
Latihan Soal

Olimpiade Komputer

Oleh :

dwi sakethi, s.si, m.kom

<http://dwijim.staff.unila.ac.id>

dwijim@unila.ac.id

0816 403 432

pengrajin teknologi informasi

LEMBAH (JANGAN) BANJIR

TANJUNG SENENG 2006

1 Kata Pengantar

Puji yang sejati hanya untuk *Allah* Yang Maha Tinggi, puja yang sempurna hanya untuk *Allah* Yang Maha Kuasa. *Sholawat* dan *salam* semoga senantiasa dilimpahkan kepada tauladan semua yang mengaku merupakan himpunan bagian dari kelompok manusia, Muhammad saw., keluarganya, sahabatnya, pengikutnya dan seluruh muslimin kapan dan dimana *bae*.

Tulisan ini disusun sebagai rangkaian kegiatan ketika membina tim olimpiade komputer yang dimulai dengan pembinaan terhadap SMA Negeri 1 Gading Rejo Lampung Tengah. Untuk itu saya sangat berterima kasih kepada Bapak Jumiran dan anak-anak SMA Negeri 1 Gading Rejo waktu itu, yang mohon maaf sekarang saya sudah lupa lagi ... :-) Kegiatan itu kalau saya tidak salah dilakukan tahun 2004. Kemudian berlanjut membina tim olimpiade komputer SMA Negeri 2 Bandar Lampung pada bulan Agustus 2006.

Semoga ini menjadi bagian amal kebaikan yang dapat mendorong kesejahteraan di bumi pertiwi ini ... Sehingga rakyat selalu mendendangkan lagu **Disini senang, di sana senang**, bukan lagu **Padamu Negeri**.

dwi sakethi
dwijim@unila.ac.id
<http://dwijim.staff.unila.ac.id>
hp : 0816 403 432 (nomor cantik ya ...)
pengrajin teknologi informasi

Daftar Isi

1	Kata Pengantar	i
2	Reverse	2
2.1	Masukan	2
2.2	Keluaran	2
2.3	Contoh Penyelesaian	2
2.4	Contoh hasilnya	3
3	Menghitung Massa	3
3.1	Format Masukan	4
3.2	Format Keluaran	4
3.3	Contoh penyelesain	4
3.4	Contoh hasil <i>running</i>	9
4	Ekspresi Aljabar	10
4.1	Masukan	10
4.2	Keluaran	11
4.3	Contoh Penyelesaian	12
4.4	Contoh hasil <i>running</i>	15
5	MULTIPALINDROM	16
5.1	Format Masukan	16
5.2	Format Keluaran	16
5.3	Contoh Penyelesaian	17
5.4	Hasil Program	19
6	Menghitung Jumlah Huruf	21
6.1	Contoh Penyelesain	21
6.2	Contoh Keluaran	23
7	Variasi Masalah Bilangan Prima	24
7.1	Contoh Penyelesain	24
7.2	Contoh Keluaran	26

2 Reverse

Wah ... mudah amat ya ... soal olimpiade komputer. Yah ... ini memang soal pemanasan dan pengenalan, jadi sangat mudah. Soalnya sebagai berikut :

Buatlah program REVERSE.PAS menurut penjelasan berikut ini. Sebagai latihan, Anda belajar menentukan cara bagaimana membaca masukan string yang sangat panjang.

2.1 Masukan

Program itu harus membaca masukan dari file bernama REVERSE.IN. File ini akan berisikan satu baris teks dengan *panjang* ≤ 1000 karakter. (*Hint* : ini lebih panjang dari panjang maksimum string di Pascal).

2.2 Keluaran

Program harus menuliskan keluaran dalam file REVERSE.OUT. Keluaran adalah hanya satu baris teks yaitu string hasil pembalikan yang telah anda lakukan dari string masukan.

reverse.in

Olimpiade Nasional

reverse.out

lanoisaN edaipmilO

2.3 Contoh Penyelesaian

Penyelesaian : Berikut ini salah satu contoh penyelesaian masalah di atas. Masih banyak cara yang bisa dilakukan untuk menyelesaikannya.

```
program membalik_deretan_karakter;
uses crt;
var i,jumlah : integer;
    file_data : text;
    karakter : char;
    data_asal : array[1..1000] of char;
begin
  clrscr;
  jumlah := 0;
  assign(file_data,'data.txt');
  reset(file_data);
  writeln('data reverse');
  writeln('-----');
  while not eof (file_data) do
  begin
    jumlah := jumlah + 1;
    read(file_data,karakter);
    write(karakter);
    data_asal[jumlah] := karakter;
```

```

        end;
    close(file_data);
    writeln;
    writeln('-----');
    writeln('hasil reverse');
    writeln('-----');
    for i:= jumlah downto 1 do write(data_asal[i]);
    readln;
end.

