    R3, 0       ; Ako je NUL, onda
         JR_Z   KRAJ        ; idi na kraj.

         ; JE LI TRENUTAČNI ZNAK RAZMAK ILI NIJE
         CMP    R3, 20      ; 20 = ASCII razmak
         JR_EQ  RAZMAK
```

>>>>

```
<<<<      ; TRENUTAČNI ZNAK JE SLOVO

          ; JE LI PRETHODNI BIO RAZMAK ILI NIJE
SLOVO     CMP     R0, RAZ
          JR_EQ   RAZMAK_PA_SLOVO

SLOVO_PA_SLOVO
          STOREB R3, (R2)    ; Kopiraj znak
          ADD     R1, 1, R1  ; Pomakni R1 i R2
          ADD     R2, 1, R2
          JR      PETLJA

RAZMAK_PA_SLOVO
          MOVE    SLO, R0    ; stanje = slovo
          STOREB R3, (R2)    ; Kopiraj znak
          ADD     R1, 1, R1  ; Pomakni R1 i R2
          ADD     R2, 1, R2
          JR      PETLJA                    >>>>
```

```
<<<<      ; TRENUTAČNI ZNAK JE RAZMAK

          ; JE LI PRETHODNI BIO RAZMAK ILI NIJE
RAZMAK    CMP     R0, RAZ
          JR_EQ   RAZMAK_PA_RAZMAK

SLOVO_PA_RAZMAK
          MOVE    RAZ, R0       ; stanje = razmak
          STOREB  R3, (R2)      ; Kopiraj znak
          ADD     R1, 1, R1     ; Pomakni R1 i R2
          ADD     R2, 1, R2
          JR      PETLJA

RAZMAK_PA_RAZMAK
          ADD     R1, 1, R1     ; Pomakni R1
          JR      PETLJA
```

>>>>

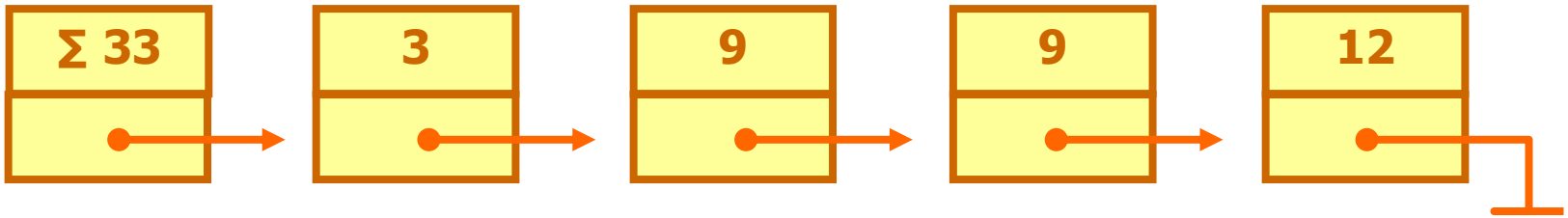
<<<<

```
KRAJ    STOREB R3, (R2)  ; Kopiraj NUL-znak

        POP    R3        ; Obnovi registre
        POP    R2
        POP    R1
        POP    R0

        RET


; definiranje "konstanti"

