

# Програмирање 1

Групно спремање – задаци

# Увод и примери (за почетнике)



Написати програм који учита два броја са стандардног улаза и испише њихов збир на стандардни излаз.

верзија 1:

```
PROGRAM prog(input, output);  
VAR x, y, z: INTEGER;  
BEGIN  
    readln(x, y);  
    z:=x+y;  
    writeln(z);  
END.
```

верзија 2:

```
PROGRAM prog(input, output);  
VAR x, y: INTEGER;  
BEGIN  
    readln(x, y);  
    writeln(x+y);  
END.
```

Написати програм који учита два броја са стандардног улаза и испише њихов збир на стандардни излаз ако је збир позитиван.

```
PROGRAM prog(input, output);  
VAR x, y, z: INTEGER;  
BEGIN  
    readln(x, y);  
    z := x + y;  
    if z > 0 then  
        writeln(z)  
    else  
        writeln('Zbir je negativan.');
```

END.



Написати програм који учита редни број дана у недељи и испише (словима) који је то дан.

```
PROGRAM prog(input, output);
VAR dan: INTEGER;
BEGIN
    readln(dan);
    CASE dan OF
        1: writeln('ponedeljak');
        2: writeln('utorak');
        3: writeln('sreda');
        4: writeln('cetvrtak');
        5: writeln('petak');
        6: writeln('subota');
        7: writeln('nedelja');
    else
        writeln('greska');
    END;
END.
```

Написати програм који учита неколико бројева (низ бројева) и испише квадрат сваког броја.

```
PROGRAM prog(input, output);
VAR niz: ARRAY [1..100] OF INTEGER;
    i,n: INTEGER;
BEGIN
    {duzina niza}
    writeln('Koliko brojava zelite da unesete?');
    readln(n);
    {unos elemenata}
    writeln(Unesite elemente:');
    FOR i:= 1 TO n DO
        read(niz[i]);
    {racunanje kvadrata}
    writeln('Kvadrati unetih brojeva su:');
    FOR i:= 1 TO n DO
        writeln(sqr(niz[i]));
    END.
```



Написати програм који учита неколико бројева (низ бројева) и испише корен сваког броја (уназад).

```
PROGRAM prog(input, output);  
VAR niz: ARRAY [1..100] OF INTEGER;  
    i,n: INTEGER;  
BEGIN  
    writeln('Koliko brojava zelite da unesete?');  
    readln(n);  
    FOR i:= 1 TO n DO  
        read(niz[i]);  
  
    writeln('Kvadrati unetih brojeva su:');  
    FOR i:= n DOWNT0 1 DO  
        writeln(sqrt(niz[i]));  
END.
```

Написати програм који учита низ бројева и израчуна њихов збир.

```
PROGRAM prog(input,output);  
VAR niz: ARRAY [1..100] OF INTEGER;  
    i,n,zbir: INTEGER;  
BEGIN  
    writeln('Koliko brojava zelite da unesete?');  
    readln(n);  
    FOR i:= 1 TO n DO  
        read(niz[i]);  
  
    zbir:= 0;  
    FOR i:= 1 TO n DO  
        zbir:= zbir + niz[i];  
    writeln('Zbir brojeva je: ', zbir);  
END.
```



Написати програм који учита низ бројева и израчуна њихов производ.

```
PROGRAM prog(input,output);  
VAR niz: ARRAY [1..100] OF INTEGER;  
    i,n,proizvod: INTEGER;  
BEGIN  
    writeln('Koliko brojava zelite da unesete?');  
    readln(n);  
    FOR i:= 1 TO n DO  
        read(niz[i]);  
  
        proizvod:= 1;  
        FOR i:= 1 TO n DO  
            proizvod:= proizvod * niz[i];  
        writeln('Prizvod brojeva je: ', proizvod);  
    END.
```

Написати програм који учита низ бројева и испише суму сваког другог броја (сума бројева на непарним позицијама).

верзија 1:

```
PROGRAM prog(input,output);
VAR niz: ARRAY [1..100] OF INTEGER;
    i,n,zbir: INTEGER;
BEGIN
    writeln('Koliko brojava zelite da unesete?');
    readln(n);
    FOR i:= 1 TO n DO
        read(niz[i]);

        zbir:= 0;
    FOR i:= 1 TO n DO
        IF i mod 2 = 1 THEN
            zbir:= zbir + niz[i];
        writeln('Zbir brojeva je: ', zbir);
    END.
```



Написати програм који учита низ бројева и испише суму сваког другог броја (сума бројева на непарним позицијама).