```

2.4 Contoh hasilnya

```

data reverse
-----
program membalik_deretan_karakter; salam keadilan dan kesejahteraan untuk rakyat
di bumi pertiwi. supaya yakin maka tulisan ini dibuat lebih dari 255 karakter.
apakah anda sudah mempersiapkan bekal untuk kehidupan setelah kematian nanti ? d
wi sakethi pengrajin teknologi informasi 0816 403 432

-----
hasil reverse
-----

234 304 6180 isamrofni igolonket nijargnep ihtekas iwd ? itnan naitamek haletes
napudihek kutnu lakeb nakpaisrepmem hadus adna hakapa .retkarak 552 irad hibel t
aubid ini nasilut akam nikay ayapus .iwitrep imub id taykar kutnu naarethajesek
nad nalidaek malas ;retkarak_natered_kilabmem margorp

```

3 Menghitung Massa

Suatu molekul terdiri atas sejumlah atom dan tersusun membentuk rumus kimia yang dituliskan dengan huruf-huruf yang menyatakan masing-masing atom ini. Misalnya H menyatakan atom hidrogen, C menyatakan atom karbon, O menyatakan atom oksigen. Jadi rumus kimia COOH menyatakan suatu molekul yang berisikan satu atom karbon, dua atom oksigen dan satu atom hidrogen.

Untuk menuliskan rumus ini secara efisien kita menggunakan aturan-aturan berikut ini.

Huruf-huruf yang menyatakan beberapa atom dapat dikelompokkan dengan pembatas tanda kurung yang disebut juga dengan istilah gugus atom. Misalnya rumus CH(OH) berisi gugus OH. Dalam suatu gugus bisa terdapat gugus-gugus lebih kecil. Untuk menyederhanakan suatu rumus kimia, kemunculan sejumlah huruf secara berturut-turut dapat digantikan dengan satu huruf saja tapi diikuti oleh suatu bilangan yang menyebutkan jumlah kemunculannya. Misalnya huruf COOHHH dapat ditulis sebagai CO₂H₃ dan ia mempresentasikan suatu molekul yang berisikan satu atom karbon, dua atom oksigen dan tiga atom hidrogen.

Selanjutnya, kemunculannya yang berturut-turut dari gugus yang sama dapat digantikan dengan gugus tersebut diikuti oleh bilangan yang menyatakan

jumlah kemunculan gugus tersebut. Misalnya $\text{CH}(\text{CO}_2\text{H})(\text{CO}_2\text{H})(\text{CO}_2\text{H})$ dapat dituliskan sebagai $\text{CH}(\text{CO}_2\text{H})_3$ dan molekul tersebut berisikan empat atom karbon, dua atom oksigen dan tiga atom hidrogen.

Dalam rumus kimia sebenarnya tentu bilangan yang menyatakan pengulangan kemunculan suatu gugus atom tersebut bisa berharga berapapun asal ≥ 1 . Dalam soal di sini bilangan tersebut dibatasi sampai dengan 9.

Massa dari suatu molekul adalah jumlah dari massa dari setiap atom yang tergantung di dalamnya. Satu atom hidrogen memiliki massa satu, satu atom karbon memiliki massa 12 dan satu atom oksigen memiliki massa 16.

Tuliskan suatu program dengan nama MASSA.PAS yang dapat menghitung massa molekul dari rumus molekul yang diberikan.

3.1 Format Masukan

File masukan adalah file teks dengan nama MASSA.IN. File berisi satu baris yang didalamnya tertuliskan rumus molekul yang hendak dihitung massanya. Rumus molekul hanya akan berisikan kemungkinan karakter-karakter H, C, O, (,), 2, 3, ..., 9. Panjang string tidak akan lebih dari 100 karakter.

3.2 Format Keluaran

File keluaran adalah file teks dengan nama MASSA.OUT. Satu-satunya baris keluaran hanya berisikan massa dari molekul yang dinyatakan dengan rumus yang diberikan. Bilangan massa tidak akan lebih besar dari 1000 karakter.