SLO     EQU    0
RAZ     EQU    1
```

Ostali primjeri

Primjer:

Jednostruko povezana lista ima čvorove koji u memoriji zauzimaju dvije 32-bitne riječi: prva riječ sadrži NBC-broj (koji predstavlja vrijednost čvora), a druga riječ je pokazivač na sljedeći čvor liste (tj. sadrži adresu sljedećeg čvora). Čvorovi su sortirani prema svojoj vrijednosti.



Prvi čvor liste ima specijalno značenje i u njemu se pamti zbroj svih vrijednosti preostalih čvorova liste. Prvi čvor je uvijek prisutan, bez obzira postoje li ostali čvorovi. Pretpostavka je da u zbroju nikada neće doći do prekoračenja opsega.

Zadnji čvor u listi prepoznaje se po NULL-pokazivaču (tj. lokacija s pokazivačem sljedećeg čvora sadrži nulu).

>>>>

Ostali primjeri

<<<<

Treba napisati potprogram UBACI koji ubacuje novi čvor u postojeću sortiranu listu tako da ona ostane sortirana.

Parametri potprograma su adresa prvog elementa liste i adresa novog čvora. Parametri se šalju preko stoga.

Povratna vrijednost je novi zbroj iz prvoga čvora liste. Vrijednost se vraća pomoću R0.

Rješenje:

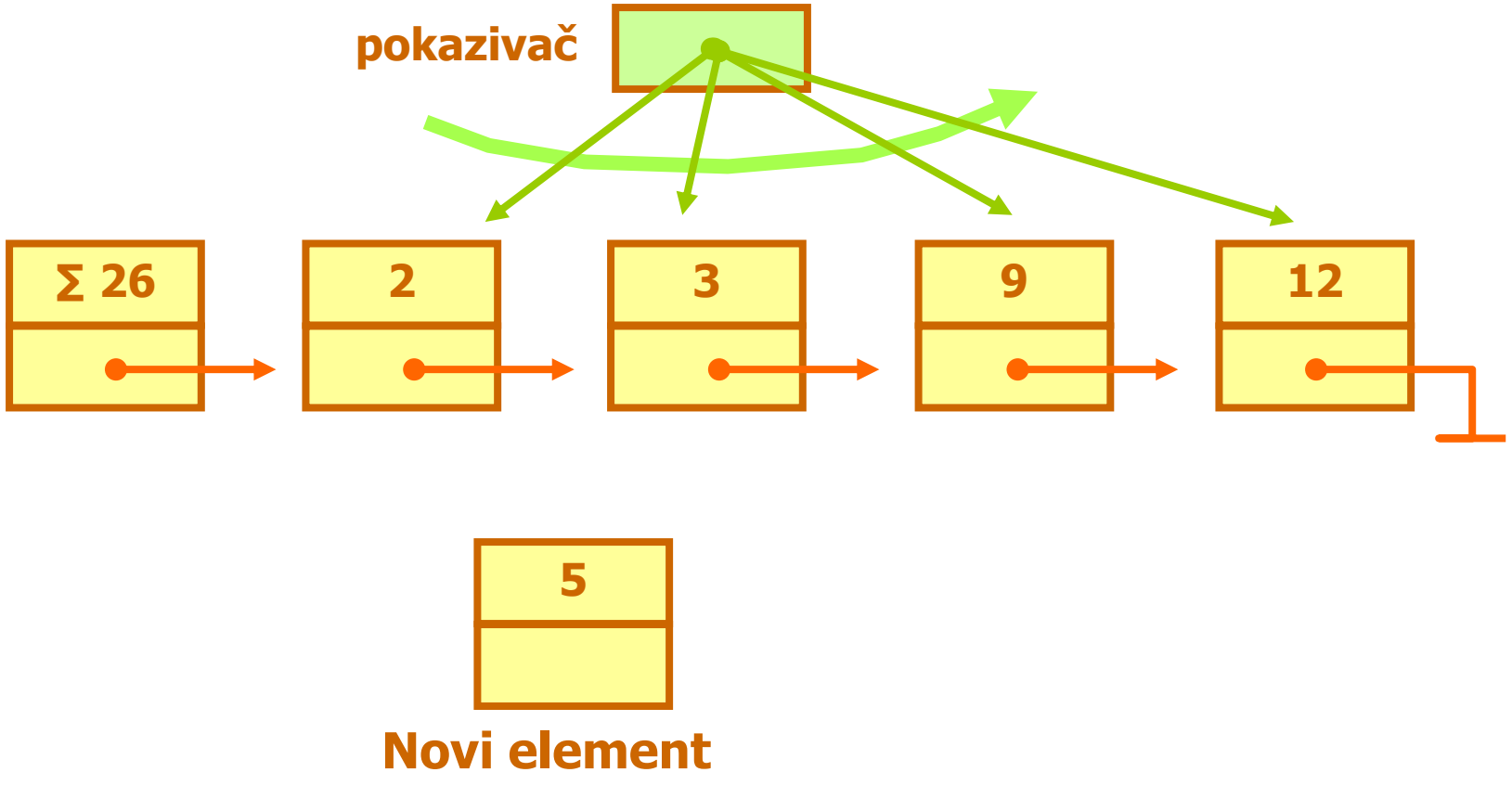
Potprogram će čuvati vrijednosti registara, a parametre će sa stoga uklanjati glavni program.

>>>>

Ostali primjeri

<<<< Idejno rješenje:

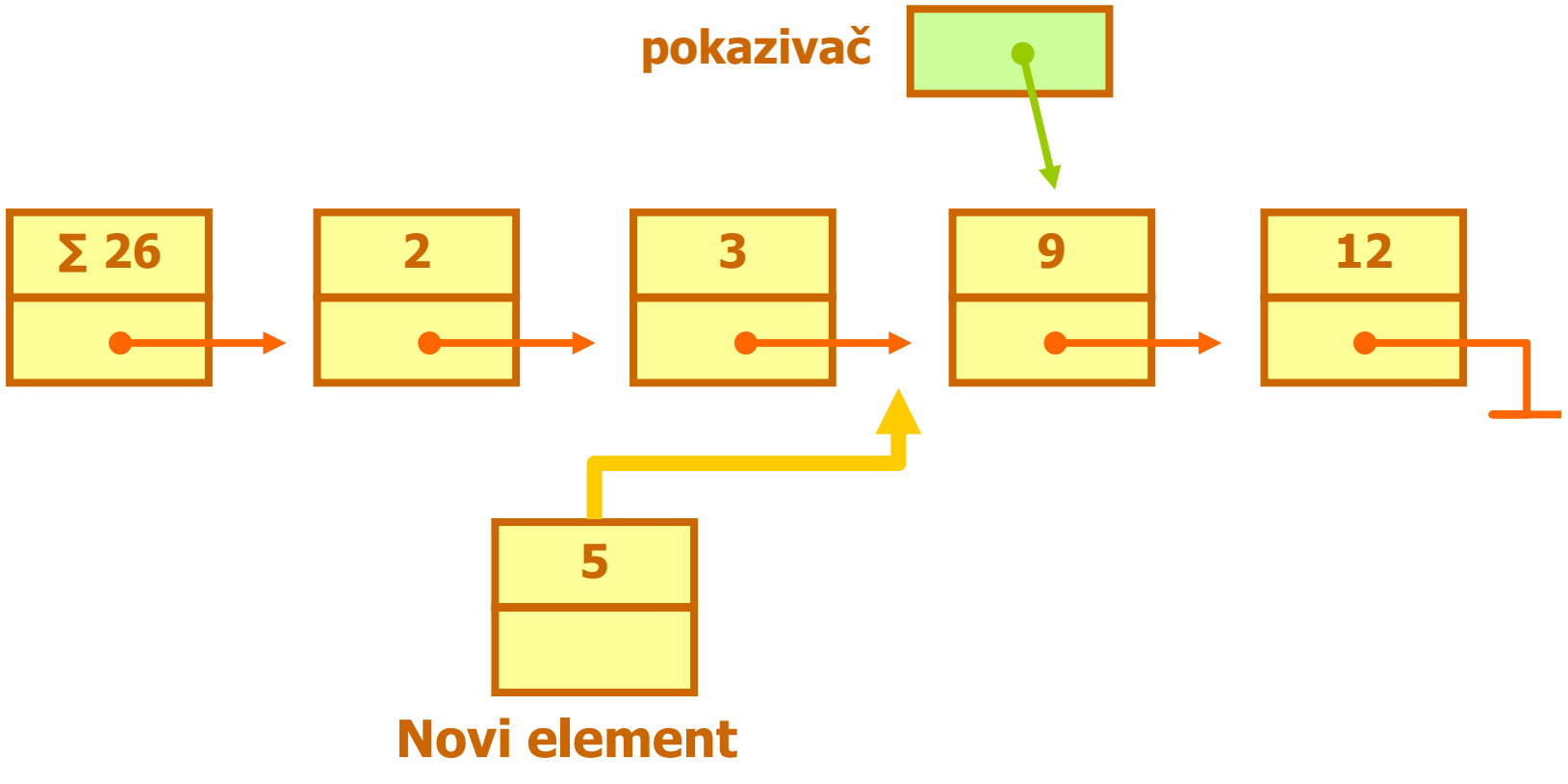
- Lista se pretražuje dok se ne nađe mjesto za ubacivanje.
- Jednim pokazivačem krećemo se po čvorovima koje ispitujemo.



Ostali primjeri