верзија 2:

```
PROGRAM prog(input,output);
VAR niz: ARRAY [1..100] OF INTEGER;
    i,n,zbir: INTEGER;
BEGIN
    writeln('Koliko brojava zelite da unesete?');
    readln(n);
    FOR i:= 1 TO n DO
        read(niz[i]);

        zbir:= 0;
    FOR i:= 1 TO n div 2 DO
        zbir:= zbir + niz[i*2-1];
    writeln('Zbir brojeva je: ', zbir);
END.
```

Написати програм који учита низ бројева и испише суму сваког другог броја (сума бројева на непарним позицијама).

верзија 3:

```
PROGRAM prog(input,output);
VAR niz: ARRAY [1..100] OF INTEGER;
    i,n,zbir: INTEGER;
BEGIN
    writeln('Koliko brojava zelite da unesete?');
    readln(n);
    FOR i:= 1 TO n DO
        read(niz[i]);

        zbir:= 0;
        i:= 1;
        WHILE i <= n DO BEGIN
            zbir:= zbir + niz[i];
            i:=i+2;
        END;
    writeln('Zbir brojeva je: ', zbir);
END.
```



# Низови

# Дефинисање низа

Синтакса:

```
ARRAY [opseg] OF tip
```

Пример 1:

```
VAR a: ARRAY [1..100] OF INTEGER;
```

Пример 2:

```
TYPE niz = ARRAY [1..100] OF INTEGER;  
VAR a: niz;
```

Пример 3:

```
TYPE nizRealnih = ARRAY ['a'..'z'] OF REAL;  
VAR a: nizRealnih;
```



# Учитавање низа

- Процедура читава низ непознате дужине.
- Као параметри се задају празан низ  $a$  и празна променљива  $n$  у коју ће се уписати дужина.
- Сазнаје дужину низа тако што пита корисника да је унесе (помоћу `readln`).
- Не проверава да ли је учитана дужина валидна.

```
PROCEDURE ucitaj(VAR a:niz; VAR n:integer);  
VAR i: integer;  
BEGIN  
    writeln('Unesite duzinu niza: ');  
    readln(n);  
    writeln('Unesite niz: ');  
    FOR i:= 1 TO n DO  
        read(a[i]);  
    END;
```

# Учитавање низа

- Функција учитава низ непознате дужине.
- Као параметри се задају празан низ *a* и празна променљива *n* у коју ће се уписати дужина.
- Сазнаје дужину низа тако што пита корисника да је унесе (помоћу `readln`).
- Проверава да ли је учитана дужина валидна:
  - Ако је унета добра дужина учитава низ и враћа `true`
  - Ако је унета лоша дужина не учитава низ и враћа `false`

```
FUNCTION ucitaj(VAR a:niz; VAR n:integer):boolean;  
VAR i: integer;  
BEGIN  
    writeln('Unesite duzinu niza: ');  
    readln(n);  
    IF (n>0) AND (n<=MAX_DUZ) THEN BEGIN  
        writeln('Unesite niz: ');  
        FOR i:= 1 TO n DO  
            read(a[i]);  
        ucitaj:=true;  
    END  
    ELSE ucitaj:=false;  
END;
```



# Учитавање низа

- Процедура учитава низ непознате дужине.
- Као параметри се задају празан низ  $a$  и празна променљива  $n$  у коју ће се уписати дужина.
- Сазнаје дужину низа тако што пита корисника да је унесе (помоћу `readln`).
- Ако корисник унесе погрешну дужину низа, програм ће му тражити поновни унос.
- Верзија 1 (уvek се исписује иста порука кориснику)

```
PROCEDURE ucitaj(VAR a:niz; VAR n:integer);
VAR i: integer;
BEGIN
    REPEAT
        writeln('Unesite duzinu niza: ');
        readln(n);
    UNTIL (n > 0) AND (n <= MAX_DUZ);
    writeln('Unesite niz: ');
    FOR i:= 1 TO n DO
        read(a[i]);
    END;
```

