```
{ *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.
}
PROGRAM zad1_1_9_2011(input,output);
CONST MAX_DUZ = 199;
TYPE niz = array [1..MAX_DUZ] of integer;

{ucitavanje niza nepoznate duzine}
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;

{ispis niza duzine n}
PROCEDURE ispisi(a:niz; n:integer);
VAR i:integer;
BEGIN
   FOR i:= 1 TO n DO
      write(a[i], ' ');
   writeln(); {prelazak u novi red na kraju ispisa svih elemenata}
END;

PROCEDURE obrada(a,b:niz; VAR c:niz; n:integer);
VAR i: integer;
BEGIN
   FOR i:= 1 TO n DO BEGIN
      IF a[i] > b[i] THEN c[i]:= a[i]
      ELSE c[i]:= b[i];
   END;
END;

{glavni program}
VAR a,b,c:niz;
    na,nb:integer;
BEGIN
   WHILE ucitaj(a,na) AND ucitaj(b,nb) AND (na = nb) DO BEGIN
   {nizovi moraju biti ucitani i jadnaki pa se program dotle ponavlja}
      obrada(a,b,c,na); {moglo je i nb}
      writeln('Rezultujuci niz');
      ispisi(c,na); {niz c je takodje duzine na=nb}
   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.
}
PROGRAM zad1_1_9_2011_ver2(input,output);

CONST MAX_DUZ = 199;
TYPE niz = array [1..MAX_DUZ] of integer;

{ucitavanje niza poznate duzine}
PROCEDURE ucitaj(VAR a:niz; n:integer);
VAR i: integer;
BEGIN
   writeln('Unesite niz: ');
   FOR i:= 1 TO n DO
      read(a[i]);
END;

{ispis niza duzine n}
PROCEDURE ispisi(a:niz; n:integer);
VAR i:integer;
BEGIN
   writeln('Niz: ');
   FOR i:= 1 TO n DO
      write(a[i], ' ');
   writeln();
END;

PROCEDURE obrada(a,b:niz; VAR c:niz; n:integer);
VAR i: integer;
BEGIN
   FOR i:= 1 TO n DO BEGIN
      IF a[i] > b[i] THEN c[i]:= a[i]
      ELSE c[i]:= b[i];
   END;
END;

{glavni program}
VAR a,b,c:niz;
    n:integer;
BEGIN
   writeln('Unesite duzinu nizova: ');
   readln(n);
   WHILE (n>0) AND (n<MAX_DUZ) DO BEGIN
      ucitaj(a,n); ucitaj(b,n);
      obrada(a,b,c,n);
      ispisi(c,n);
      {duzina za sledeci prolazak}
      writeln('Unesite duzinu nizova: ');
      readln(n);
   END;
END.
```