<<<< Idejno rješenje:

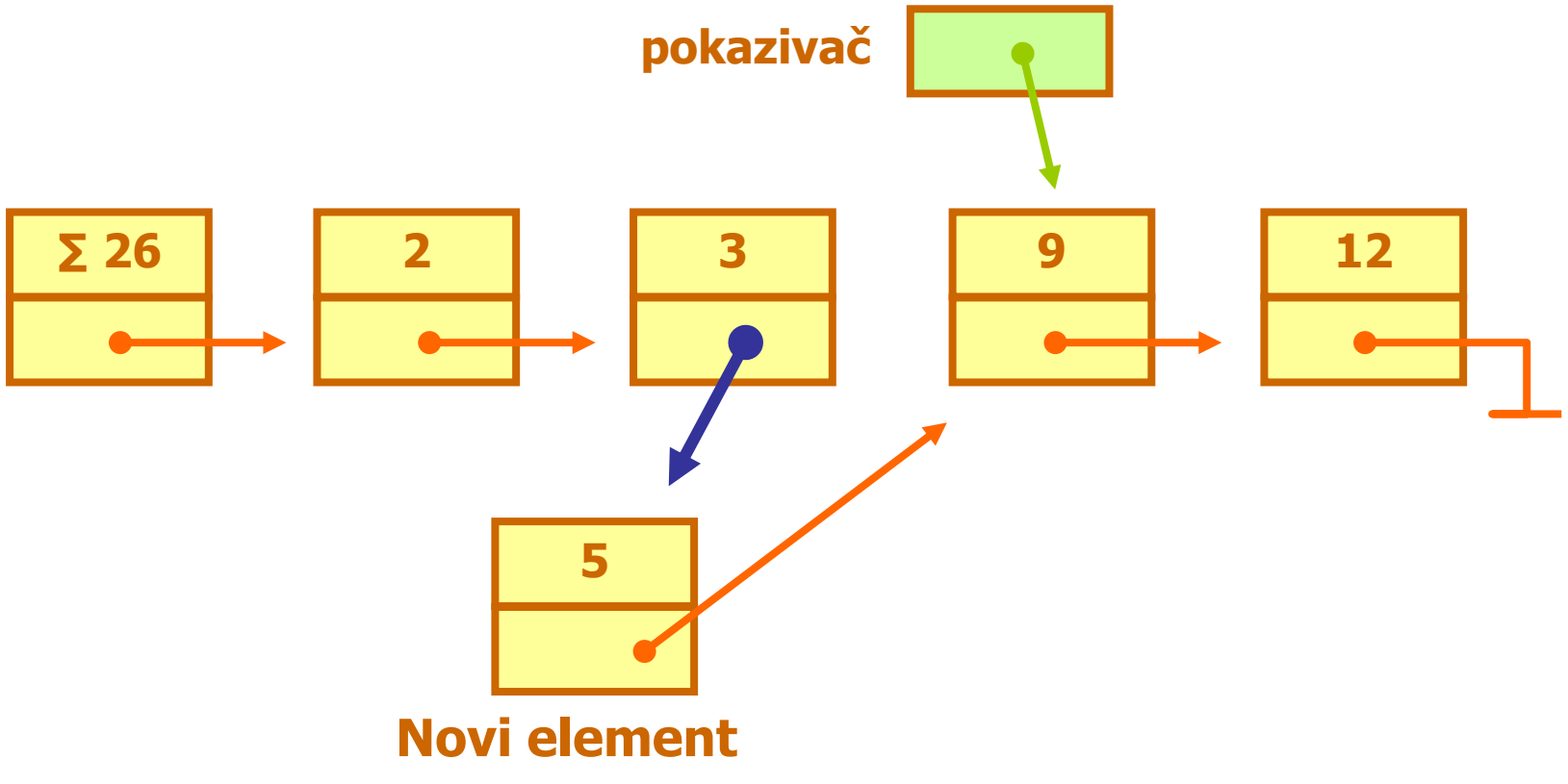
- Mjesto za ubacivanje je ISPRED prvog čvora koji ima veću ili jednaku vrijednost novom čvoru.



Ostali primjeri

<<<< Idejno rješenje:

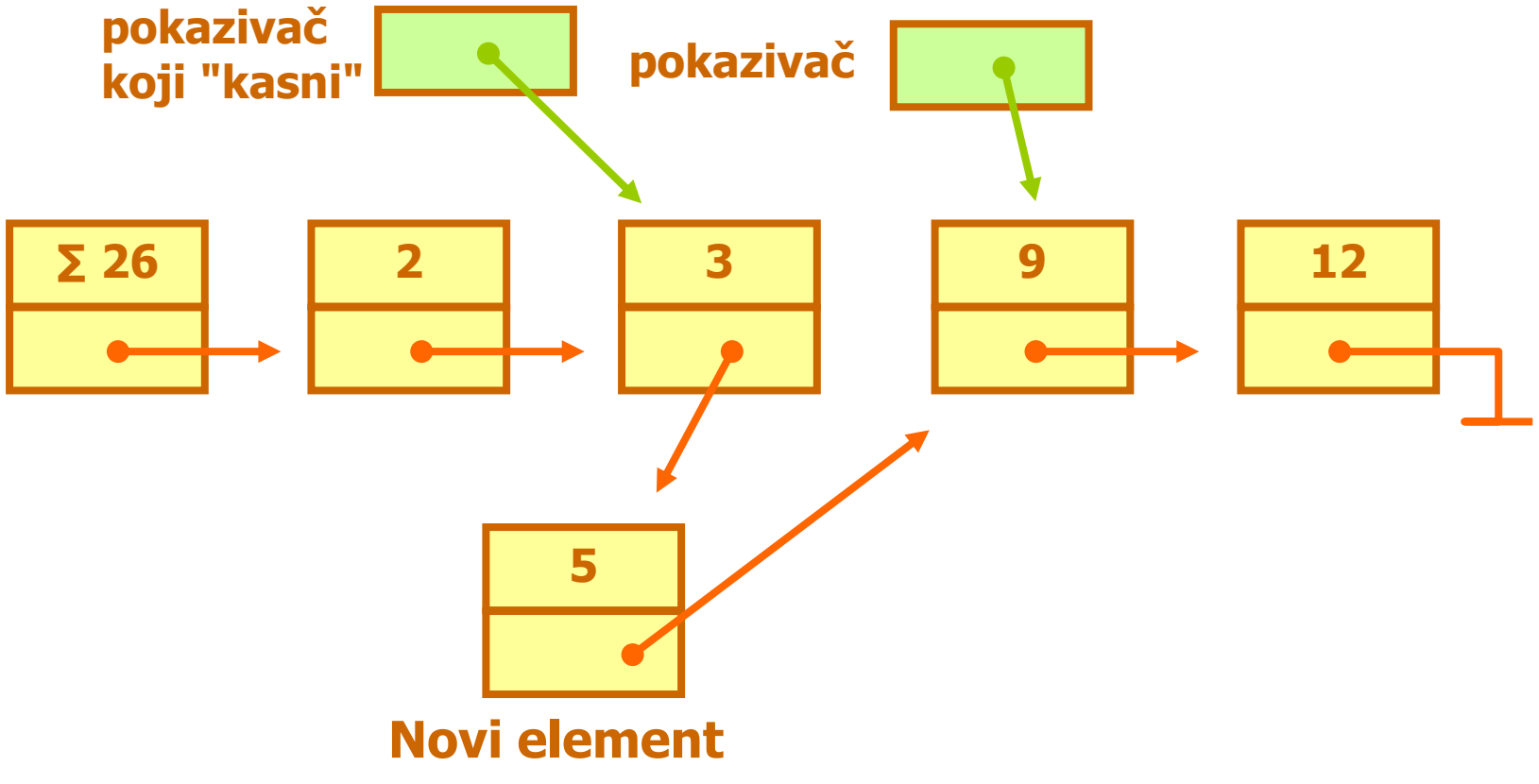
- Budući da se čvor mora ubaciti ISPRED čvora na kojeg pokazuje pokazivač, onda ne možemo dohvatiti prethodni čvor u kojem treba promijeniti pokazivač na sljedeći (plava strelica)



Ostali primjeri

<<<< Idejno rješenje:

- Zato trebamo još jedan pokazivač koji će uvijek "kasniti" za jedan čvor u odnosu na pokazivač ispitivanog čvora, tj. pokazivač će pokazivati jedan čvor ispred onog čvora kojeg ispitujemo

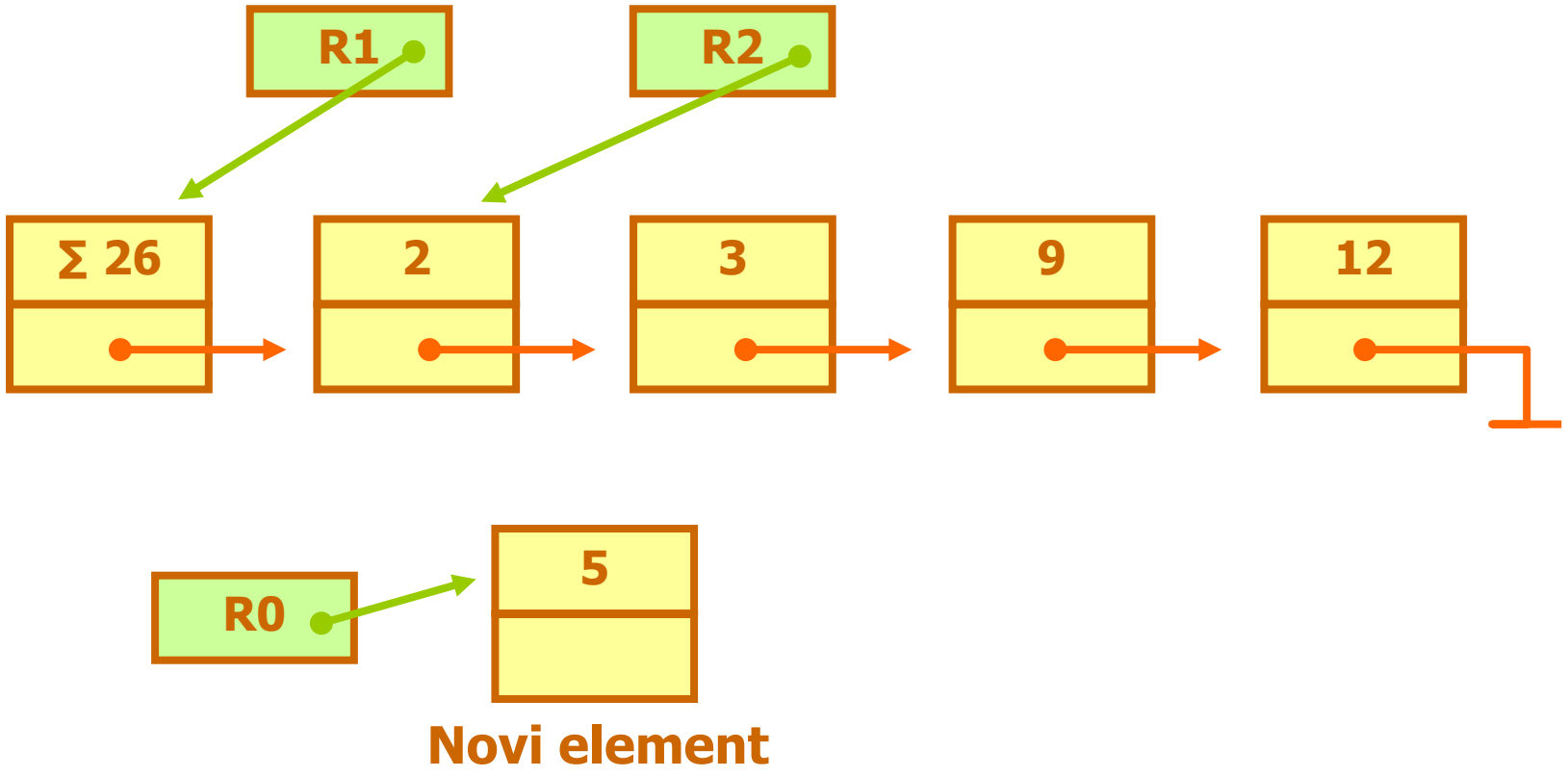


Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)

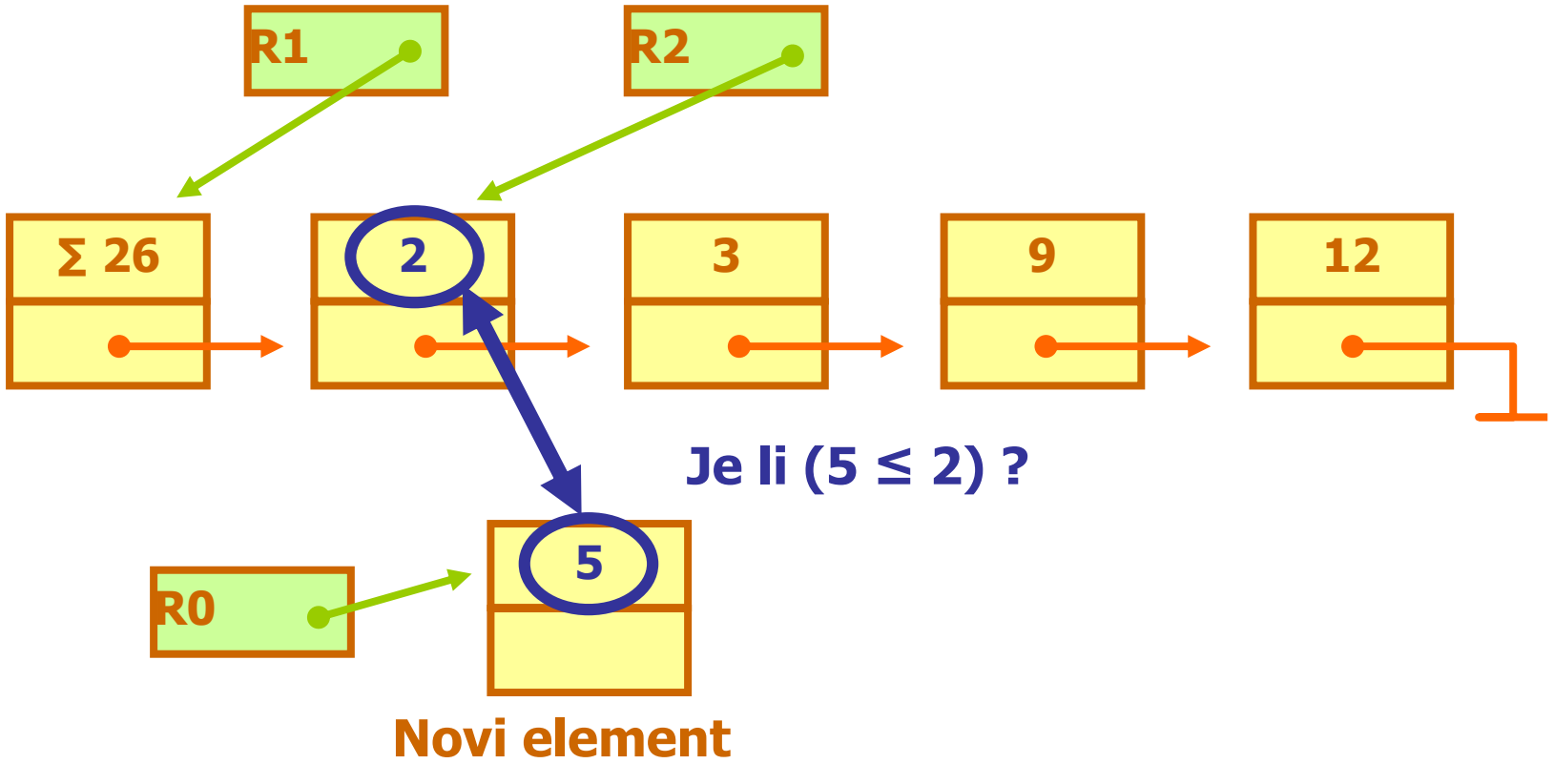


Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)

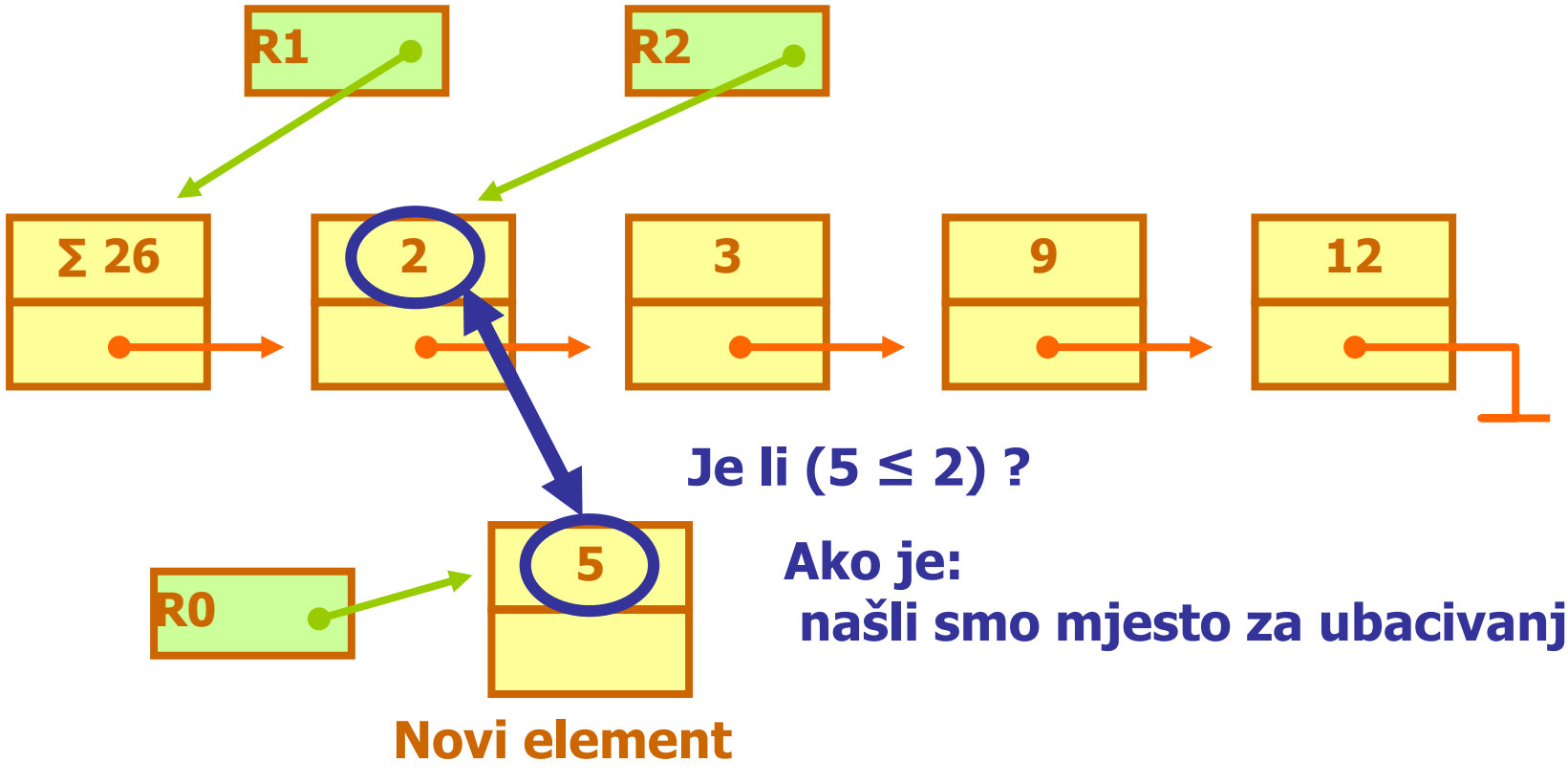


Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)

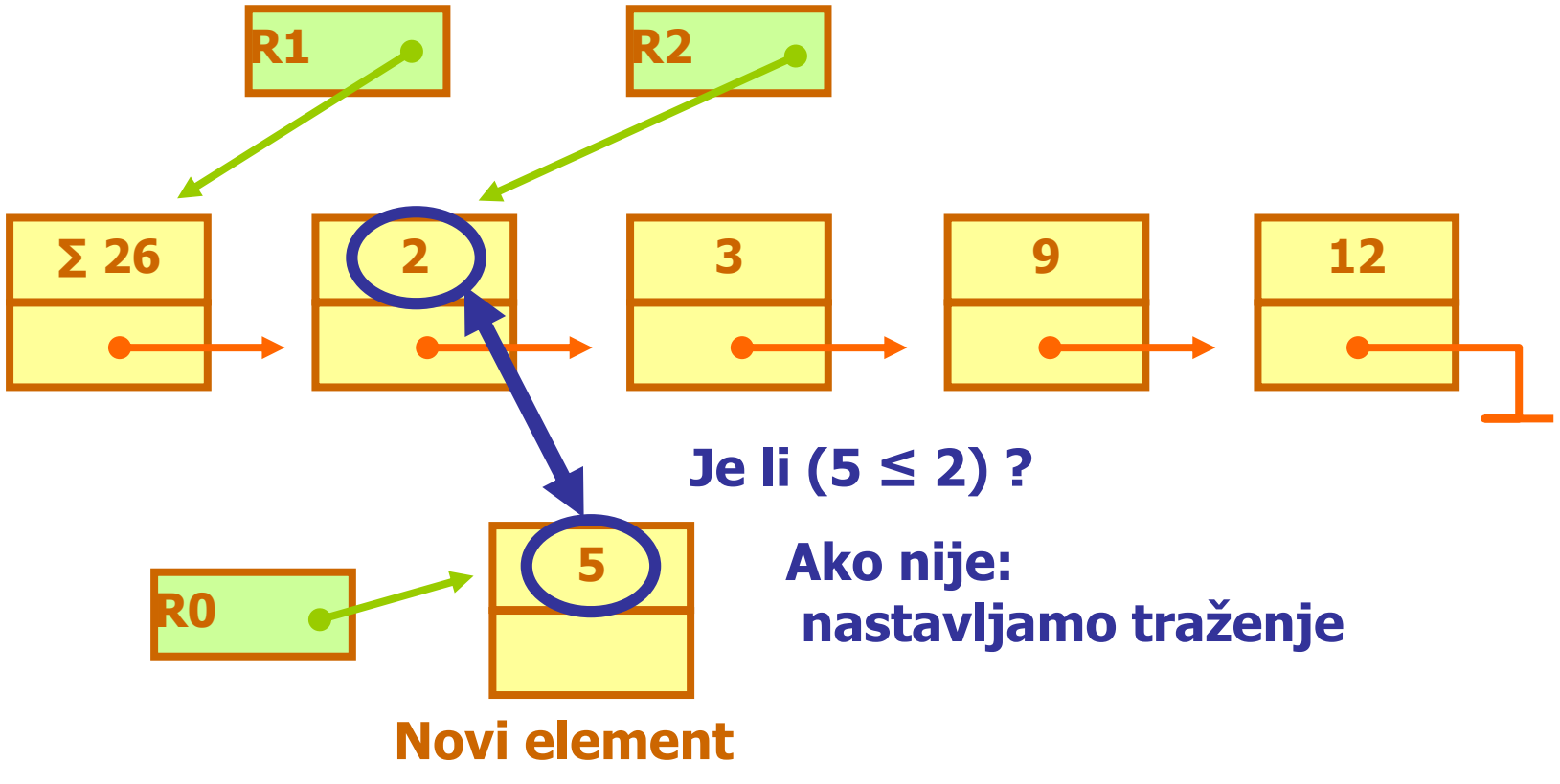


Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)

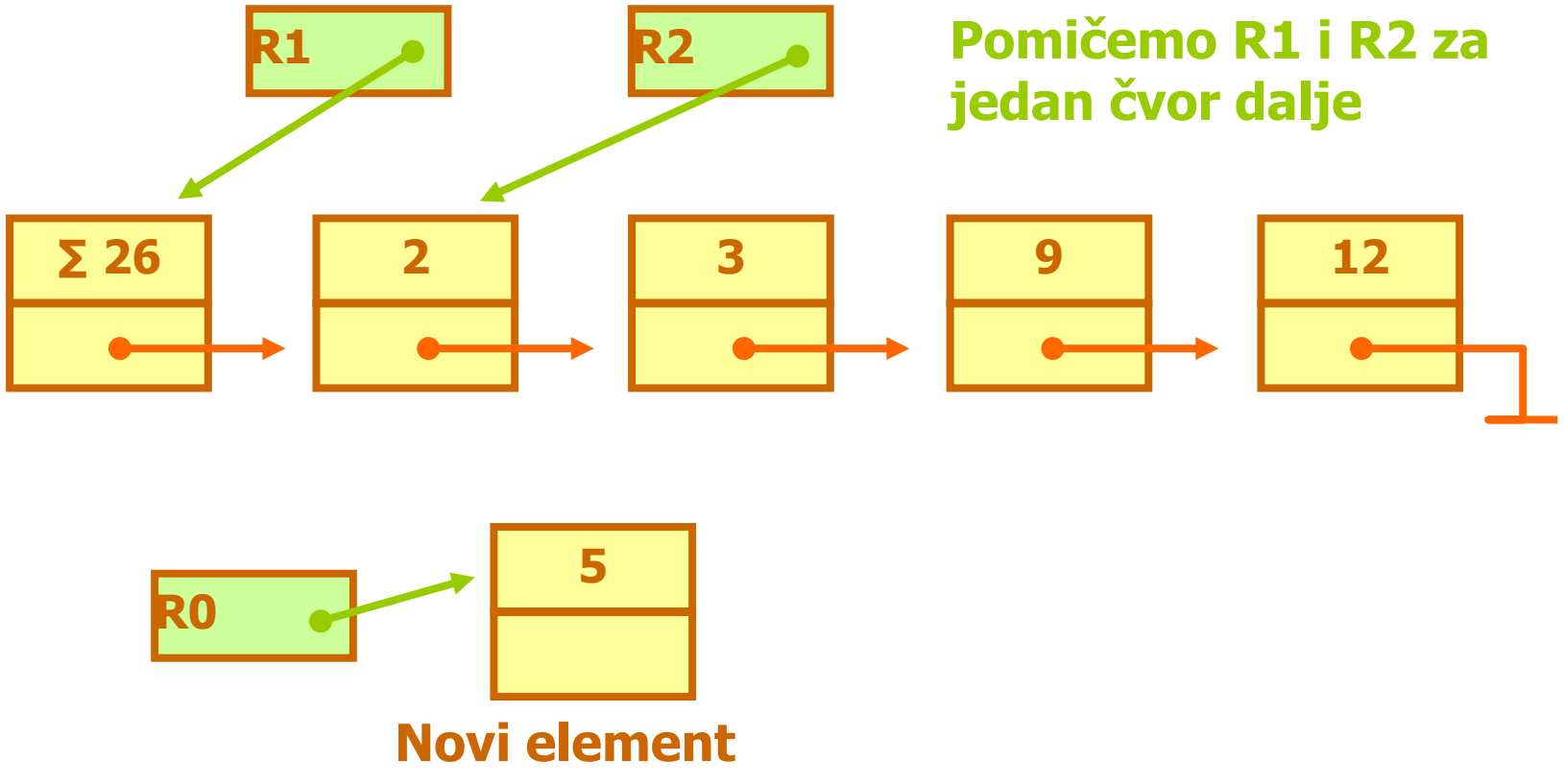


Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)

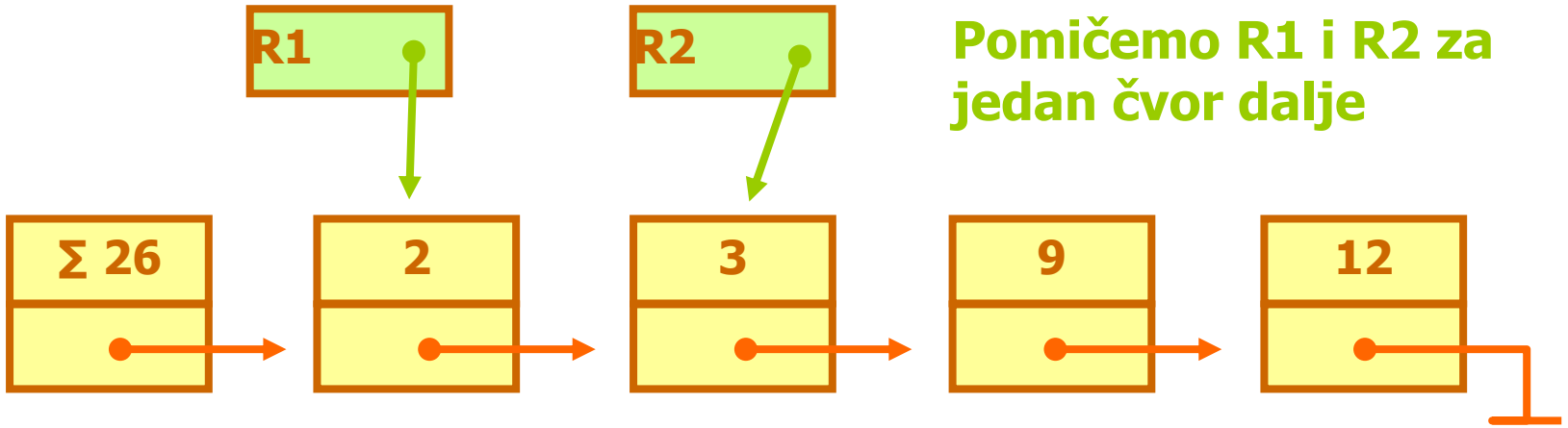


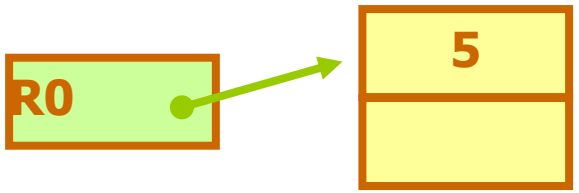
Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)





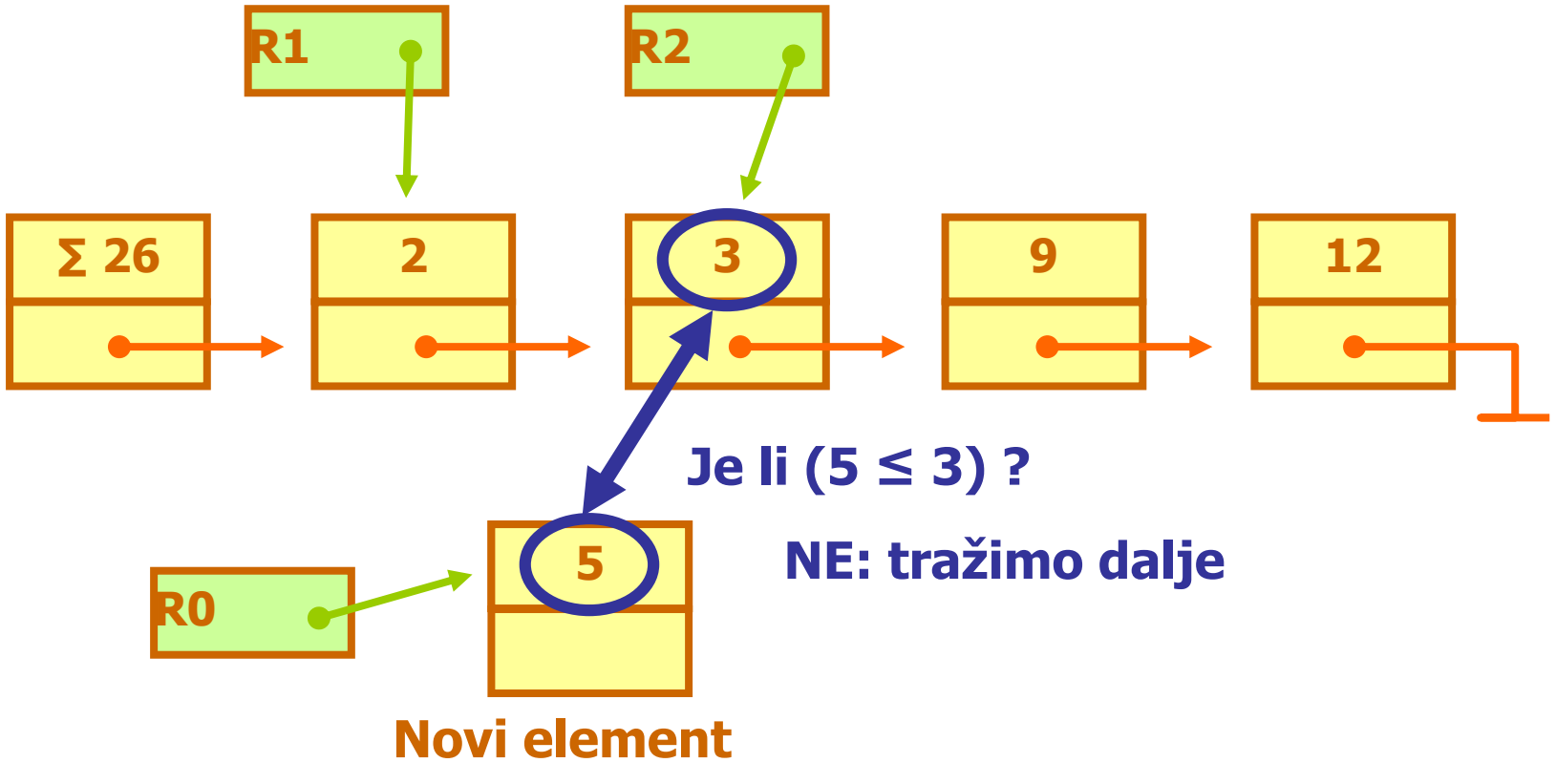
Novi element

Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)

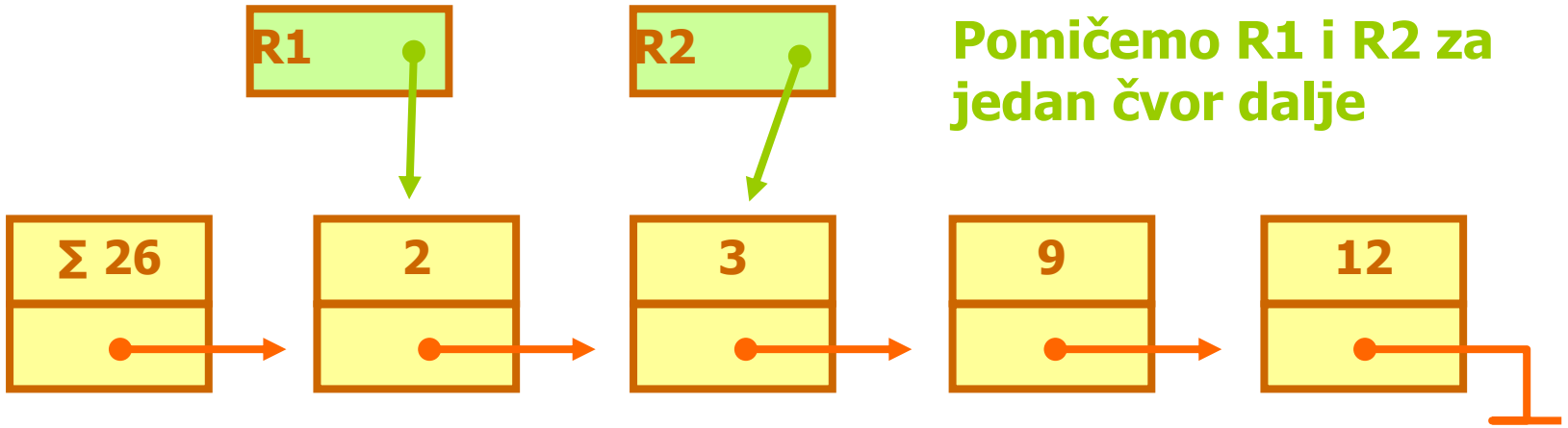