# Учитавање низа

- Процедура учитава низ непознате дужине.
- Као параметри се задају празан низ  $a$  и празна променљива  $n$  у коју ће се уписати дужина.
- Сазнаје дужину низа тако што пита корисника да је унесе (помоћу `readln`).
- Ако корисник унесе погрешну дужину низа, програм ће му тражити поновни унос.
- Верзија 2 (ако корисник погрешно дужину, испише му се порука грешке и тражи поновни унос)

```
PROCEDURE ucitaj(VAR a:niz; VAR n:integer);
VAR i: integer;
BEGIN
    writeln('Unesite duzinu niza: ');
    readln(n);
    WHILE (n <= 0) OR (n > MAX_DUZ) DO BEGIN
        writeln('Duzina moze biti 1-',MAX_DUZ,'. Unesite opet: ');
        readln(n);
    END;
    writeln('Unesite niz: ');
    FOR i:= 1 TO n DO
        read(a[i]);
    END;
```



# Учитавање низа

- Процедура учитава низ позитивних бројева непознате дужине.
- Као параметри се задају празан низ  $a$  и празна променљива  $n$  у коју ће се уписати дужина.
- Низ се уноси док се не унесе елемент мањи или једнак 0. Дужина низа се сазнаје у току учитавања (не питамо корисника)

```
PROCEDURE ucitaj(VAR a:niz; VAR n:integer);
VAR i, broj: integer;
BEGIN
    writeln('Unesite niz (0 za prekid): ');
    i:= 0; {koliko brojeva ima u nizu}
    read(broj); {cita se prvi broj}
    WHILE (broj>0) AND (n <= MAX_DUZ) DO BEGIN
        i:= i+1;
        a[i]:= broj;
        read(broj); {cita se novi broj}
    END;
    n:= i; {upisujemo u n duzinu niza}
END;
```

# Исписивање низа

- Исписује задати низ.

```
PROCEDURE ispisi(a:niz; n:integer);  
VAR i:integer;  
BEGIN  
    FOR i:= 1 TO n DO  
        write(a[i], ' ');  
    END;
```



# Сортирање низа

- Сортира учитани низ у неоппадајућем поретку

```
PROCEDURE sortiraj(VAR a:niz; n:integer);  
VAR i,j: integer;  
    pom: integer;  
BEGIN  
    FOR i:= 1 TO n-1 DO  
        FOR j:= i+1 TO n DO  
            IF a[i] > a[j] THEN  
                BEGIN  
                    pom:= a[i];  
                    a[i]:= a[j];  
                    a[j]:= pom;  
                END;  
            END;  
        END;  
    END;  
END;
```

# Избацивање задатог елемента

- Процедура оставља у низу елемент који не треба избацити.

```
PROCEDURE izbaci(VAR a:niz; VAR n:integer; broj:integer);  
VAR i,k: integer;  
BEGIN  
    k:= 1;  
    FOR i:= 1 TO n DO  
        IF a[i] <> broj THEN  
            BEGIN  
                a[k]:= a[i];  
                k:= k+1;  
            END;  
        n:= k-1;  
    END;  
END;
```



## Септембар 2011. (1. задатак)

Написати програм на програмском језику Паскал који од два низа целих бројева једнаке дужине формира трећи низ тако да буде задовољен услов  $c[i] = \max(a[i], b[i])$ . Програм треба да са главног улаза учита дужину  $N$  ( $N < 200$ ) и елементе низова, затим формира и испише резултујући низ. Учитавање низа, обраду и испис резултата реализовати као засебне потпрограме који са главним програмом комуницирају искључиво путем аргумената и повратних вредности. Програм треба да понавља описани поступак све док се за дужину низова не унесе некоректна вредност.



## Јануар 2012. (1. задатак)

Написати програм на програмском језику Паскал који врши спајање два неоппадајуће уређена низа целих бројева. Програм најпре треба да учита низове, провери да ли су низови уређени неоппадајуће, а затим спајањем почетних низова формира и испише трећи низ који је уређен на исти начин као почетни низови. Почетни низови могу бити различите дужине (највише 100 елемената). Уколико низови нису уређени неоппадајуће исписати одговарајућу поруку и затражити поновно учитавање почетних низова. Учитавање, проверу уређености и спајање низова реализовати као засебне потпрограме који са главним програмом комуницирају искључиво путем својих аргумената и повратне вредности. Програм треба да понавља претходне кораке све док се за дужину низова не унесе некоректна вредност.