1

```
{ *Januar 2012. (1. zadatak)*
  Napisati program na programskom jeziku Paskal koji vrši spajanje dva neopadajuće uređena
  niza celih brojeva. Program najpre treba da učita nizove, proveri da li su nizovi
  uređeni neopadajuće, a zatim spajanjem početnih nizova formira i ispiše treći niz koji
  je uređen na isti način kao početni nizovi. Početni nizovi mogu biti različite dužine
  (najviše 100 elemenata). Ukolико nizovi nisu uređeni neopadajuće ispisati odgovarajuću
  poruku i zatražiti ponovno učitavanje početnih nizova. Učitavanje, proveru uređenosti i
  spajanje nizova realizovati kao zasebne potprograme koji sa glavnim programom komuniciraju
  isključivo putem svojih argumenata i povratne vrednosti. Program treba da ponavlja
  pretходne korake sve dok se za dužinu nizova ne unese nekorektna vrednost.
}
PROGRAM zad1_18_1_2012(input,output);

CONST MAX_DUZ = 100;

TYPE niz100 = array [1..MAX_DUZ] of integer;
     niz200 = array [1..MAX_DUZ*2] of integer;

FUNCTION ucitaj(VAR a:niz100; 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;

FUNCTION daLiJeNeopadajuci(a:niz100; n:integer):boolean;
VAR i: integer;
BEGIN
   daLiJeNeopadajuci:= true;
   {uporedjujemo parove}
   FOR i:= 1 TO n-1 DO
      IF (a[i+1] < a[i]) THEN
         daLiJeNeopadajuci:= false;
END;

PROCEDURE ispisi(a:niz200; n:integer);
VAR i:integer;
BEGIN
   FOR i:= 1 TO n DO
      write(a[i], ' ');
   writeln();
END;
```



```
PROCEDURE spoji(a:niz100; na:integer; b:niz100; nb:integer; VAR c:niz200; VAR nc:integer);
VAR i,j,k:integer;
BEGIN
   i:= 1; j:= 1; k:= 1;
   WHILE (i<=na) AND (j<=nb) DO BEGIN
      IF (a[i] < b[j]) THEN BEGIN
         c[k]:= a[i];
         i:= i+1;
      END
      ELSE BEGIN
         c[k]:= b[j];
         j:= j+1;
      END;
      k:= k+1;
   END;

   {prepisujemo onaj niz koji je preostao}
   WHILE (i<=na) DO BEGIN
      c[k]:= a[i];
      i:= i+1; k:= k+1;
   END;
   WHILE (j<=nb) DO BEGIN
      c[k]:= b[j];
      j:= j+1; k:= k+1;
   END;
   nc:= na+nb;
END;

VAR a,b:niz100;
    c:niz200;
    na,nb,nc:integer;
    uspelo1,uspelo2:boolean;
BEGIN
   REPEAT
      uspelo1:= ucitaj(a,na);
      WHILE uspelo1 AND NOT daLiJeNeopadajuci(a,na) DO BEGIN
          writeln('Niz mora biti neopadajuci!');
          uspelo1:= ucitaj(a,na);
      END;

      uspelo2:= ucitaj(b,nb);
      WHILE uspelo2 AND NOT daLiJeNeopadajuci(b,nb) DO BEGIN
          writeln('Niz mora biti neopadajuci!');
          uspelo2:= ucitaj(b,nb);
      END;

      IF uspelo1 AND uspelo2 THEN
      BEGIN
         spoji(a,na,b,nb,c,nc);
         ispisi(c,nc);
      END;
   UNTIL NOT(uspelo1 AND uspelo2);
END.
```

2

```
{ *Април 2006. (2. задатак)*
  Написати програм који ради са једноструко 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).
}
PROGRAM zad2_9_4_2006;
CONST GRANICA = 904.06;
TYPE pelem = ^elem;
     elem = RECORD
        info: real;
        sled: pelem;
     END;

{dodavanje jednog elementa na kraj neke ulancane liste}
PROCEDURE dodaj(VAR lista:pelem; broj:real);
VAR novi,tek:pelem;
BEGIN
    new(novi);
    novi^.info:= broj;
    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;

{ucitava listu uz pomoc procedure za dodavanje jednog elementa}
PROCEDURE ucitaj(VAR lista:pelem);
VAR broj:real;
BEGIN
    {radimo ucitavanje i dodavanje dokle god u redu imamo brojeve
    koji su neprocitani (dok ne dodjemo do kraja reda)}
    WHILE NOT eoln(input) DO BEGIN
        read(broj);
        dodaj(lista,broj);
    END;
END;

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;
```



```
PROCEDURE obrada(lista:pelem);
VAR tek:pelem;
BEGIN
    tek:= lista;
    WHILE tek <> NIL DO BEGIN
        IF (tek^.info * tek^.info * tek^.info) > GRANICA THEN
            writeln(tek^.info);
        tek:= tek^.sled;
    END;
END;

VAR lista:pelem;
BEGIN
    lista:= NIL;
    ucitaj(lista);
    obrada(lista);
    obrisi(lista);
END.
```