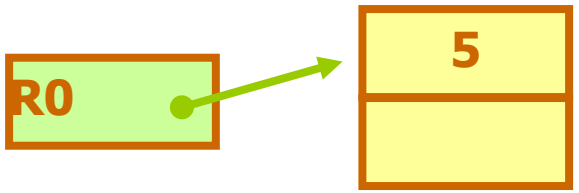
Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)





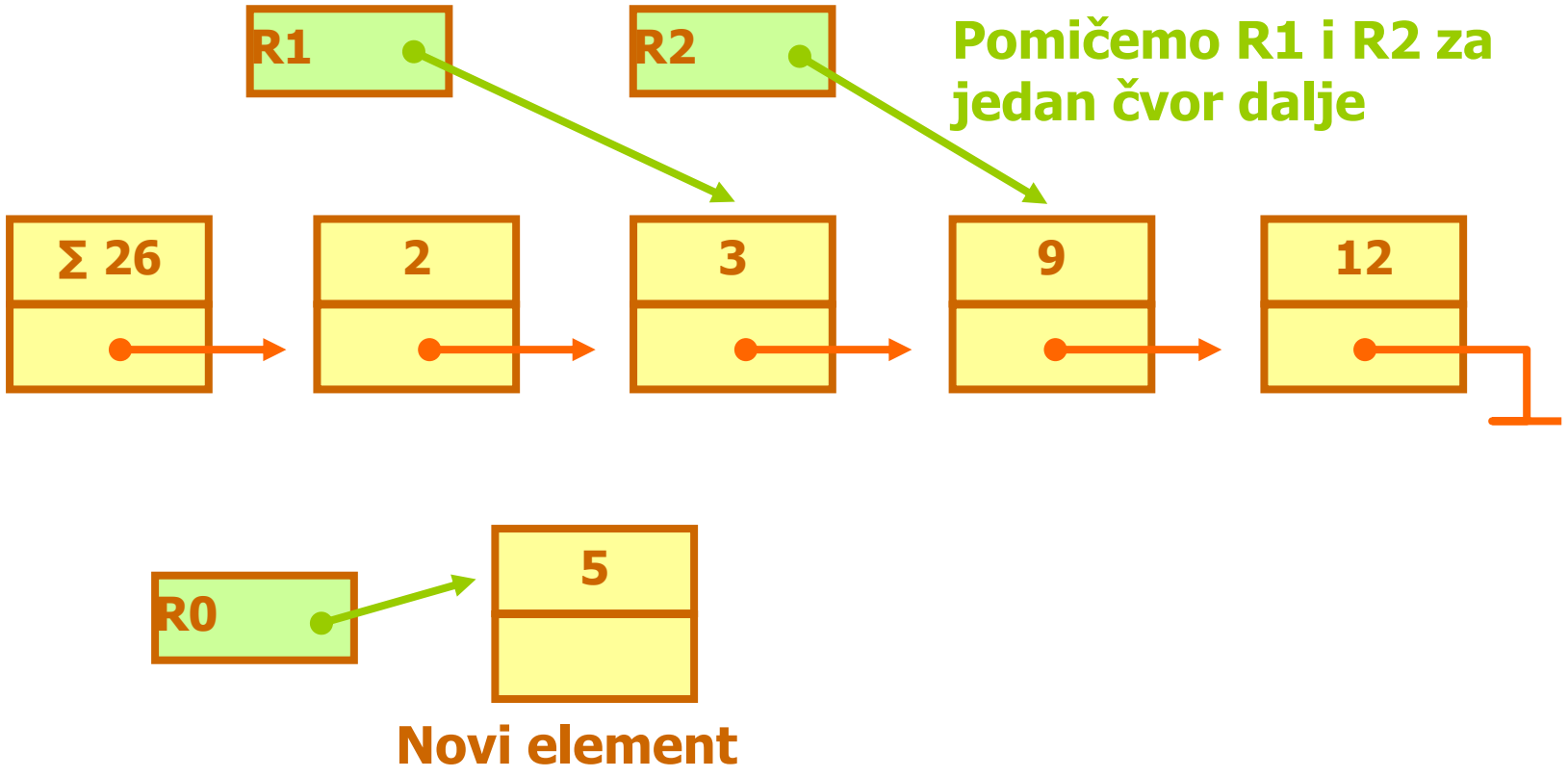
Novi element

Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)

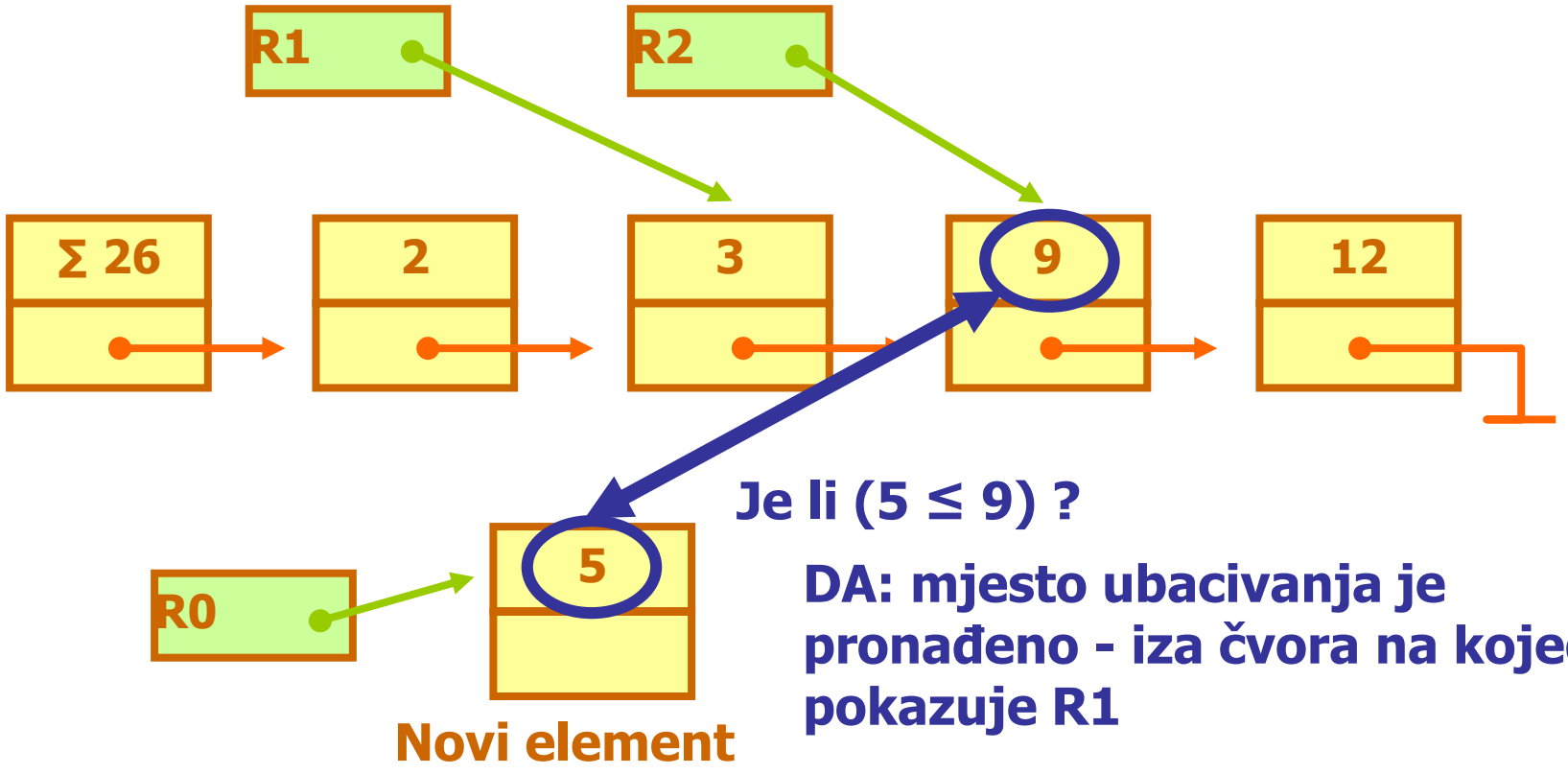


Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)

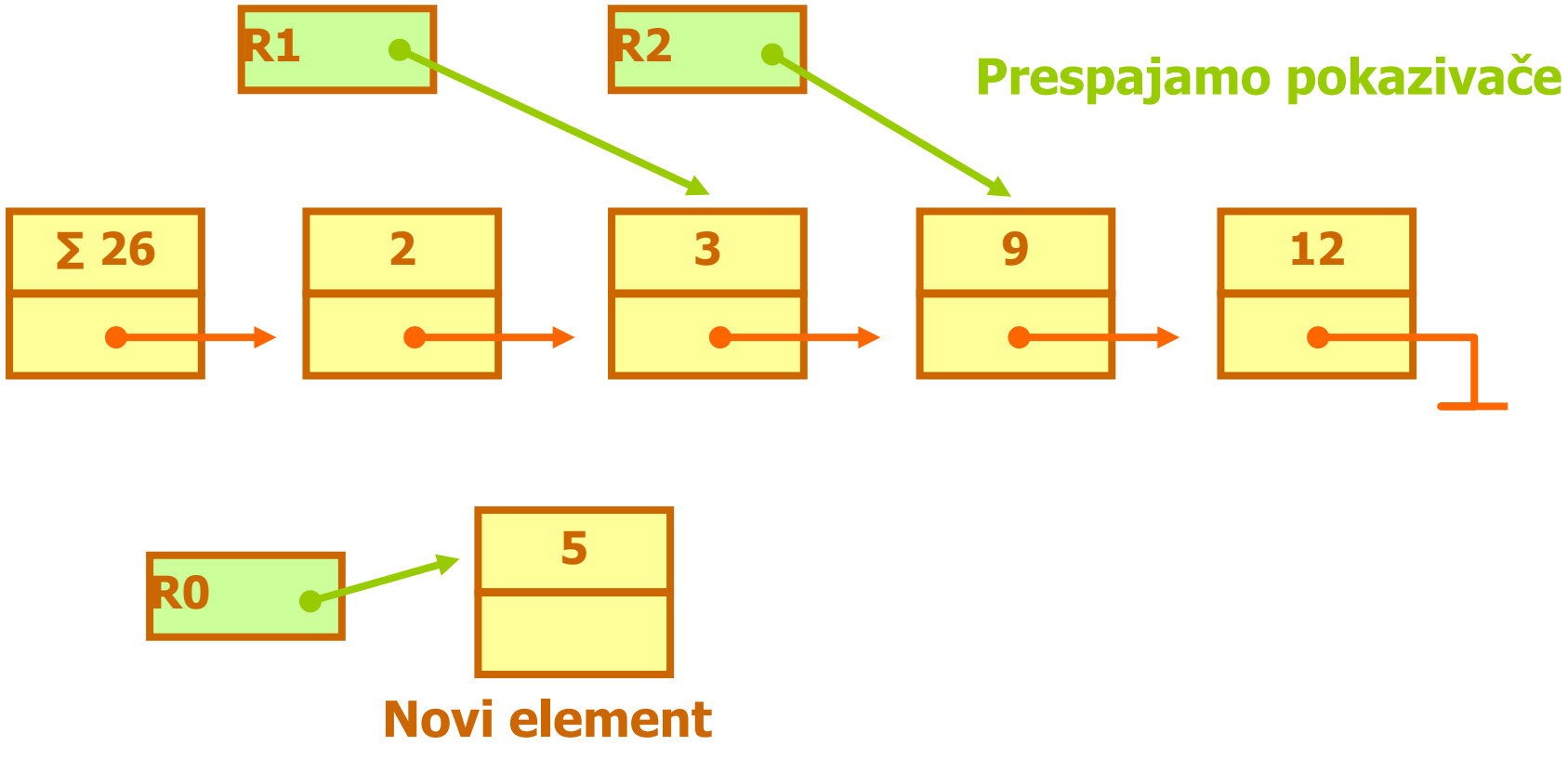


Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)

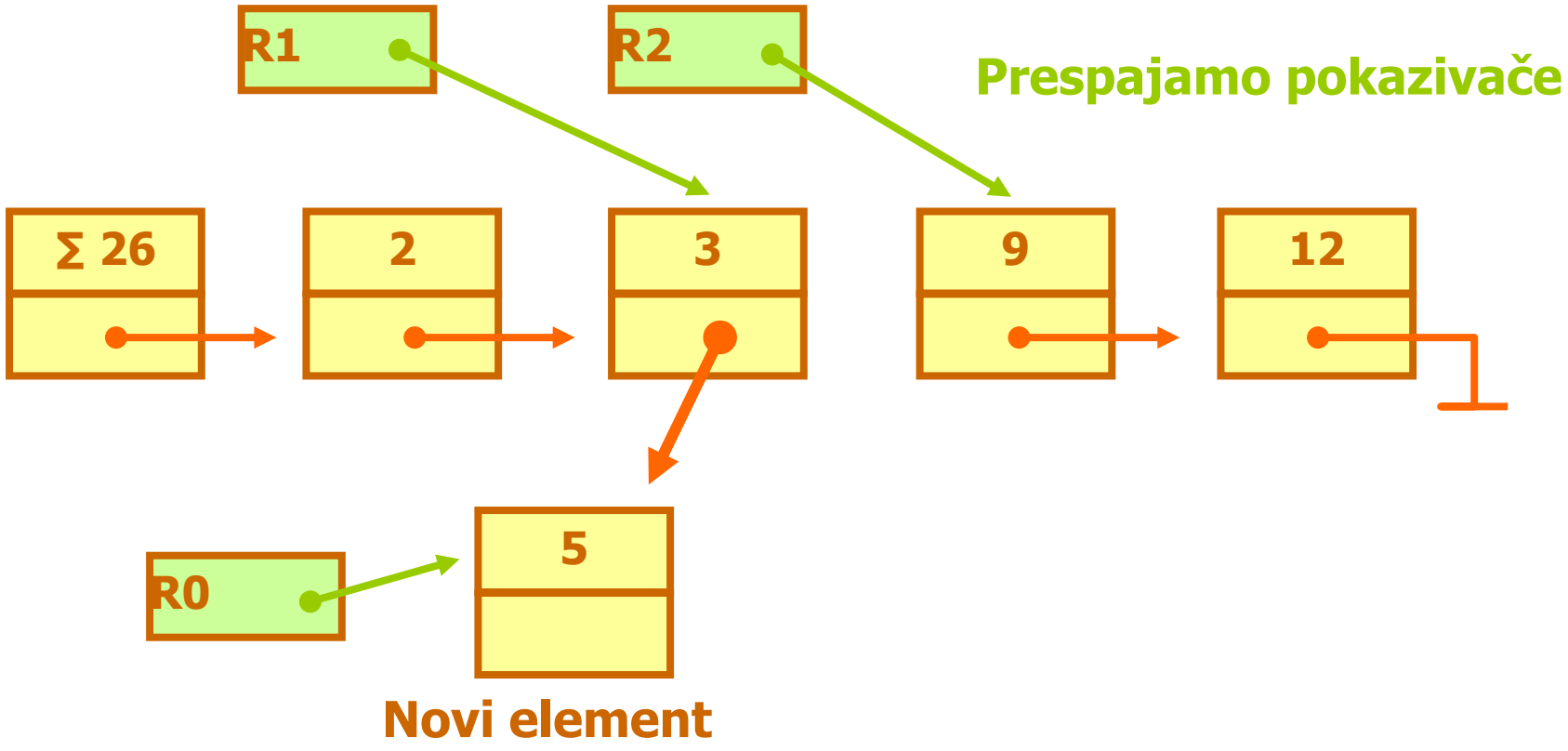


Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)

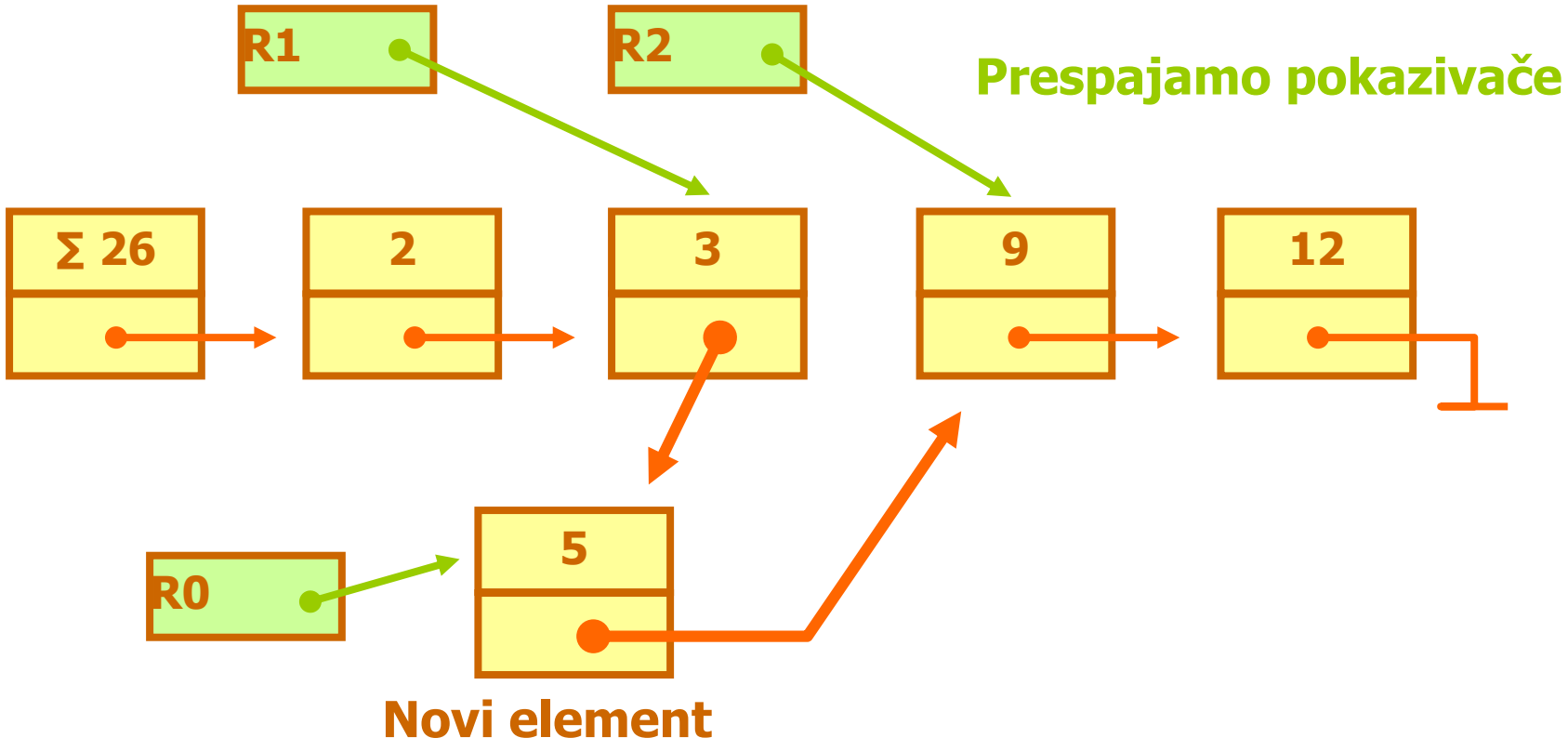


Ostali primjeri

Način rada potprograma:

Pokazuje čvor iza kojeg se ubacuje novi čvor (početno na prvi čvor)

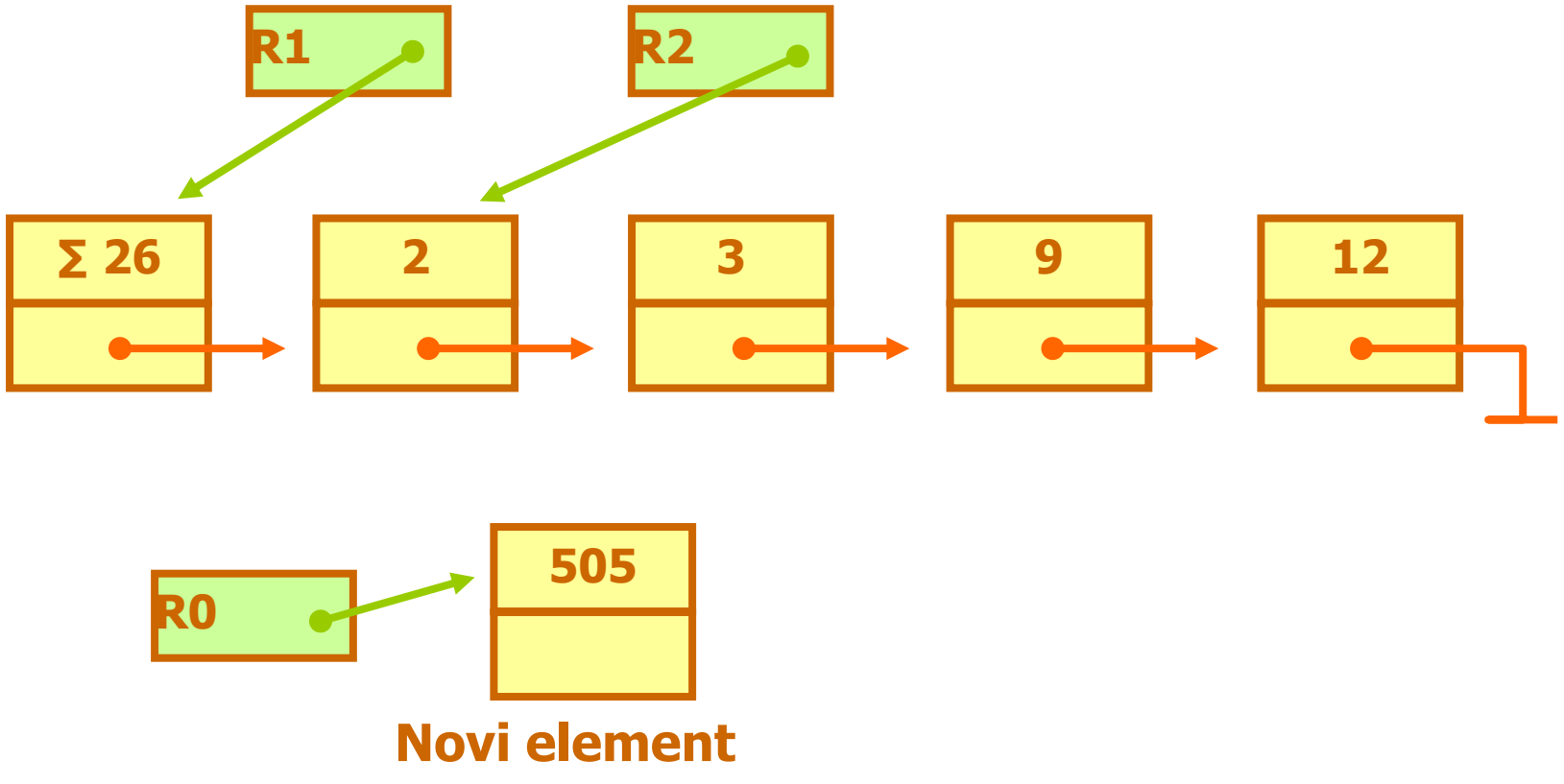
Pokazuje čvor čiju vrijednost uspoređujemo s novim čvorom (jedan čvor dalje od R1)



Ostali primjeri

Način rada potprograma:

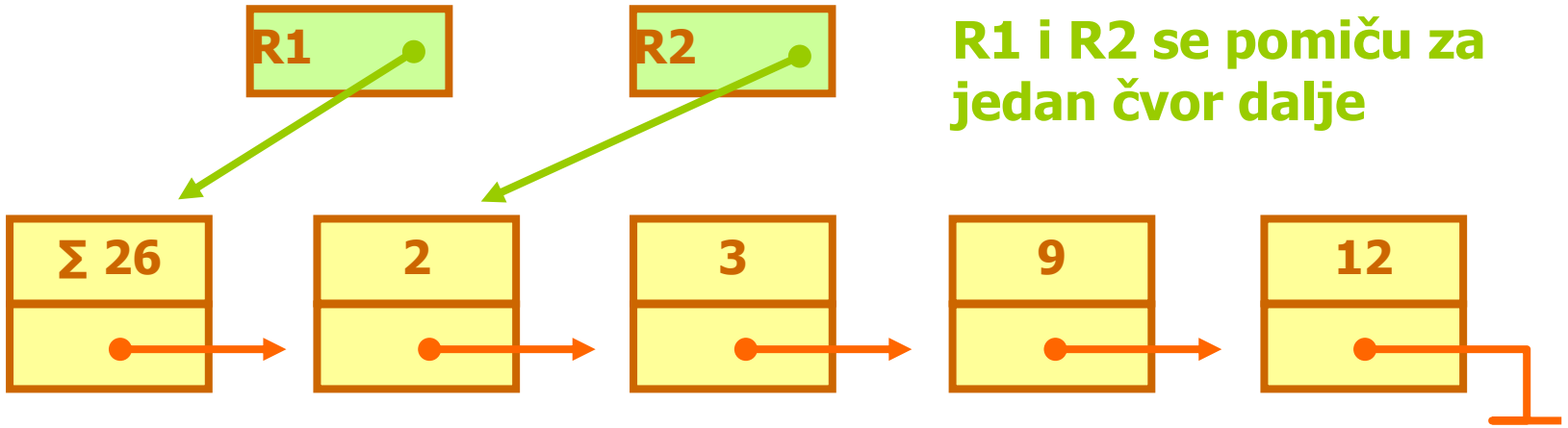
Pogledajmo još slučaj kad su u listi svi elementi manji od novog člana (ili je lista prazna)

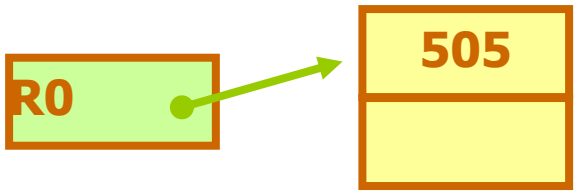


Ostali primjeri

Način rada potprograma:

Pogledajmo još slučaj kad su u listi svi elementi manji od novog člana (ili je lista prazna)

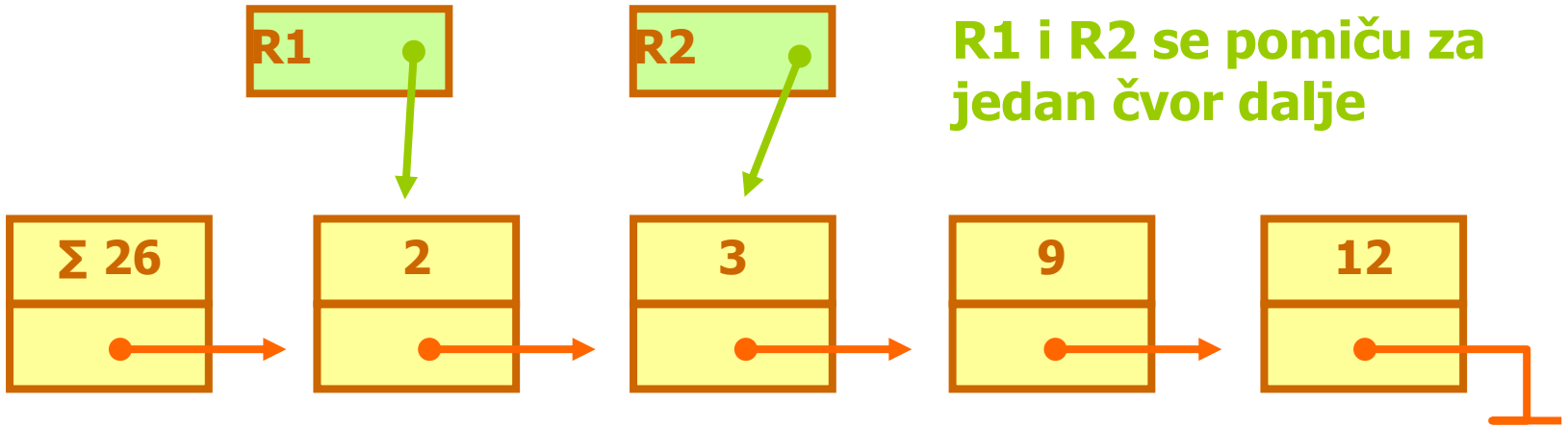


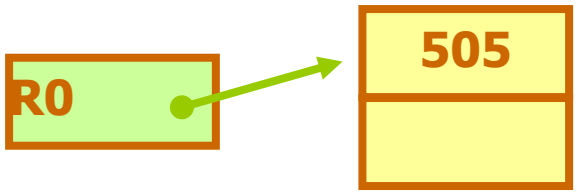


Novi element

Ostali primjeri

Pogledajmo još slučaj kad su u listi svi elementi manji od novog člana (ili je lista prazna)



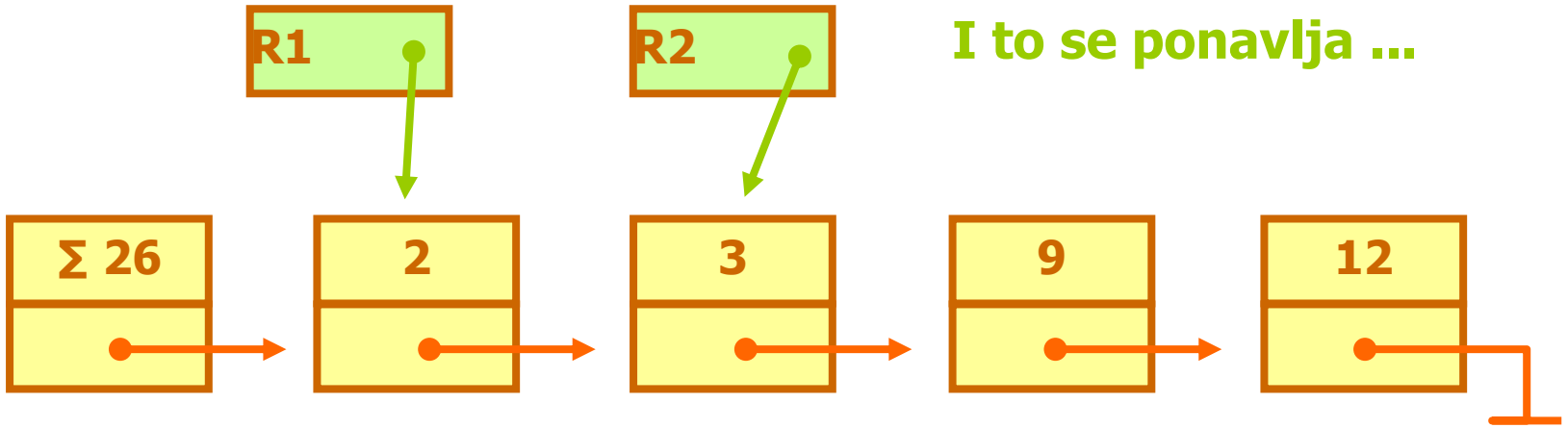


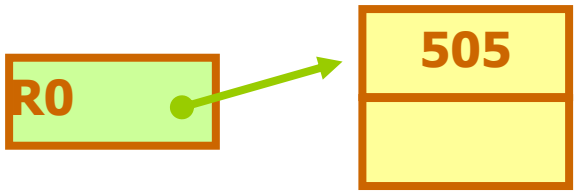