3.3 Contoh penyelesaian

```
uses crt;
var hasil_kurung_total, hasil_kurung, hasil_akhir, kimia : string;
    p1, p : string;
    massa, tingkatx, k, i, j, jumlah_kurung : integer;
    akhir_sebelum, awal_sebelum, panjang, tingkat : integer;
    jumlah_asal, sekarang, kode_error : integer;
    awal, akhir, tingkat_kurung_awal, sudah_dihitung : array[1..25] of integer;
    pengali, kurung_punya, tingkat_kurung_akhir, kepakai : array[1..25] of integer;
    ada_kurung, selesai : boolean;
begin
    clrscr;
{ -----
    silahkan ganti-ganti bentuk kimia sesuai yang dikehendaki
    ----- }
    kimia := 'O(CO2H)3';
    kimia := 'CH(CO2H)3';
    kimia := '((CH)2(OH2H)(C(H))O)3';
    kimia := 'COOH';
```

```

writeln('bentuk asal : ',kimia);
hasil_akhir := '';
panjang := length(kimia);
tingkat := 0;
jumlah_kurung := 0;
for i:=1 to panjang do
begin
p := copy(kimia,i,1);
kepakai[i] := 0;
sudah_dihitung[i] := 0;
tingkat_kurung_akhir[i]:=0;
if p='(' then
begin
inc(tingkat);
inc(jumlah_kurung);
awal[jumlah_kurung] := i;
tingkat_kurung_awal[jumlah_kurung] := tingkat;
akhir[jumlah_kurung] := 0;
kurung_punya[i] := jumlah_kurung;
end
else
if p=')' then
begin
kurung_punya[i] :=0;
tingkat_kurung_akhir[i] := tingkat_kurung_awal[jumlah_kurung];
akhir[jumlah_kurung] := i;
dec(tingkat);
end;
end;
{ mencari posisi kurung tutup }
tingkatx := 0;
jumlah_kurung := 0;
for i:=1 to panjang do
begin
p := copy(kimia,i,1);
if p='(' then
begin
inc(tingkatx);
inc(jumlah_kurung);
end
else
if p=')' then
begin
for j:=i downto 1 do
begin
p1 := copy(kimia,j,1);
if p1='(' then
begin
if kepakai[j]=0 then
begin

```

```

                                akhir[kurung_punya[j]] := i;
                                kepakai[j] := 1;
                                j := 1;
                                end;
                                end;
                                end;
                                dec(tingkatx);
                                end;
                                end;
for i:=1 to jumlah_kurung do
begin
    write('tingkat : ',tingkat_kurung_awal[i],
        ' mulai ',awal[i], ' sampai ',akhir[i]);
    p:=copy(kimia,akhir[i]+1,1); { mencari faktor pengali untuk }
    val(p,k,kode_error);         { masing-masing kurung      }
    if (k=0) then k:=1;
    pengali[i] :=k;
    writeln(' : ',pengali[i]);
end;
for i:=1 to jumlah_kurung do kepakai[i]:=0;
{ kepakai di sini digunakan untuk menandai bahwa
  suatu karakter sudah dihitung }

selesai := false;
hasil_kurung_total := '';
repeat
    sekarang := 0;
    for i:=1 to jumlah_kurung do
        begin
            if (tingkat_kurung_awal[i]>sekarang) and (kepakai[i]=0)
            then sekarang:=i;
            { akan memproses tingkat kurung tertinggi dan
              pada posisi itu memang belum diproses }

            end;
        write('kurung ke-',kurung_punya[awal[sekarang]], ' ');
        write('tingkat ',tingkat_kurung_awal[sekarang], ' : ');
        kepakai[sekarang]:=1; { menandai bahwa posisi ini sudah
                               diproses }

        hasil_kurung := '';
        for i:=awal[sekarang] to akhir[sekarang] do
            begin
                p := copy(kimia,i,1);
                if ((p='C') or (p='H') or (p='O')) and (sudah_dihitung[i]=0) then
                    begin
                        sudah_dihitung[i]:=1;
                        p1 := copy(kimia,i+1,1);
                        val(p1,k,kode_error);
                        if (k>=2) then
                            for j:=1 to k do hasil_kurung:=hasil_kurung + p

```



```

        else
            hasil_kurung := hasil_kurung + p;
        end;
    end;

    { mencari apakah di dalam kurung yang sekarang
      ada kurung lagi di dalamnya }
    ada_kurung := false;
    for i:=awal[sekarang]-1 downto 1 do
        begin
            p:=copy(kimia,i,1);
            if (p='(') then
                begin
                    awal_sebelum := awal[kurung_punya[i]];
                    akhir_sebelum := akhir[kurung_punya[i]];
                    if (awal_sebelum<awal[sekarang]) then
                        if (akhir_sebelum>akhir[sekarang]) then
                            begin
                                write('awal : ',awal_sebelum,' - ',awal[sekarang], ' ');
                                write('ada di dalam kurung', ' ');
                                ada_kurung := true;
                                i := 1;
                            end;
                        end;
                    end;
                end;
            end;

        { hasil yang didapat, dikalikan dengan pengali untuk
          masing-masing kurung }
        k := pengali[kurung_punya[awal[sekarang]]];
        hasil_kurung_total := '';
        if (awal[sekarang]<>1) and (akhir[sekarang]<>panjang-1) then
            for i:=1 to k do
                begin
                    hasil_kurung_total:= hasil_kurung_total+ hasil_kurung;
                end
            else
                hasil_kurung_total:= hasil_kurung_total+ hasil_kurung;
            writeln('pengali : ',k);
            writeln('hasil kurung : ',hasil_kurung_total);
            hasil_akhir := hasil_akhir + hasil_kurung_total;
            writeln('hasil akhir: ',hasil_akhir);
        sekarang := 0;

        { kalau semua posisi sudah diproses artinya kepakai[i]=1
          berarti proses selesai }
        selesai := true;
        for i:=1 to jumlah_kurung do
            if kepakai[i]=0 then
                begin
                    selesai := false;