3

```
{ *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 неопадајућем 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.
}
PROGRAM zad1_1_2_2006;
TYPE pelem = ^elem;
     elem = RECORD
        info: integer;
        sled: pelem;
     END;

PROCEDURE dodaj(VAR lista:pelem; broj:integer);
VAR novi,tek,pret:pelem;
BEGIN
    {trazimo gde da umetnemo novi element}
    tek:= lista; pret:=nil;
    WHILE (tek <> NIL) AND (broj < tek^.info) DO BEGIN
        pret:= tek;
        tek:= tek^.sled; {krecemo se kroz listu dokle god je broj
                          koji dodajemo manji od tekuceg elementa
                          liste, dok ne nadjemo prvi veci element}
    END;
    {pravimo novi element}
    new(novi);
    novi^.info:= broj;
    novi^.sled:= tek; {dodajemo ispred tekuceg, dakle on je
                       sledbenik novog elementa}
    IF tek = lista THEN {ako dodajemo ispred prvog}
        lista:= novi
    ELSE
        pret^.sled:= novi;
END;

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



```
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;

VAR lista:pelem;
    broj:integer;
BEGIN
    lista:= NIL;
    WHILE NOT eoln(input) DO BEGIN
        read(broj);
        dodaj(lista,broj);
    END;
    ispisi(lista); {ovo smo vec pisali ranije}
    obrisi(lista); {i ovo vec ima napisano}
END.
```

4

```
{
  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.
}
PROGRAM zad2_14_2_2009;
TYPE pelem = ^elem;
     elem = RECORD
        info: integer;
        sled: pelem;
     END;

FUNCTION razlika(lista:pelem):integer;
VAR min, max: integer;
BEGIN
    IF lista <> NIL THEN {ako lista nije prazna}
    BEGIN
        {prvi element prograsavamo minimalnim i maksimalnim}
        min:= lista^.info;
        max:= lista^.info;
        {trazimo elemente koji su veci od max i manji od min}
        tek:= lista;
        WHILE tek <> NIL DO BEGIN
            IF tek^.info > max THEN
                max:= tek^.info;
            IF tek^.info < min THEN
                min:= tek^.info;
            tek:= tek^.sled;
        END;
            razlika:= max-min;
    END
    ELSE {ako je prazna}
        razlika:=0;
END;

VAR lista:pelem;
BEGIN
    lista:= NIL; {lista je prazna}
    writeln( razlika(lista) );
END.
```

Zadatak 2 (13.02.2009)
U tekst datoteci ulaz.txt nalaze se pozitivni celi brojevi manji od 200. Broj redova u datoteci kao i broj brojeva u svakom redu nije unapred poznat. Napisati program na programskom jeziku Pascal koji iz ove datoteke čita brojeve i u izlaznu tekst datoteku izlaz.txt prepiše svaki pročitan broj uz očuvanje rasporeda brojeva po redovima, uz ograničenje da se broj ne prepisuje ako se prethodno pojavio u datom redu. Drugim rečima, u jednom redu izlazne datoteke svi brojevi treba da budu različiti. Za pamćenje unetih brojeva koristiti skupovni tip podataka.

(po šablonu)

```
PROGRAM zad2_13_02_2009 (ulaz, izlaz);

VAR ulaz, izlaz: text;
    broj: integer;
    red: set of 1..200;

BEGIN
   assign(ulaz, 'ulaz.txt');
   reset(ulaz);
   assign(izlaz, 'izlaz.txt');
   rewrite(izlaz);

   while not eof(ulaz) do begin

      red:=[];
      while not eoln(ulaz) do begin {dok se ne dođe do kraja reda}
         read(ulaz, broj);
         IF NOT (broj IN red) THEN {ako se broj nije pojavio}
            write(izlaz, broj);
         red:= red+[broj]; {dodajemo broj u skup onih koji su se
                            pojavili}
      end;
      readln(ulaz); {kada se dođe do kraja reda prelazi se
                     u novi red da bi se nastavilo čitanje}
      writeln(izlaz); {a tokdje se pisanje prebaci u novi red}
   end;

   close(ulaz);
   close(izlaz);
END.
```


Zadatak 1 (07.03.2005)
Data je tekstualna datoteka ispit.txt koja je formirana na osnovu ocena studenata na predmetu Programiranje 1. Svaki red u datoteci sadrži podatke za jednog studenta, i to: prezime i ime (svako od polja je niz od najviše 40 karaktera), podatak da li se ispit polaže integralno ili preko kolokvijuma (jedan karakter: 'i' odnosno 'k' respektivno) i ocena na ispitu (ceo broj od 5 do 10). Podaci (prezime, ime, način polaganja i ocena) su međusobno razmaknuti sa jednim ili više blanko znakova. Potrebno je izračunati statistiku prolaznosti na ispitu i na glavnom izlazu ispisati ukupan broj studenata, broj i procenat studenata koji su položili ispit (ocena veća od 5) i broj i procenat studenata koji su ostvarili ocenu 10. Napisati program na programskom jeziku Pascal koji obavlja opisanu obradu.

Primer izgleda datoteke ispit.txt:
Petrovic Petar k 8
Markovic Marko i 7

(po šablonu)