Novi element

Ostali primjeri

Način rada potprograma:

Pogledajmo još slučaj kad su u listi svi elementi manji od novog člana (ili je lista prazna)



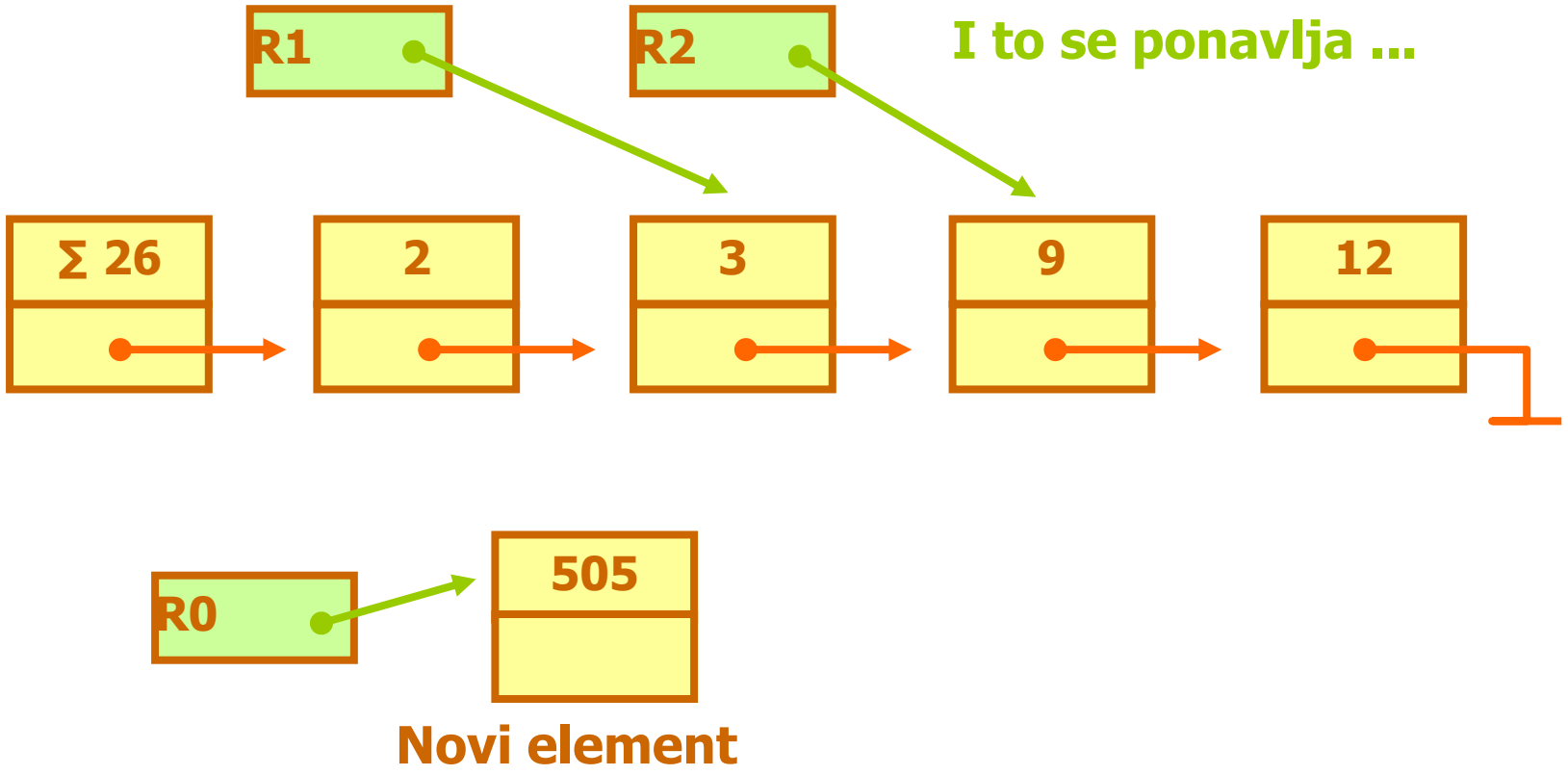


Novi element

Ostali primjeri

Način rada potprograma:

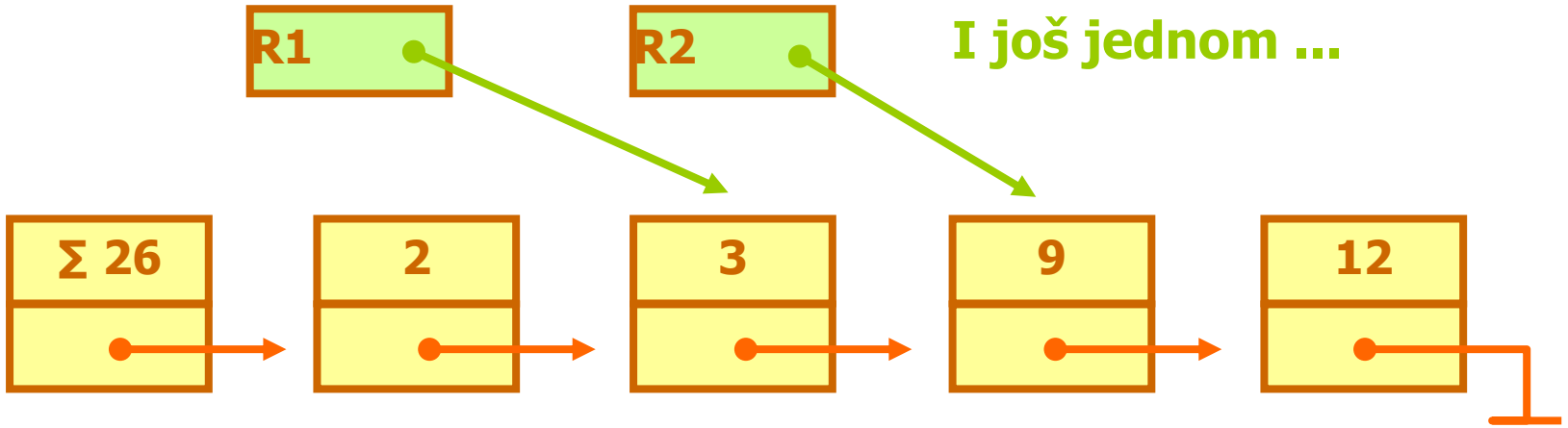
Pogledajmo još slučaj kad su u listi svi elementi manji od novog člana (ili je lista prazna)

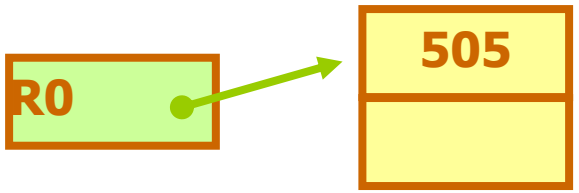


Ostali primjeri

Način rada potprograma:

Pogledajmo još slučaj kad su u listi svi elementi manji od novog člana (ili je lista prazna)



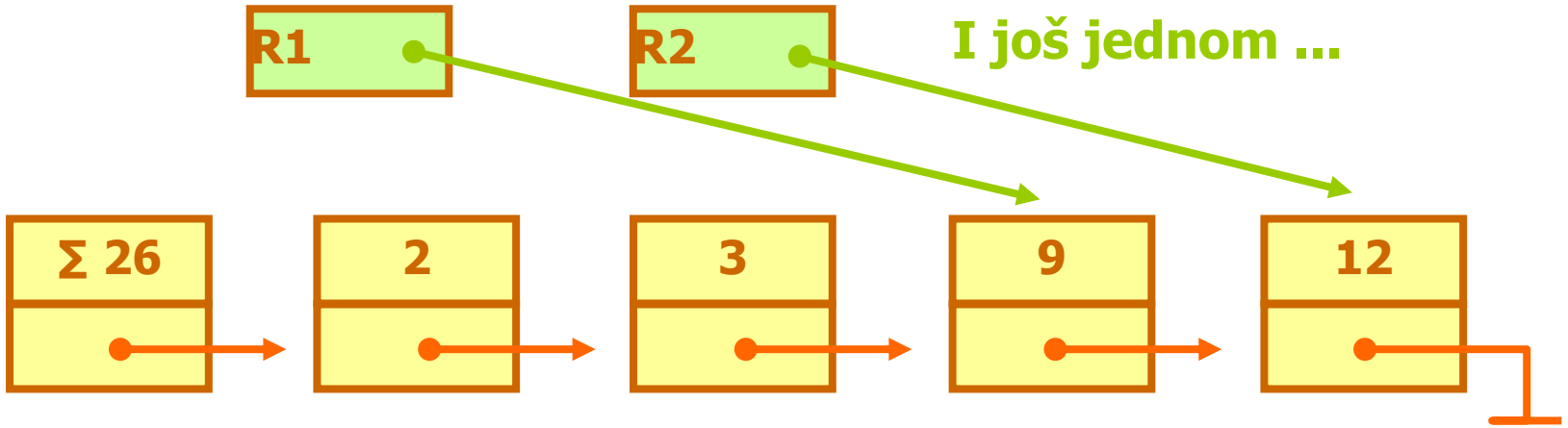


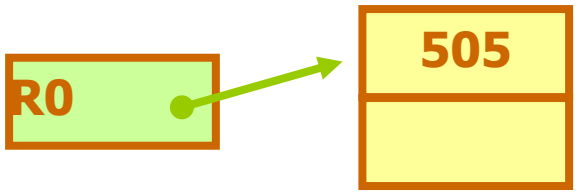
Novi element

Ostali primjeri

Način rada potprograma:

Pogledajmo još slučaj kad su u listi svi elementi manji od novog člana (ili je lista prazna)





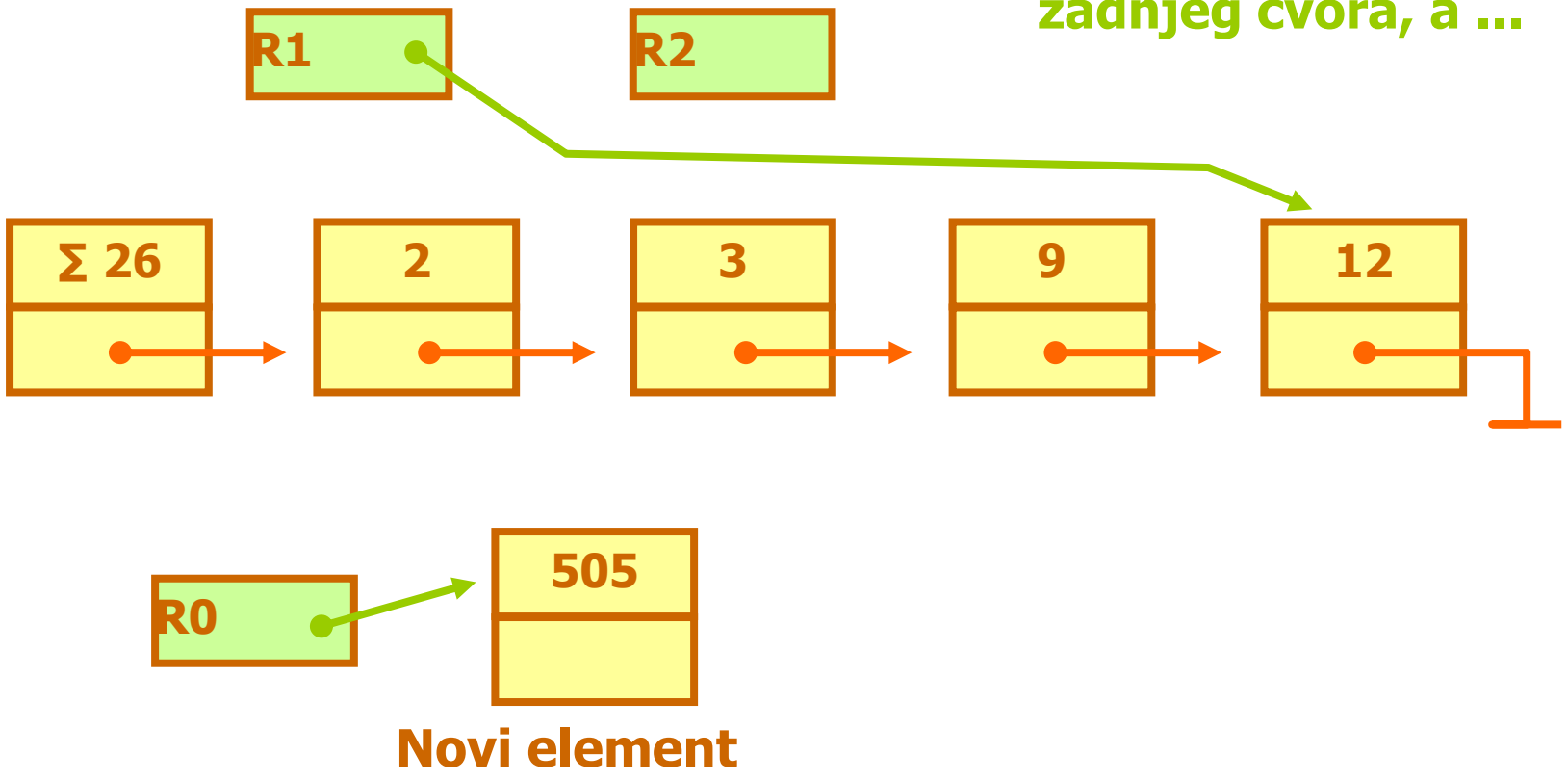
Novi element

Ostali primjeri

Način rada potprograma:

Pogledajmo još slučaj kad su u listi svi elementi manji od novog člana (ili je lista prazna)

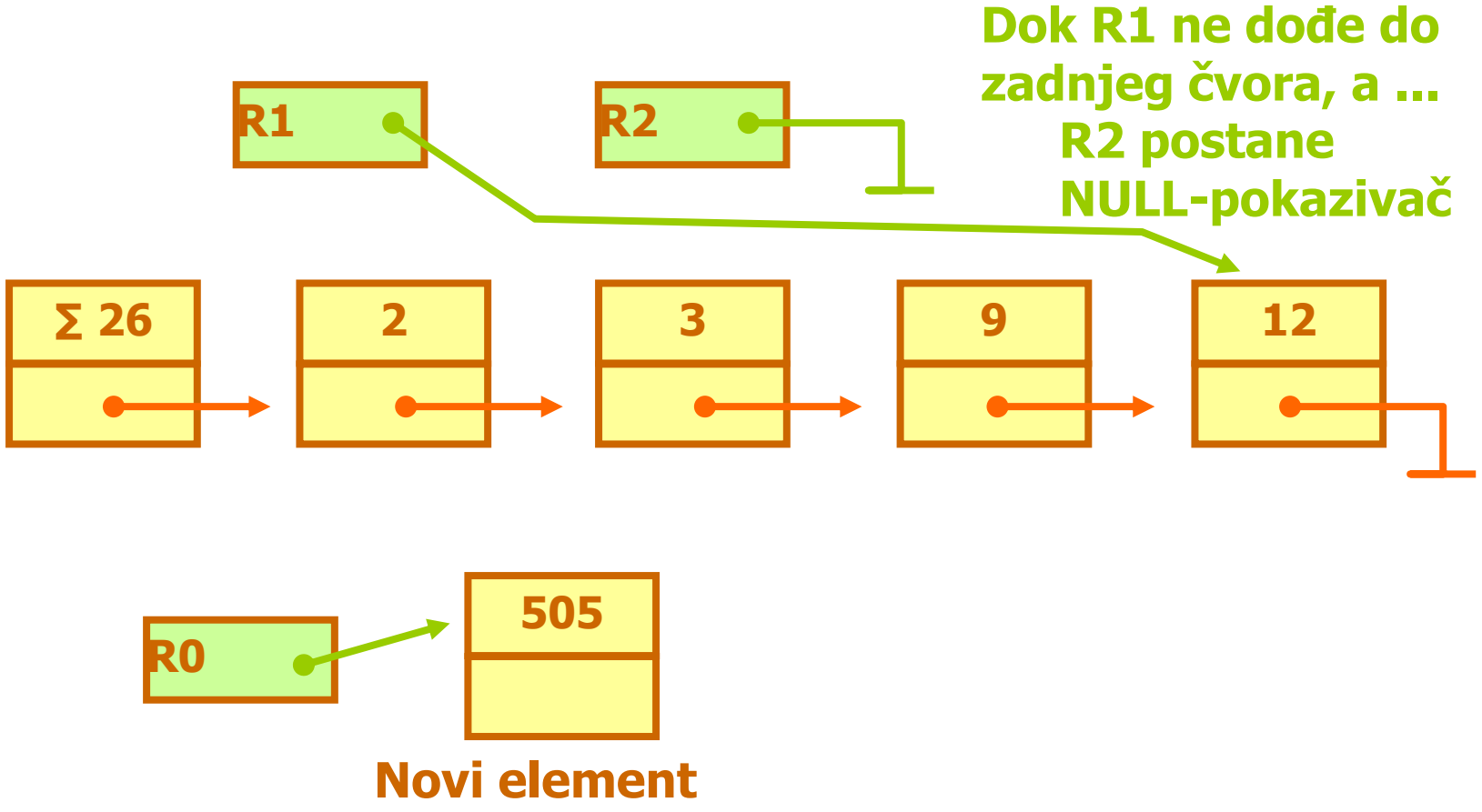
Dok R1 ne dođe do zadnjeg čvora, a ...



Ostali primjeri

Način rada potprograma:

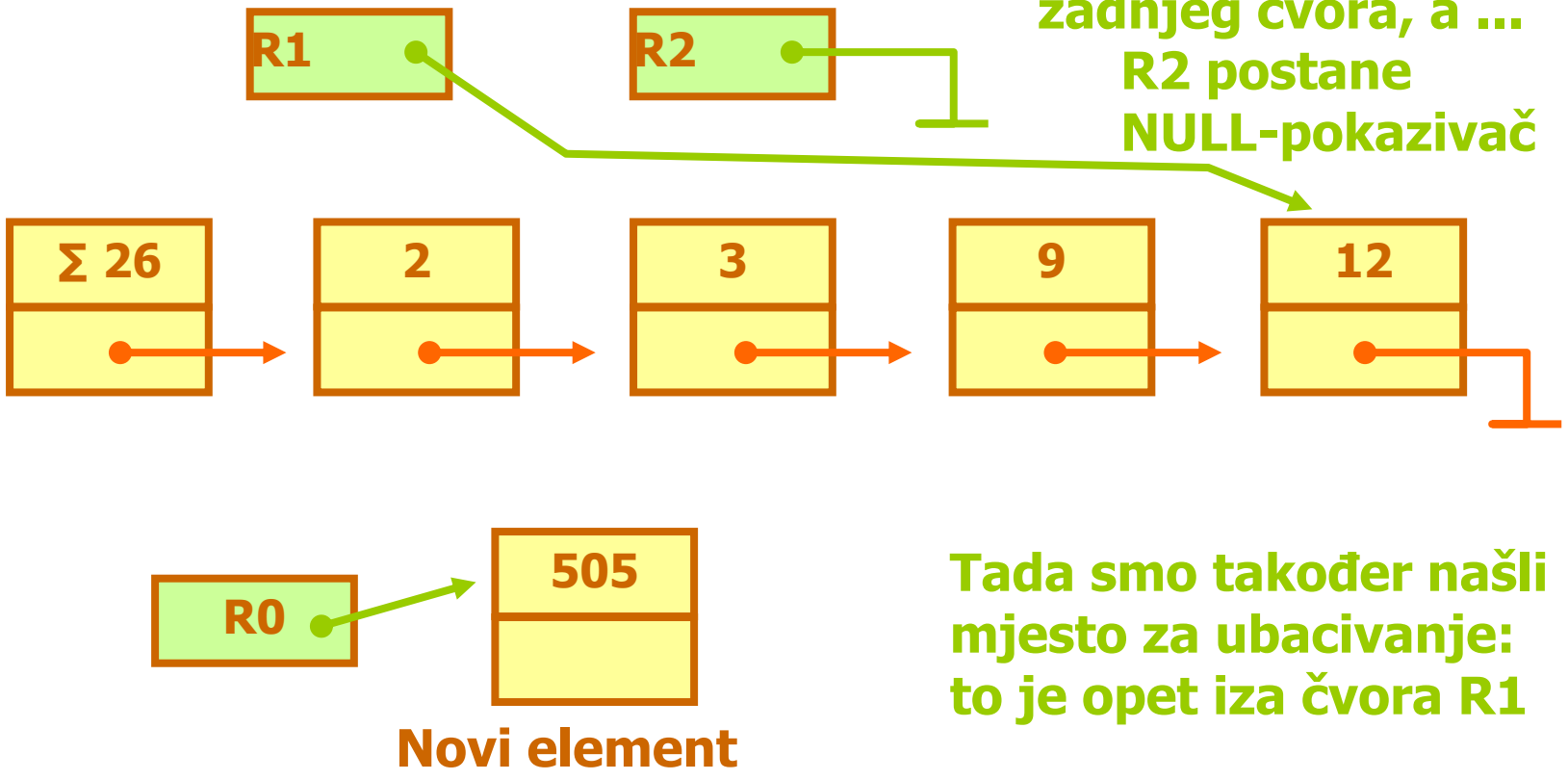
Pogledajmo još slučaj kad su u listi svi elementi manji od novog člana (ili je lista prazna)



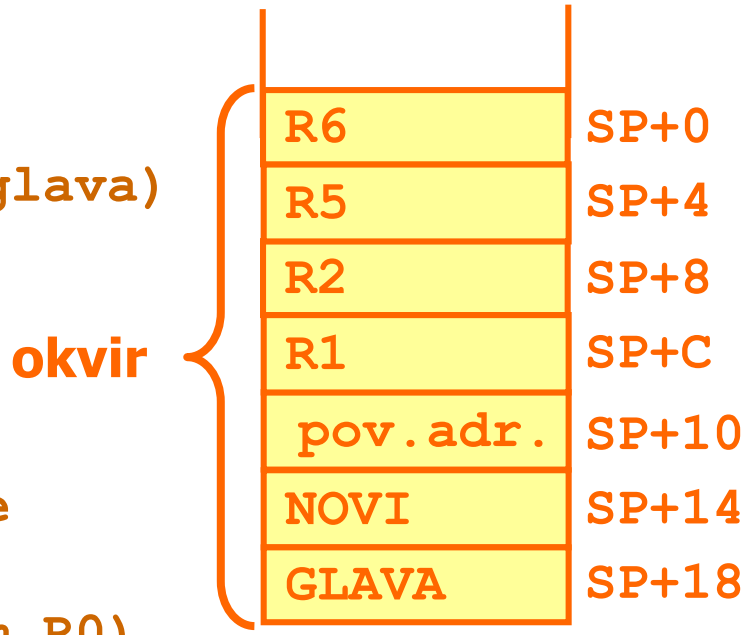
Ostali primjeri

Način rada potprograma:

Pogledajmo još slučaj kad su u listi svi elementi manji od novog člana (ili je lista prazna)



