Pokazivači i liste zadaci

Problem 1. Napisati funkciju koja ispisuje sve elemente liste. Struktura liste je sledeća:

```
============================================================
01     struct List
02            val: integer
03            next: pointer List
04     end struct
============================================================
```

Rešenje. Nama je dat samo pokazivač na prvi element liste, ali znamo vezu između susednih elemenata u listi. Dovoljno je da ispišemo vrednost prvog elementa, i da pređemo na drugi element koristeći vrednost promenljive *next*. Kada ispišemo drugi element liste, prelazimo na treći, itd. sve dok ne dođemo do kraja liste, tj. do elementa lista koji ne pokazuje ni na jedan element.

```
============================================================
01     function PrintList( list: pointer List )
02            while list <> null do
03                   print list.val
04                   list = list.next
05            end while
06     end function
============================================================
```

Problem 2. Napisati funkciju koja izbacuje n-ti element iz liste. Struktura liste je ista kao u prethodnom zadatku.

Rešenje. Da bismo izbacili n-ti element iz liste, potrebno je da obrišemo vezu između elemenata na pozicijama $n-1$ i n, postavimo da element na poziciji $n-1$ pokazuje na element na poziciji $n+1$ i da obrišemo n-ti element iz memorije. Potrebno je paziti na slučajeve kada ne postoje neki od elemenata na pozicjama $n-1$, n i $n+1$.

```
============================================================
01     function DeleteElement( ref list: pointer List, n: integer )
02            if ( list = null ) then
03                   return
04            end if
05
06            if n = 1 then       // potrebno je izbaciti prvi element liste
07                   previousElement = lista
08                   list = list.next
09                   delete previousElement
10            else
11                   currentElement = list
12                   previousElement = nil
13                   while n > 1 and currentElement <> null do
14                          n = n - 1
15                          previousElement = currentElement
16                          currentElement = currentElement.next
17                   end while
```

```
18
19                  if currectElement <> null do     // postoji n-ti element
20                        nextElement = currentElement.next
21                        previousElement.next = nextElement     //  (n-1)-vi
element pokazuje na (n+1)-vi element
22                        delete currentElement     // obrisi n-ti element iz
memorije
23                  end if
24            end if
25      end function
        ============================================================
```

Zadatak 3. N dece sede u krugu i igra se sledeća igra. Počinje se od deteta X i broje se deca u smeru kazaljke na satu, kada se dođe do broja M dete na koje je stalo brojanje ispada iz igre. Sledeća runda počinje od deteta koje bi bilo broj $M + 1$, međutim menja se smer brojanja. Igra se $N - 1$ rundi, sve dokle ne ostane samo jedno dete. Vama su data imena dece, ime deteta od kojeg se kreće brojanje u prvoj rundi, kao i brojevi N i M, potrebno je odrediti koje dete pobeđuje u igri.

Rešenje. U svakoj rundi je potrebno izbaciti M-to dete, a videli smo da izbacivanje kod liste možemo na lep način da uradimo, tj. napravićemo listu gde će svaki element predstavljati jedno dete. U ovom zadatku ćemo koristiti kružne liste, tj. poslednji u listi pokazuje na prvog, to će nam pomoći ukoliko dođemo do kraja liste iako nismo odbrojali do M. U tekstu zadatka piše da se smer brojanja menja u svakoj rundi, tako da će nam ovde koristiti da imamo dvostruko povezane liste, kako bi mogli da se krećemo kroz listu u oba smera. Potrebno je primetiti da ukoliko trenutno u listi imamo P elemenata, nema potrebe da se krećemo kroz listu M puta, već je dovoljno da se pomerimo samo $M\ mod\ P$ puta. Kod programa bi mogao da izgleda ovako:

```
        ============================================================
01      ReadInput()
02      MakeList()
03      currentChild = FindElement(X)
04      for i = 1 to N-1 do
05            if  i mod 2 = 0 then
06                  Play( currentChild, N-i+1, M, true )
07            else
08                  Play( currentChild, N-i+1, M, false )
09            end if
10      end for
11      Print( currentChild->name )
        ============================================================
```

Ovde ćemo napisati samo kod funkcije Play(currentChild, sizeOfList, M, clockwise).

```
        ============================================================
01      function Play( ref currentChild: pointer List, sizeOfList: integer, M:
integer, clockwise: boolean )
02            for i = 1 to (M mod sizeOfList) do
03                  if clockwise then
04                        currentChild = currentChild->next
05                  else
06                        currentChild = currentChild->previous
07                  end if
08            end for
```

```
09
10          previousChild = currentChild->previous
11          nextChild = currentChild->next
12
13          previousChild->next = nextChild
14          nextChild->previous = previousChild
15          delete currentChild
16
17          if clockwise then
18                currentChild = nextChild
19          else
20                currentChild = previousChild
21          end if
22    end function
============================================================
```