```
PROGRAM zad1_7_3_2005 (ispit, output);

TYPE student = RECORD
         ime, prezime: string[40];
         nacinPolaganja: char;
         ocena: 5..10;
     END;

{procedura za citanje jedne reci, opisana gore}
PROCEDURE citajRec(VAR dat:TEXT; VAR rec:STRING);
VAR c:CHAR;
BEGIN
    rec := '';
    read(dat,c);
    while (c <> ' ') do begin
         rec := rec + c;
         read(dat,c);
    end;
END;

VAR ispit: text;
    s: student;
    ukupno, polozili, deset: integer;

za sve datoteke
BEGIN
   assign(ispit, 'ispit.txt');
   reset(ispit);

   {postavljamo bojace na 0}
   ukupno:= 0;
   polozili:= 0;
   deset:= 0;
```

```
   while not eof(ispit) do begin
      citajRec(ispit, s.prezime);
      citajRec(ispit, s.ime);
      readln(ispit, s.nacinPolaganja, s.ocena);

      ukupno:= ukupno + 1;
      if s.ocena > 5 then polozili:= polozili + 1;
      if s.ocena = 10 then deset:= deset + 1;
   end;

   writeln('Statistika');
   writeln('---------------------');
   writeln('Ukupno: ', ukupno);
   writeln('Polozioli: ', polozili, ', ', polozili/ukupno, '%');
   writeln('Desetke: ', deset, ', ', deset/ukupno, '%');
   writeln('---------------------');

   close(ispit);
END.
```

Zadatak 1 (09.06.2011)
Neka se u datoteci biblio.txt nalaze podaci o knjigama koje poseduje neka biblioteka. Svaki red sadrži šifru knjige (ceo broj), broj raspoloživih kopija (ceo broj) i naslov knjige do kraja reda (ne duži od 255 karaktera). U datoteci izdato.txt se nalaze podaci o knjigama koje su izdate na čitanje po sledećem formatu: šifra knjige (ceo broj) i šifra korisnika (ceo broj) koji je pozajmio knjigu na čitanje. Napisati program na programskom jeziku Pascal koji pročita sadržaj navedenih datoteka i na glavnom izlazu ispiše naslove svih onih knjiga čije su sve kopije izdate na čitanje korisnicima. Voditi računa o ispravnom korišćenju zauzetih resursa.

```
PROGRAM zad1_9_6_2011(biblio, izdat, output);

TYPE knjiga = RECORD
         sifra, brojKopija: integer;
         naslov: string[255];
     END;

     pelem = ^elem;
     elem = RECORD
         k: knjiga;
         sled: pelem;
     END;


PROCEDURE ucitaj(VAR dat: text; VAR lista:pelem);
VAR k:knjiga;
    novi,posl:pelem;
BEGIN
   WHILE NOT eof(dat) DO BEGIN
      readln(dat, k.sifra, k.brojKopija, k.naslov);
      new(novi);
      novi^.k:= k;
      novi^.sled:= NIL;
      IF lista = NIL THEN
         lista:= novi
      ELSE
         posl^.sled:= novi;
      posl:= novi;
   END;
END;

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.
```

```
PROCEDURE izdaj(lista:pelem; sifraKnjige:integer);
VAR tek:pelem;
BEGIN
   tek:= lista;
   WHILE tek <> NIL DO BEGIN
      IF tek^.k.sifra = sifraKnjige THEN
         {smanjujemo broj kopija, ako jos uvek ima kopija}
         IF tek^.k.brojKopija > 0 THEN
            tek^.k.brojKopija:= tek^.k.brojKopija - 1;

      tek:= tek^.sled;
   END;
END;