```
UBACI ;;;;; Potprogram UBACI
      ; Parametri na stogu:
      ; 1. parametar:
      ;    adresa prvog čvora (glava)
      ; 2. parametar:
      ;    adresa novog čvora

      PUSH   R1   ; Spremi sve
      PUSH   R2   ; registre koje
      PUSH   R5   ; potprogram
      PUSH   R6   ; mijenja (osim R0).

      ; dohvat vrijednosti novog čvora za pretraživanje

      LOAD   R0, (SP+14)    ; adresa novog čvora u R0
      LOAD   R6, (R0)       ; vrijednost novog čvora u R6

      ; priprema pokazivača R1 i R2 za pretraživanje

      LOAD   R1, (SP+18)    ; adresa prvog čvora u R1
      LOAD   R2, (R1+4)     ; adresa drugog čvora u R2
```



>>>>

<<<<

```
      ;;;; PETLJA ZA TRAŽENJE MJESTA ZA UBACIVANJE

TRAZI CMP    R2, 0        ; Ako je kraj liste:
      JR_Z   NASAO        ; =>  onda ubacujemo na kraj

      ; dohvati u R5 vrijednost trenutačnog čvora i
      ; usporedi je s vrijednošću R6 novog čvora
      LOAD   R5, (R2)
      CMP    R6, R5

      JR_ULE NASAO    ; ako je novi ≤ trenutačni =>
                      ; pronađeno je mjesto za ubacivanje

      ; inače treba nastaviti s petljom za traženje

      MOVE   R2, R1       ; pomakni se na sljedeći čvor
      LOAD   R2, (R1+4)

      JR     TRAZI        ; nastavi s traženjem
```

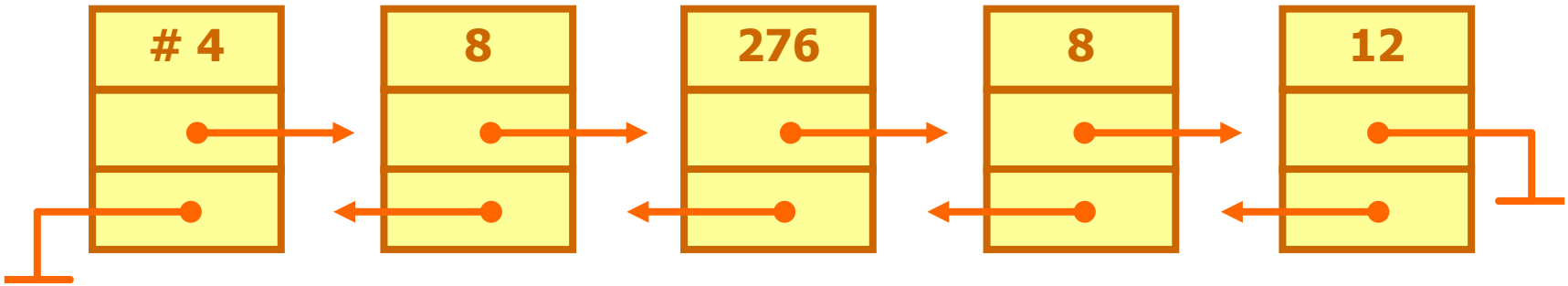
>>>>

```
<<<<
      ;;;; DIO ZA UMETANJE NOVOG ČVORA U LISTU

NASAO STORE  R0, (R1+4) ; stavi novi čvor iza prethodnog
      STORE  R2, (R0+4) ; stavi trenutačni čvor iza novog

      ; izračunavanje novog zbroja u prvom čvoru:

      LOAD   R1, (SP+18); dohvati adresu prvog
      LOAD   R0, (R1)   ; dohvati dosadašnji zbroj

      ADD    R0, R6, R0 ; novi zbroj stavi u R0 (rezultat)
      STORE  R0, (R1)   ; spremi ga u prvi čvor

      POP    R6     ; obnovi
      POP    R5     ; vrijednosti
      POP    R2     ; spremljenih
      POP    R1     ; registara