# Матрице

# Учитавање матрице

- Учитава матрицу непознатих димензија и враћа true ако успе или false ако не.

```
FUNCTION ucitaj(VAR mat:matrica; VAR m,n:integer):boolean;  
VAR i,j:integer;  
BEGIN  
    writeln('Unesite dimenzije matrice:');  
    readln(m,n);  
    IF (m>0) AND (m<=MAX_M) AND (n>0) AND (n<=MAX_N) THEN  
        BEGIN  
            writeln('Unesite matricu:');  
            FOR i:= 1 TO m DO BEGIN  
                writeln('Unesite vrstu broj ', i);  
                FOR j:= 1 TO n DO BEGIN  
                    read(mat[i,j]);  
                END;  
            END;  
            ucitaj:= true; {da oznacimo da je ucitavanje uspešno}  
        END  
    ELSE ucitaj:= false; {ucitavanje nije uspešno}  
END;
```



# Испис матрице

- Учитава матрицу непознатих димензија и враћа `true` ако успе или `false` ако не.

```
PROCEDURE ispisi(mat:matrica;m,n:integer);  
VAR i,j:integer;  
BEGIN  
    FOR i:= 1 TO m DO BEGIN  
        FOR j:= 1 TO n DO BEGIN  
            write(mat[i,j], ' ');  
        END;  
        writeln();  
    END;  
END;
```

## Фебруар 2010. (1. задатак)

Написати програм на програмском језику Паскал који обавља следећу операцију над матрицом целих бројева. Програм учита димензије и вредности елемената матрице са главног улаза, а затим испише позицију (ред и колону) и вредност оних елемената матрице који су истовремено најмањи у својој врсти и највећи у својој колони. Матрица не може бити димензија већих од 20x20. Учитавање и обраду матрице реализовати као посебне потпрограме који са главним програмом комуницирају искључиво путем својих аргумената и повратне вредности. Осим исписа података о елементима који задовољавају постављени услов, потпрограм за обраду као резултат враћа број тих елемената, а ту вредност главни програм испише на главном излазу. Програм треба да се извршава све док се за неку од димензија матрице не унесе некоректна вредност.



# Стрингови

## Јун 2006. (2. задатак)

На програмском језику Паскал написати програм који детектује слова абецеде која се нису појавила у учитаном реду. Програм најпре са стандардног улаза чита један ред текста неодређене дужине (од максимално 80 карактера) , који се искључиво састоји од великих слова абецеде. Потом одређује и исписује на стандардном излазу она слова абецеде која се нису појавила у учитаном реду.



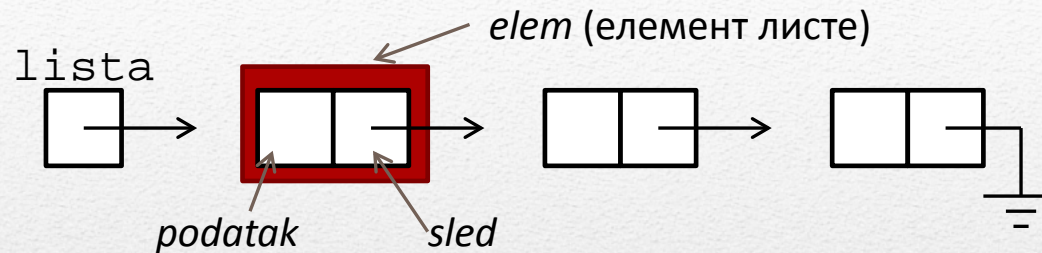
## Октобар 2012. (1. задатак)

Написати програм на програмском језику Паскал који са стандардног улаза чита текст који треба исписати на стандардни излаз, при чему треба изоставити делове текста ограђене малим, округлим заградама. Уређеност текста по редовима треба задржати, при чему треба одбацити оне преласке у нови ред који су део текста који се одбацује. Читање текста завршити када се прочита празан ред. Сматрати да су заграде у улазном тексту правилно упарене.