PROCEDURE ispisiIzdate(lista:pelem);
VAR tek:pelem;
BEGIN
   tek:= lista;
   WHILE tek <> NIL DO BEGIN
      IF tek^.k.brojKopija = 0 THEN
         writeln(output, tek^.k.naziv);
      tek:= tek^.sled;
   END;
END;

VAR knjige: pelem; {lista knjiga}
    biblio, izdat: text;
    sifraKnjige: integer;

BEGIN
   assign(biblio, 'biblio.txt');
   reset(biblio);

   assign(izdato, 'izdato.txt');
   reset(izdato);

   knjige:= NIL;
   ucitaj(biblio, knjige);

   WHILE NOT eof(izdato) DO BEGIN
      readln(izdato, sifraKnjige); {iako u ovoj datoteci pored
                                    sifre knjige postoji i
                                    sifra korisnika ona mi nije
                                    potrebna pa je necu ni citati
                                    vec ce readln automastki
                                    da predje u novi red}
      izdaj(knjige, sifraKnjige);
   END;

   {ispisujemo knjige kod kojih je stanje nakon azuriranja 0}
   ispisiIzdate(knjige);
   obrisi(knjige);

   close(biblio);
   close(izdato);
END.
```

Zadatak 1 (09.02.2011)
Napisati program na programskom jeziku Pascal koji vrši raspoređivanje studenata po salama za polaganje kolokvijuma. U jednom redu datoteke studenti.txt se nalazi broj indeksa (u formatu gggg/bbbb) i ime i prezime studenta, a u jednom redu datoteke sale.txt se nalaze podaci o nazivu slobodne sale (niz znakova od najviše 10 karaktera) i kapacitetu (ceo broj). Program treba da formira izlaznu datoteku raspored.txt u čijem će se svakom redu nalaziti indeks, ime i prezime studenta i broj sale u kojoj polaže kolokvijum. Ukoliko zbog nedovoljnog kapaciteta program ne može da rasporedi studente ni u jednu salu, ispisati u izlaznoj datoteci da su neraspoređeni (npr. 2009/0161 Laza Lazic nerasporedjen).

studenti.txt
2007/0156 Pera Petrovic
2008/0158 Mile Milic
2009/0159 Aca Acic

sale.txt
M208 2
M216 3

raspored.txt
2007/0156 Pera Petrovic M208
2008/0158 Mile Milic M208
2009/0159 Aca Acic M216

```
PROGRAM zad1_9_2_2011 (studenti, sale, raspored);

TYPE pod_student = RECORD
         indeks: string[9];
         ime, prezime: string[30];
     END;

     pod_sala = RECORD
         sifra: string[10];
         mesta: integer;
     END;

PROCEDURE citajRec(VAR dat:TEXT; VAR rec:STRING);
VAR c:CHAR;
BEGIN
    rec := '';
    read(dat,c);
    while (c <> ' ') do begin
         rec := rec + c;
         read(dat,c);
    end;
END;
```

```
VAR studenti, sale, raspored: text;
    student: pod_student;
    sala: pod_sala;

BEGIN
   assign(studenti, 'studenti.txt');
   reset(studenti);

   assign(sale, 'sale.txt');
   reset(sale);

   assign(raspored, 'raspored.txt');
   rewrite(raspored);

   WHILE NOT eof(sale) DO BEGIN
      citajRec(sale, sala.sifra);
      readln(sale, sala.mesta);

      WHILE (sala.mesta > 0) AND (NOT eof(studenti)) DO BEGIN
         read(studenti, student.indeks);
         citajRec(studenti, student.ime); {posto je rec u sredini}
         readln(studenti, student.prezime);
         writeln(raspored, student.indeks, student.ime,
                 student.prezime, sala.sifra);
         sala.mesta:= sala.mesta+1;
      END;
   END;

   {ako je ostalo jos studenata}
   WHILE NOT eof(studenti) DO BEGIN
      read(studenti, student.indeks);
      citajRec(studenti, student.ime); {posto je rec u sredini}
      readln(studenti, student.prezime);

      writeln(raspored, student.indeks, student.ime,
              student.prezime, 'nerasporedjen');
   END;

   close(studenti);
   close(sale);
   close(raspored);

END.
```