      RET
```

Ostali primjeri

Primjer:

U memoriji se nalazi dvostruko povezana nesortirana lista. Svaki čvor zauzima tri memorijske lokacije: na prvoj je NBC broj, na drugoj je pokazivač na sljedeći čvor liste, a na trećoj je pokazivač na prethodni čvor liste. Prvi čvor liste ima specijalno značenje i uvijek je prisutan u listi. U njemu se pamti ukupan broj preostalih čvorova u listi.



>>>>

Ostali primjeri

<<<<

Treba napisati potprogram IZBACI koji prima preko stoga dva parametra: pokazivač na prvi čvor liste i NBC broj (X). Potprogram traži prvo pojavljivanje čvora u kojemu je upisan broj X. Ako ga nađe, izbacuje taj čvor iz liste. Povratna vrijednost je adresa izbačenog čvora ili nula ako čvor nije pronađen. Vrijednost se vraća pomoću R1. Potprogram ne smije mijenjati sadržaje registara u glavnom programu.

Rješenje:

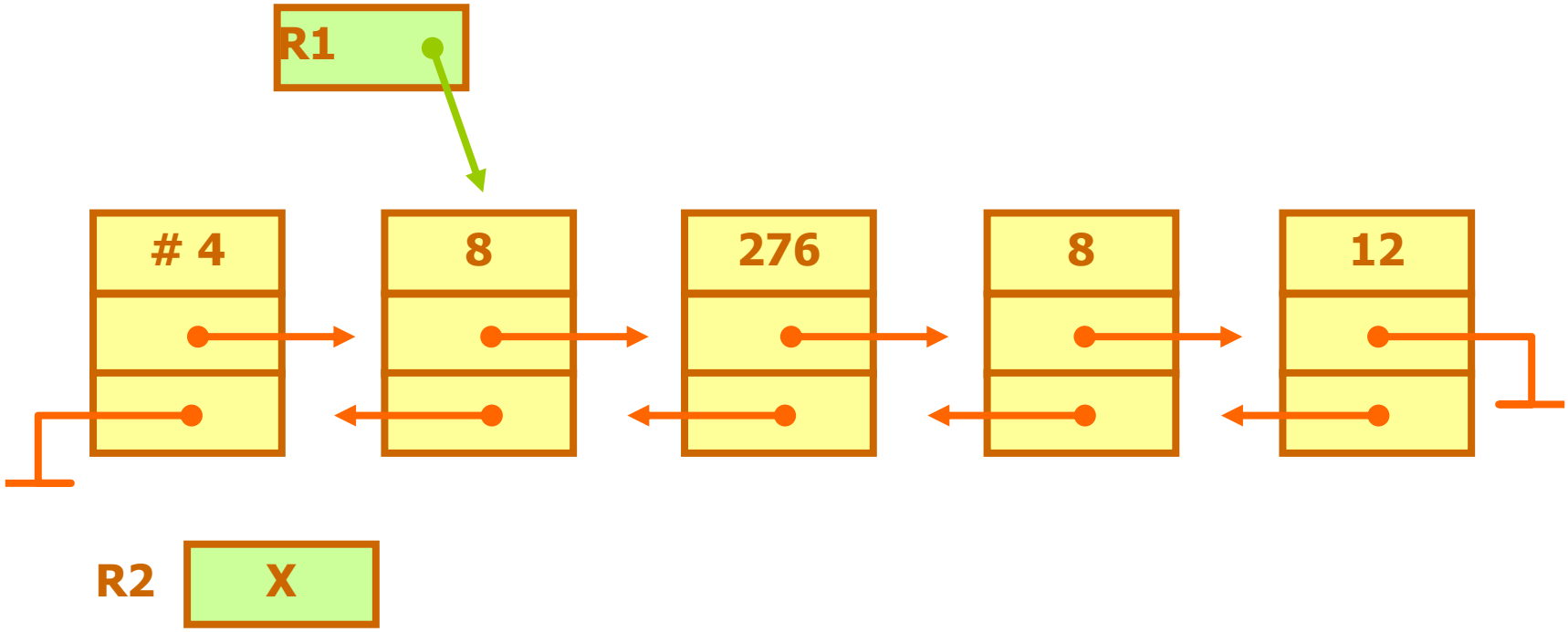
Parametre će sa stoga uklanjati glavni program.

>>>>

Ostali primjeri

<<<< Način rada potprograma:

Pokazivač na čvor koji se uspoređuje s X (početno pokazuje na drugi čvor)

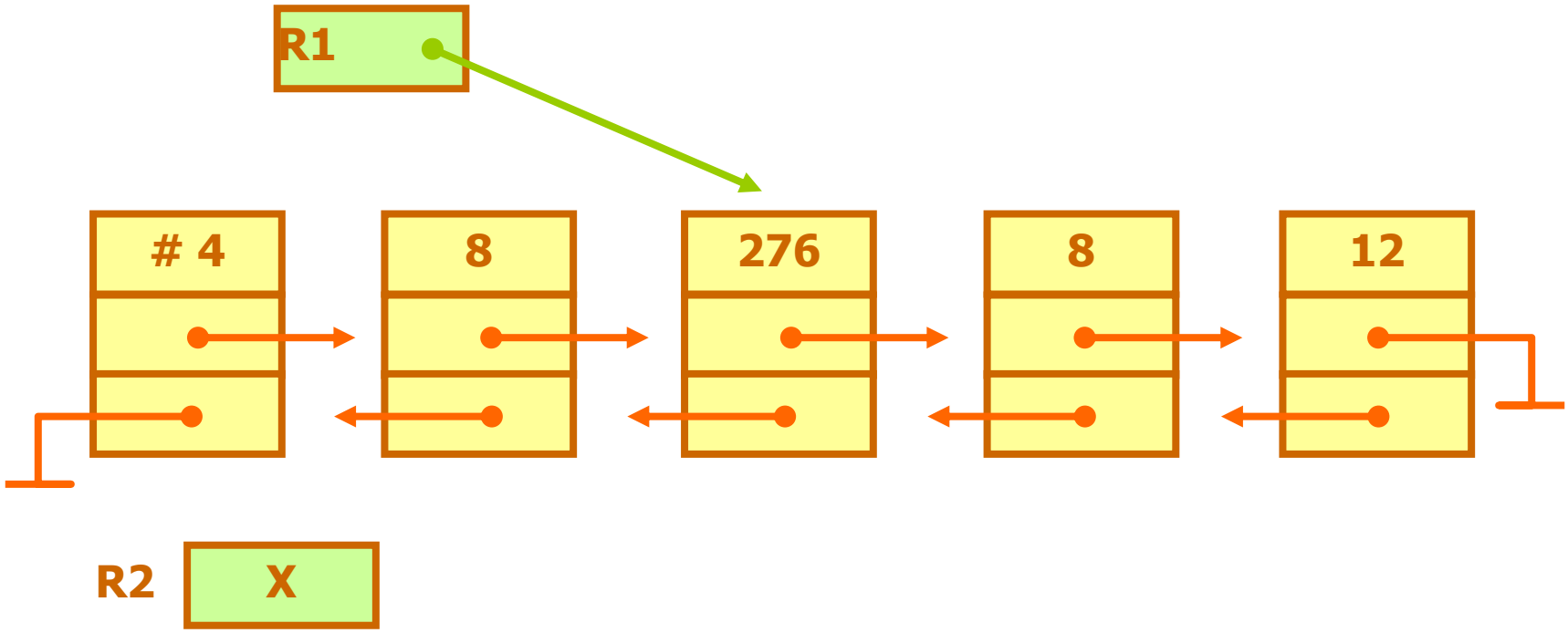


Broj X bit će stalno spremljen u R2

Ostali primjeri

<<<< Način rada potprograma:

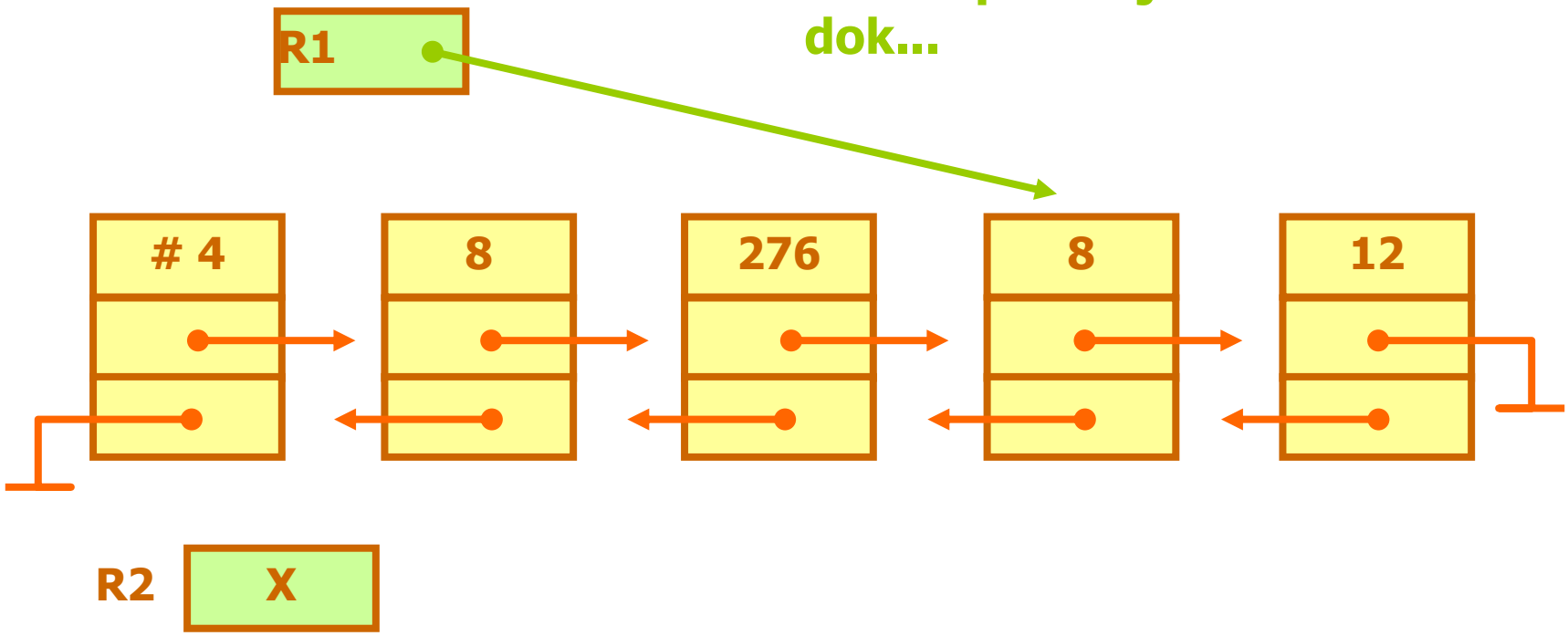
Pomičemo R1 za jedan čvor dalje ...



Ostali primjeri

<<<< Način rada potprograma:

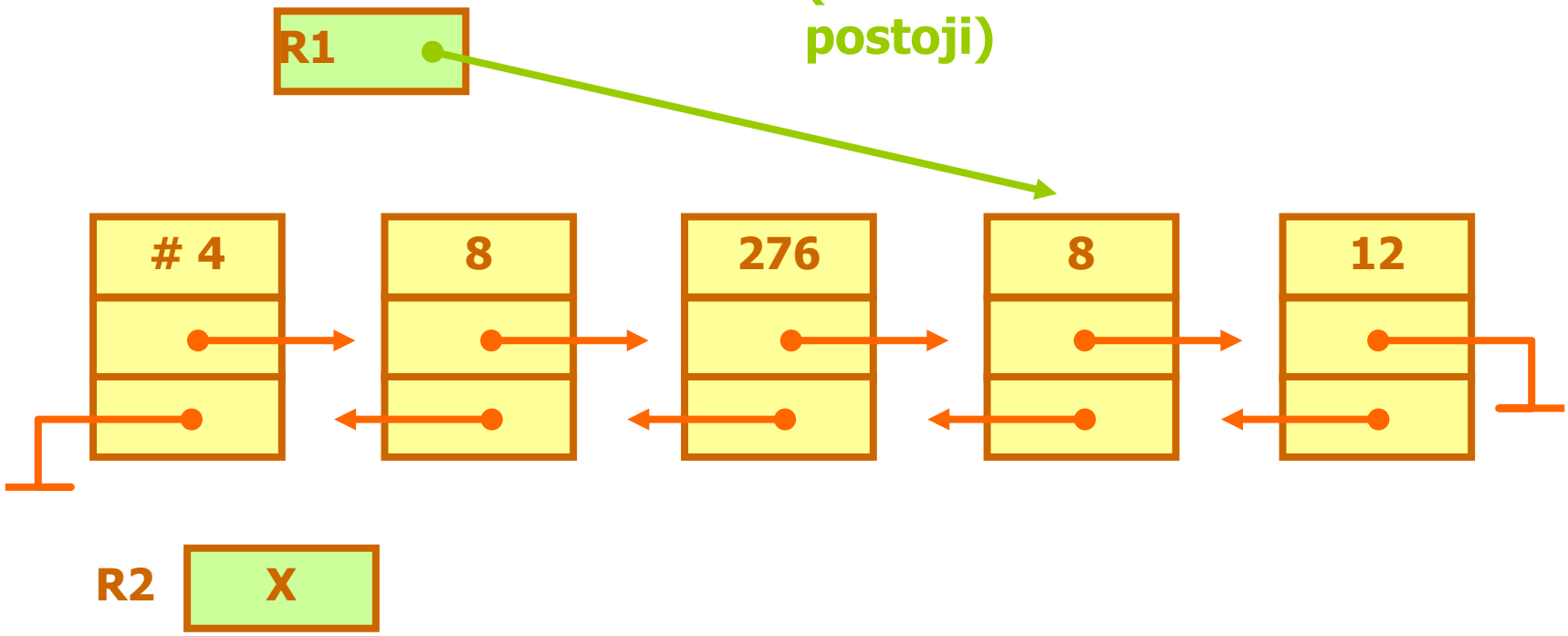
Pomičemo R1 za jedan čvor dalje ...
I to se ponavlja sve dok...



Ostali primjeri

<<<< Način rada potprograma:

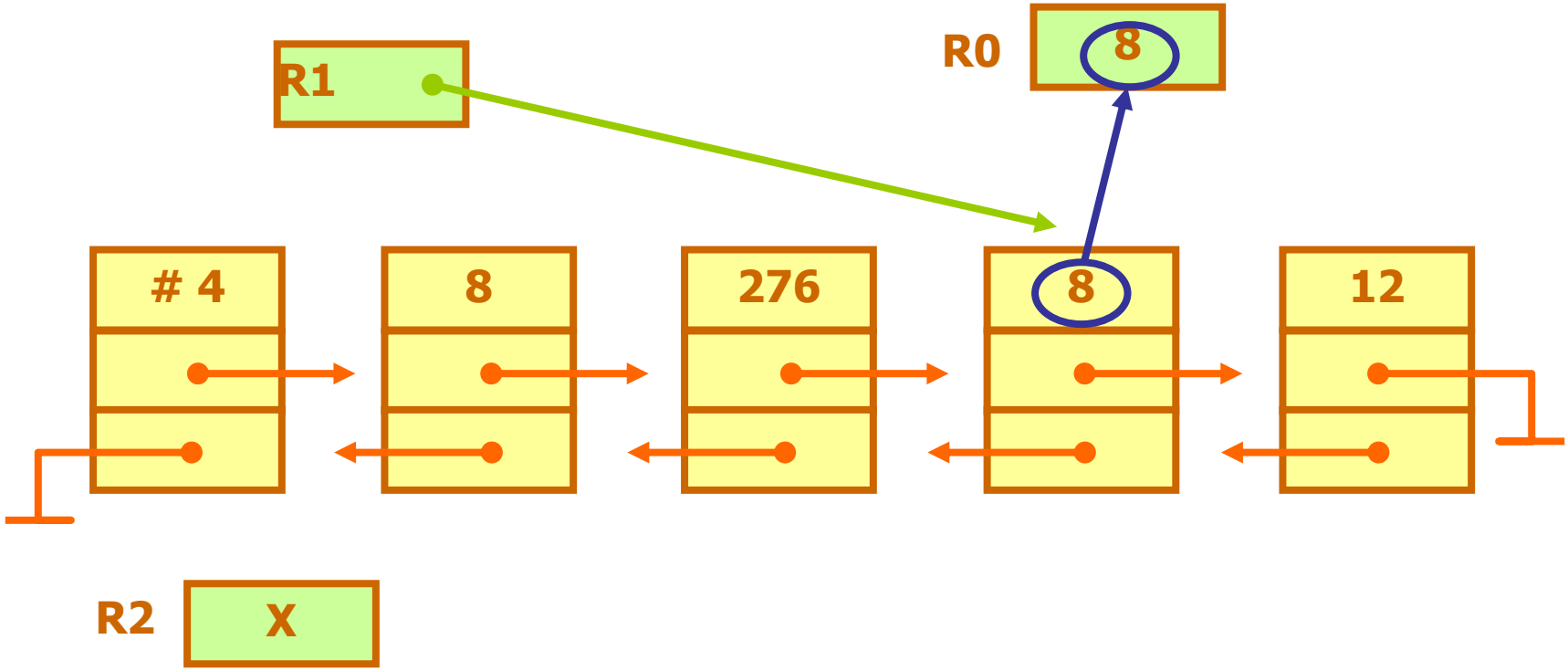
Ne pronađemo čvor ili dođemo do kraja liste (tada traženi čvor ne postoji)



Ostali primjeri

<<<< Način rada potprograma:

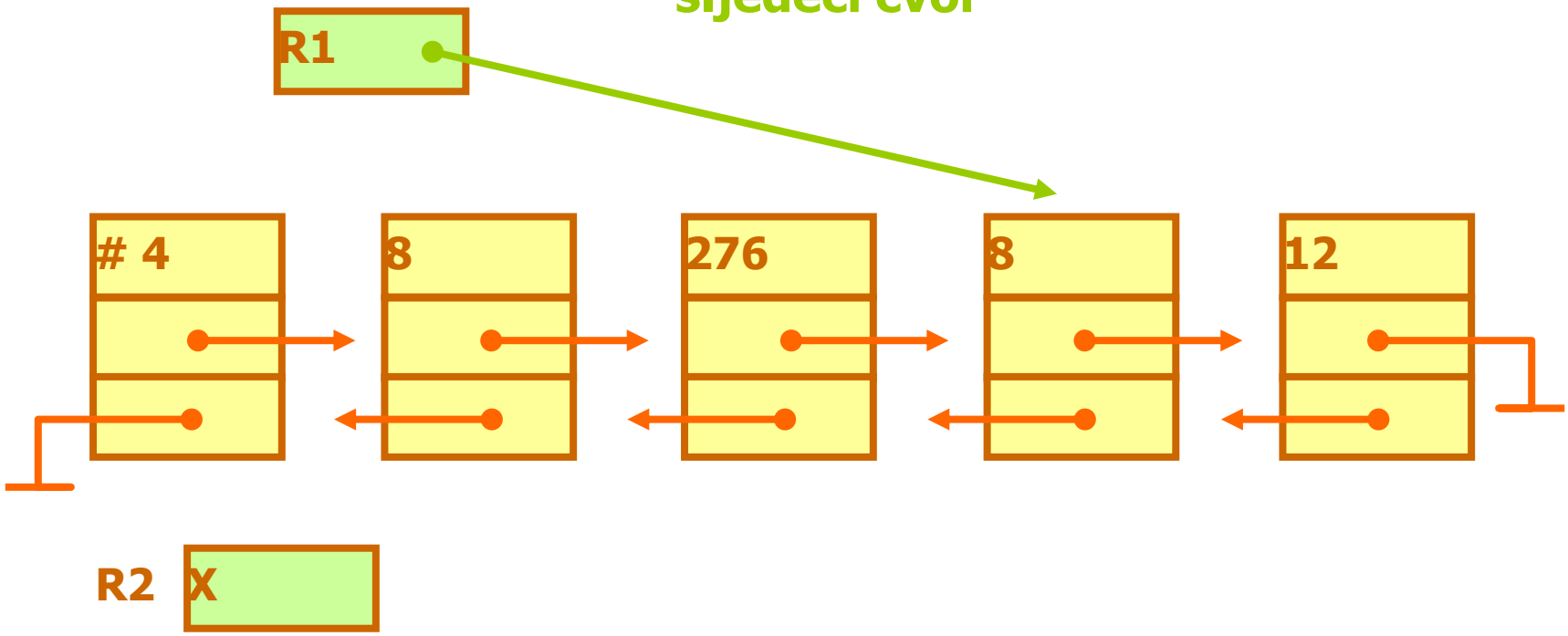
Prilikom traženja, u R0 ćemo učitati vrijednost čvora na kojeg pokazuje R1



Ostali primjeri

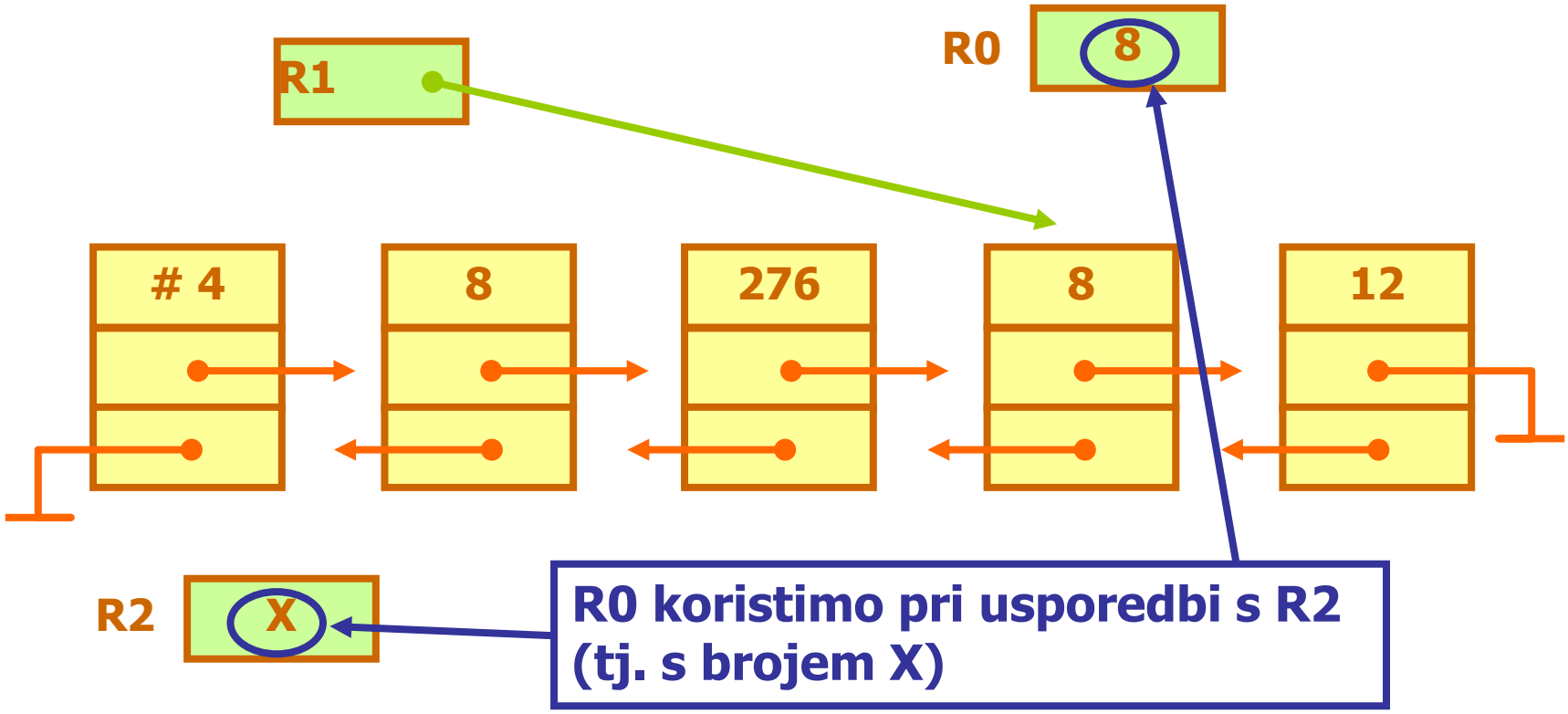
<<<< Način rada potprograma:

Ako pronađemo čvor, lako ga izbacujemo, jer u njemu postoje pokazivači i na prethodni i na sljedeći čvor



Ostali primjeri

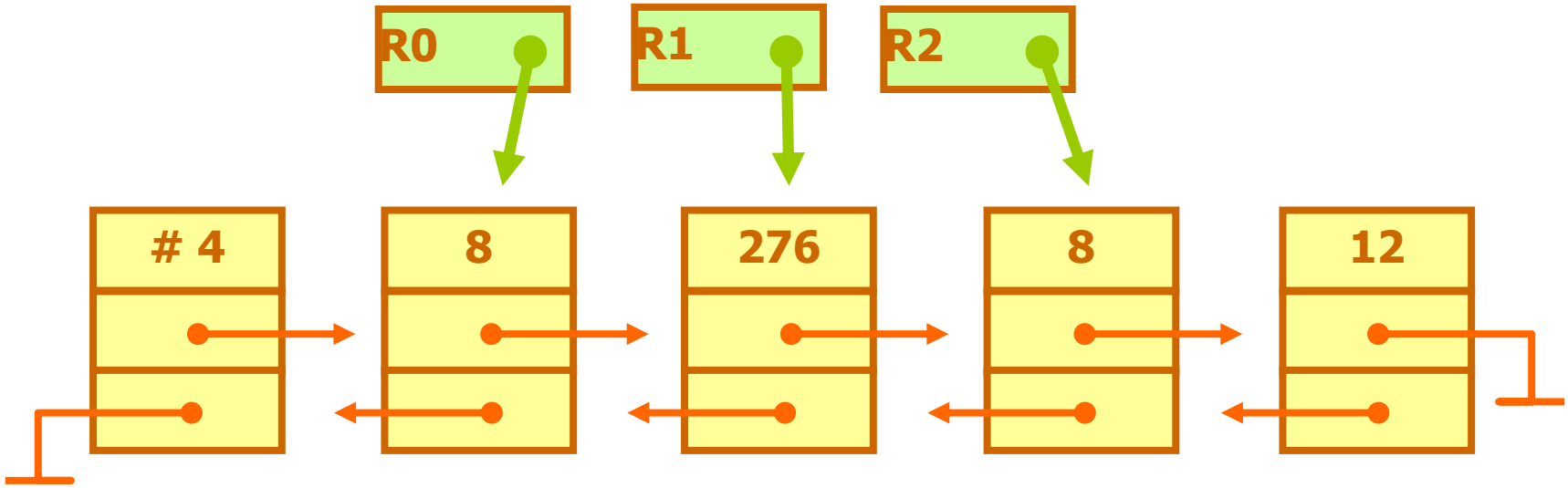
<<<< Način rada potprograma:



Ostali primjeri

<<<< Način rada potprograma:

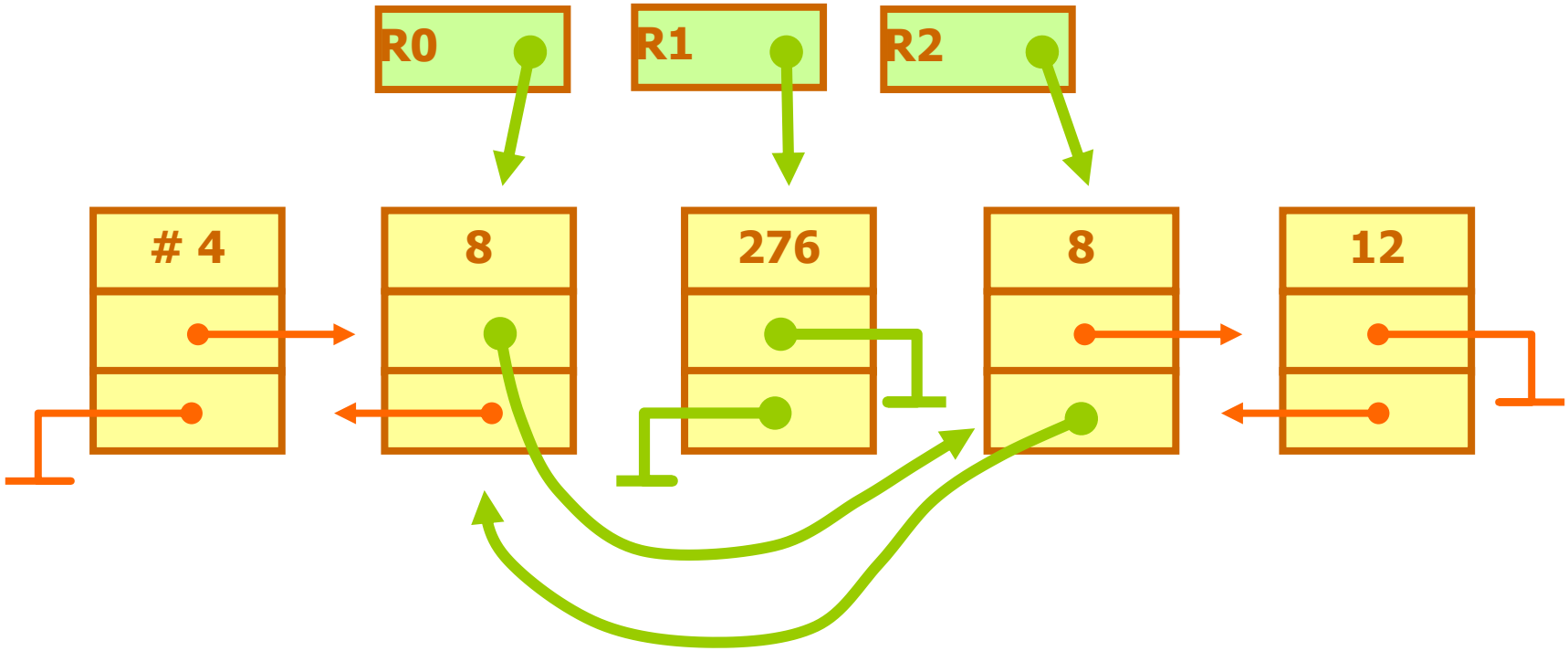
Kod izbacivanja čvora,
registri se postavljaju ovako
(na ovoj slici izbacuje se čvor 276):



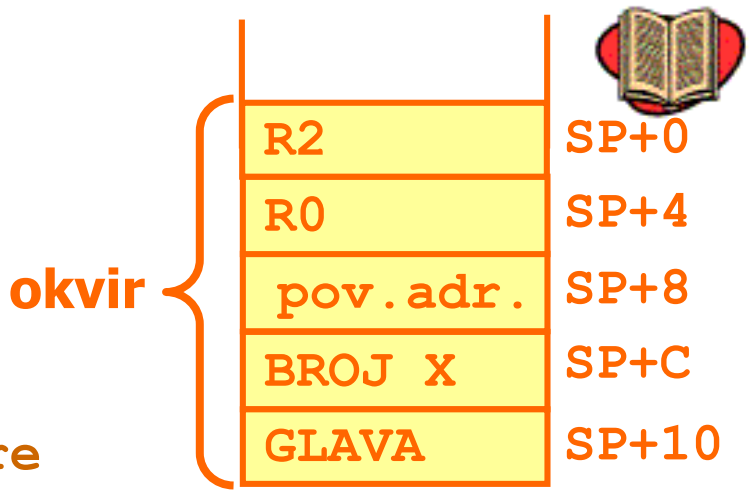
Ostali primjeri

<<<< Način rada potprograma:

Pokazivači u čvorovima se prespajaju ovako:



```
IZBACI ;;;;; Potprogram IZBACI
       ; Parametri na stogu:
       ; 1. parametar:
       ;    adresa prvog čvora
       ; 2. parametar:
       ;    broj X koji se traži

       PUSH  R0      ; spremi registre
       PUSH  R2      ; iz glavnog programa

       LOAD  R2, (SP+0C) ; dohvati broj X
       LOAD  R1, (SP+10) ; dohvati adresu prvog čvora
       LOAD  R1, (R1+4)  ; pripremi pokazivač za traženje

TRAZI  CMP   R1, 0
       JR_Z  IZLAZ       ; ako je kraj => čvor nije nađen

       LOAD  R0, (R1+0)  ; dohvati vrijednost čvora
       CMP   R0, R2      ; usporedi je sa X
       JR_EQ NASAO       ; ako su isti => čvor je nađen

       LOAD  R1, (R1+4)  ; pomakni se na sljedeći čvor
       JR    TRAZI       ; nastavi s traženjem
```



>>>>

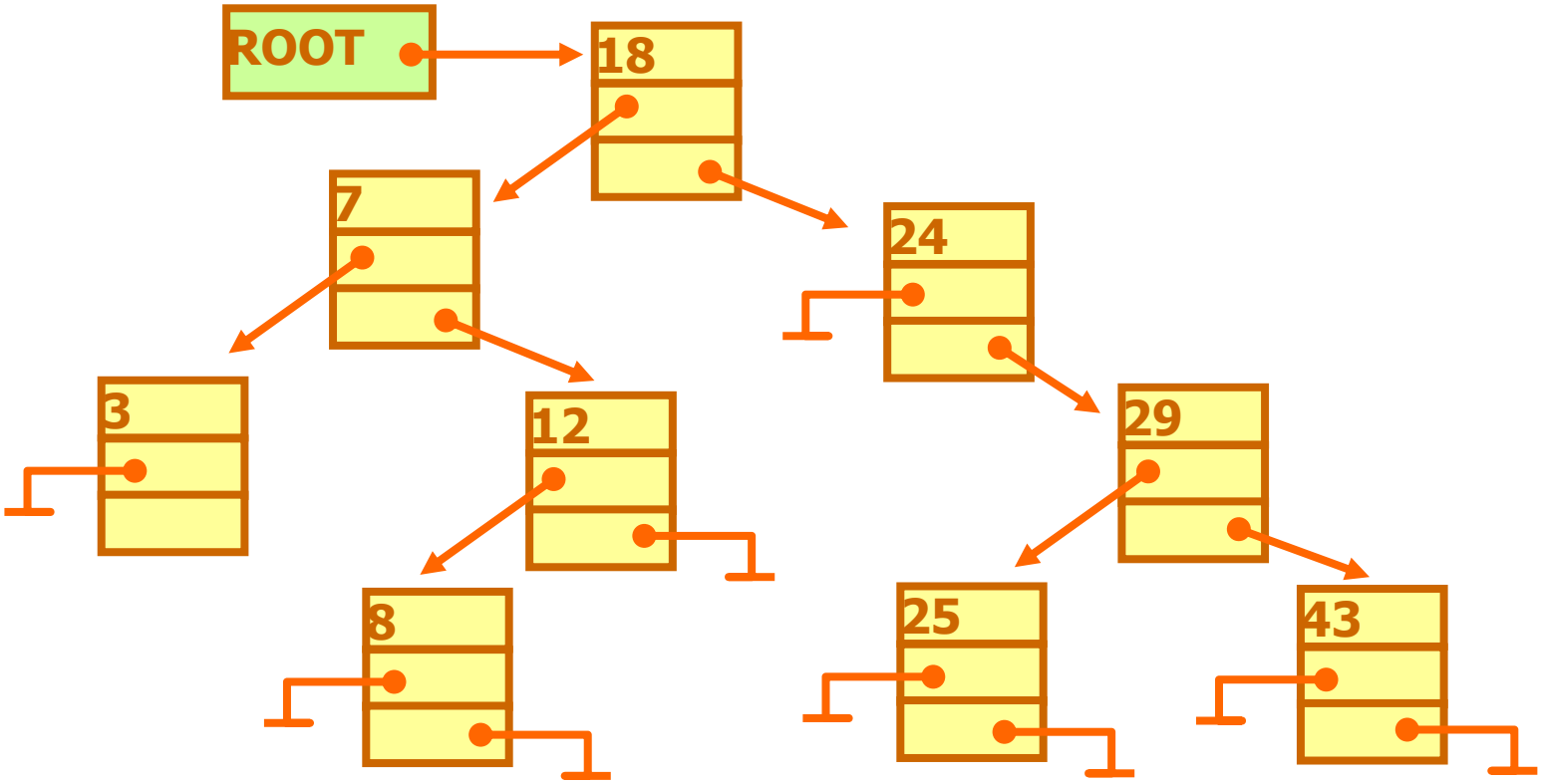
```
<<<<
NASAO ;;;; odspoji nađeni čvor iz liste
      LOAD   R0, (R1+8)   ; stavi adresu prethodnog u R0
      LOAD   R2, (R1+4)   ; stavi adresu sljedećeg u R2

      STORE  R2, (R0+4)   ; stavi sljedeći iza prethodnog

      OR     R2, R2, R2   ; provjeri izbacuje li se zadnji
      JR_Z   DALJE        ; čvor iz liste (tj. R2 je NULL)

      STORE  R0, (R2+8)   ; stavi prethodni ispred sljedeć.

DALJE MOVE   0, R0        ; odspoji pokazivače
      STORE  R0, (R1+4)   ; u čvoru kojeg
      STORE  R0, (R1+8)   ; izbacuješ iz liste

      ;;;; smanji brojač čvorova u 1. čvoru liste
      LOAD   R0, (SP+10)  ; dohvati adresu 1. čvora
      LOAD   R2, (R0+0)   ; dohvati broj čvorova liste
      SUB    R2, 1, R2    ; smanji broj čvorova
      STORE  R2, (R0+0)   ; upiši ga natrag u 1. čvor

IZLAZ POP    R2           ; obnovi registre
      POP    R0           ; iz glavnog programa
      RET
```

Ostali primjeri

Primjer:

U memoriji se nalazi sortirano binarno stablo. Svaki čvor zauzima tri memorijske lokacije: na prvoj je NBC broj, na drugoj je pokazivač na lijevo podstablo, a na trećoj je pokazivač na desno podstablo.

Poseban pokazivač (ROOT) u glavnom programu pokazuje na korijen stabla.



Ostali primjeri

<<<<

Treba napisati potprogram PISI koji će ispisati sve brojeve iz čvorova stabla, ali u rastućem redoslijedu. Parametar potprograma PISI je adresa ishodišnog čvora, a prenosi se stogom. Povratna vrijednost potprograma je broj ispisanih čvorova, a vraća se preko R0.

Ispisivanje se obavlja tako da se pozove potprogram PRINT i kao parametar mu se pošalje broj koji se želi ispisati. Parametar za PRINT šalje se preko R0, a PRINT nema povratne vrijednosti. Pretpostavite da potprogram PRINT već postoji negdje u memoriji.

Treba napisati i glavni program, koji će ispisati stablo čija adresa je u pokazivaču ROOT, a broj ispisanih čvorova treba spremiti U R5.

>>>>

Ostali primjeri

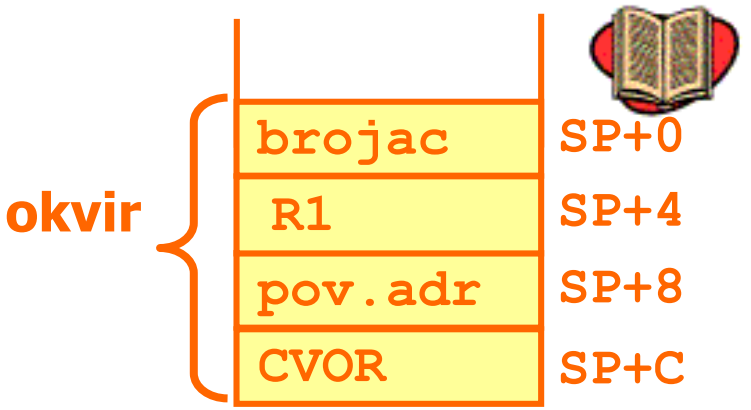
<<<<

Idejno rješenje (u C-u):

```
int PISI ( struct cvor_stabla * cvor ) {

    int brojac;

    if( cvor == NULL )
        return 0;

    // "in-order" obilazak
    brojac = PISI( cvor -> lijevi );
    PRINT( cvor -> broj );
    brojac += PISI( cvor -> desni );

    return (brojac+1);
}
```

>>>>

```
GLAVNI  LOAD R0, (ROOT)
        PUSH R0
        CALL PISI
        ADD  SP,4,SP
        MOVE R0, R5
        HALT
ROOT    DW ... ; adresa stabla
```




```
PISI ;;;;; Potprogram PISI
     ; Parametar na stogu:
     ; 1. parametar:
     ;    adresa čvora

     PUSH  R1       ; spremi registre

     SUB   SP, 4, SP    ; lokalna varijabla brojac

     LOAD  R1, (SP+C)   ; dohvati adresu čvora
     MOVE  0, R0
     CMP   R1, 0        ; if( cvor == NULL)
     JR_Z  VAN          ; idi na return
```

>>>>

<<<<

```
     LOAD   R0,(R1+4)  ; cvor->lijevi
     PUSH   R0
     CALL   PISI       ; PISI()
     ADD    SP, 4, SP

     STORE  R0,(SP+0)  ; brojac =
                       ;   povratna vrijednost

     LOAD   R0, (R1)   ; cvor->broj
     CALL   PRINT      ; PRINT()

     LOAD   R0,(R1+8)  ; cvor->desni
     PUSH   R0
     CALL   PISI       ; PISI()
     ADD    SP, 4, SP

     LOAD   R1,(SP+0)  ; stavi brojac u R1
     ADD    R1,R0,R0   ; brojac += povratna vrijednost
     ADD    R0, 1, R0  ; stavi (brojac+1) u R0 za return

     ; return
VAN  ADD    SP, 4, SP ; ukloni lokalnu var.
     POP    R1        ; obnovi registre
     RET
```