# Уланчане листе



# Дефиниција уланчане листе



```
TYPE pelem = ^elem;  
    elem = RECORD  
        podatak: TIP;  
        sled: pelem;  
    END;
```

# Додавање на крај листе

```
PROCEDURE dodaj(VAR lista:pelem; podatak:TIP);  
VAR novi,tek:pelem;  
BEGIN  
    new(novi);  
    novi^.podatak:= podatak;  
    novi^.sled:= NIL;  
    IF lista = NIL THEN  
        lista:= novi  
    ELSE BEGIN  
        tek:= lista;  
        WHILE tek^.sled <> NIL DO  
            tek:= tek^.sled;  
        tek^.sled:= novi;  
    END;  
END;
```



# Додавање на почетак

```
PROCEDURE dodajNaPocetak(VAR lista:pelem; podatak:TIP);  
VAR novi,tek:pelem;  
BEGIN  
    new(novi);  
    novi^.podatak:= podatak;  
    {sledbenik je prvi element u staroj listi}  
    novi^.sled:= lista;  
    lista:= novi;  
END;
```

## Испис (обилазак) листе

```
PROCEDURE ispisi(lista:pelem);  
VAR tek:pelem;  
BEGIN  
    tek:= lista;  
    WHILE tek<>NIL DO BEGIN  
        writeln(tek^.podatak);  
        tek:= tek^.sled;  
    END;  
END;
```



# Брисање листе

```
PROCEDURE obrisi(VAR lista:pelem);  
VAR stari,tek:pelem;  
BEGIN  
    tek:=lista;  
    WHILE tek <> NIL DO BEGIN  
        stari:= tek;  
        tek:= tek^.sled;  
        dispose(stari);  
    END;  
    lista:= NIL;  
END;
```

# Сортирање листе

**НИЗ:**

```
PROCEDURE sortiraj(VAR a:niz;  
n:integer);  
VAR i,j: integer;  
    pom: integer;  
BEGIN  
    FOR i:= 1 TO n-1 DO  
        FOR j:= i+1 TO n DO  
            IF a[i] > a[j] THEN  
                BEGIN  
                    pom:= a[i];  
                    a[i]:= a[j];  
                    a[j]:= pom;  
                END;  
            END;  
        END;  
    END;
```

**ЛИСТА:**

```
PROCEDURE sortiraj(VAR lst:pelem);  
VAR tek1,tek2: pelem;  
    pom: integer;  
BEGIN  
    tek1:= lst;  
    WHILE tek1 <> NIL DO BEGIN  
        tek2:= tek1^.sled;  
        WHILE tek2 <> NIL DO BEGIN  
            IF tek1^.pod > tek2^.pod  
            THEN BEGIN  
                pom:= tek1^.podatak;  
                tek1^.pod:= tek2^.pod;  
                tek2^.pod := pom;  
            END;  
            tek2:= tek2^.sled;  
        END;  
        tek1:= tek1^.sled;  
    END;  
END;
```



## Април 2006. (2. задатак)

Написати програм који ради са једноструком уланчаном листом реалних бројева. Са стандардног улаза се прво читају бројеви из једног реда и смештају на крај листе. Затим се врши испис оних бројева из листе чији је трећи степен већи од 904.06, редом којим су учитавани (сваки број у посебном реду). По завршеном испису, ослободити коришћену динамичку меморију. Програм реализовати као главни програм који позива три потпрограма (учитавање, испис и брисање листе).

## Јануар 2006. (1. задатак)

Елемент једноструко уланчане листе, осим показивача на следећи елемент у листи, садржи и један целобројни податак. Написати потпрограм на програмском језику Паскал којим се врши уметање елемента у листу, тако да она буде уређена по неоппадајућем редоследу у односу на вредност целобројног податка. Написати главни програм на програмском језику Паскал који са стандардног улаза чита целе бројеве и умеће их у листу позивом потпрограма све док не дође до краја реда, а затим на стандардном излазу исписује садржај листе. Водити рачуна о правилном ослобађању динамички алоциране меморије на крају програма.



Написати функцију на програмском језику Паскал, чији је аргумент показивач на почетак већ формиране једноструко уланчане листе целих бројева. Функција треба да врати разлику максималног и минималног елемента листе, а ако је листа празна, функција треба да врати нулу. Написати главни програм који позива ову функцију за празну листу и исписује резултат на стандардном излазу.

# Датотеке



# Срећно на испиту!

*Аутор:* Игор Лукић – Змајчики

*Приредила:* Студентска унија Електротехничког факултета  
фебруар 2013.