```

```

        i := jumlah_kurung;
    end;
until selesai;

panjang := length(kimia);
hasil_kurung_total := hasil_akhir;
if not ada_kurung then
    hasil_akhir := ''
else
    hasil_akhir := hasil_akhir;

{ untukantisipasi bentuk khusus dimana
  kurung terakhir di kolom terakhir-1, tapi
  kurung awalnya di kolom 1 }
k := pengali[tingkat_kurung_awal[1]];
if (awal[1]=1) and (akhir[1]=panjang-1) then
    for i:=1 to k do hasil_akhir := hasil_akhir + hasil_kurung_total;

{ untukantisipasi bentuk khusus dimana
  kurung terakhir di kolom terakhir-1, tapi
  kurung awalnya bukan di kolom 1 }

if (awal[1]<>1) and (akhir[1]=panjang-1) then
    for i:=1 to k do hasil_akhir := hasil_akhir + hasil_kurung_total;

{
  kalau-kalau ada yang belum dihitung,
  artinya nilai sudah_dihitung=0
  ini terjadi kalau tidak ada kurung sama sekali }
for i:=1 to awal[1] do
    if sudah_dihitung[i]=0 then
        begin
            p := copy(kimia,i,1);
            if (p='C') or (p='H') or (p='O') then
                hasil_akhir := hasil_akhir + p;
        end;

writeln('hasil akhirnya : ',hasil_akhir);
panjang := length(hasil_akhir);
massa := 0;
for i:=1 to panjang do
    begin
        p := copy(hasil_akhir,i,1);
        if p='H' then
            massa:=massa+1
        else
            if p='C' then
                massa:=massa+12
            else
                massa:=massa+16;
    end;
end;

```

```

        end;
writeln('Massa : ',massa);
readln;
end.

```

3.4 Contoh hasil *running*

```

bentuk asal : COOH
kurung ke-0 tingkat -28666 : pengali : 1280
hasil kurung :
hasil akhir:
hasil akhirnya : COOH
Massa : 45

```

```

bentuk asal : CH(CO2H)3
tingkat : 1 mulai 3 sampai 8 : 3
kurung ke-1 tingkat 1 : pengali : 3
hasil kurung : COOH
hasil akhir: COOH
hasil akhirnya : COOHCOOHCOOHCH
Massa : 148

```

```

bentuk asal : ((CH)2(OH2H)(C(H))O)3
tingkat : 1 mulai 1 sampai 20 : 3
tingkat : 2 mulai 2 sampai 5 : 2
tingkat : 2 mulai 7 sampai 12 : 1
tingkat : 2 mulai 13 sampai 18 : 1
tingkat : 3 mulai 15 sampai 17 : 1
kurung ke-5 tingkat 3 : awal : 13 - 15 ada di dalam kurung pengali : 1
hasil kurung : H
hasil akhir: H
kurung ke-2 tingkat 2 : awal : 1 - 2 ada di dalam kurung pengali : 2
hasil kurung : CHCH
hasil akhir: HCHCH
kurung ke-3 tingkat 2 : awal : 1 - 7 ada di dalam kurung pengali : 1
hasil kurung : OHHH
hasil akhir: HCHCHOHHH
kurung ke-4 tingkat 2 : awal : 1 - 13 ada di dalam kurung pengali : 1
hasil kurung : C
hasil akhir: HCHCHOHHHC
kurung ke-1 tingkat 1 : pengali : 3
hasil kurung : O
hasil akhir: HCHCHOHHHCO
hasil akhirnya : HCHCHOHHHCOHCHCHOHHHCOHCHCHOHHHCO
Massa : 222

```

Tapi mohon maaf sebesar-besarnya ... ternyata program tersebut belum dapat menyelesaikan :-) ... bentuk seperti ini misalnya :

```

bentuk asal : CO2H
kurung ke-0 tingkat -28666 : pengali : 1280

```

```
hasil kurung :  
hasil akhir:  
hasil akhirnya : COH  
Massa : 29
```

Mengapa demikian ? Ya ... ini menjadi PR lanjutan untuk Anda yang masih penasaran ... :-)

4 Ekspresi Aljabar

Buatlah program EKSPRESI.PAS sebagai latihan Anda menjelang ON. Latihan ini mulai agak sulit. Tujuan latihan ini untuk Anda membiasakan diri dengan kompiler Free Pascal yang digunakan di web server saat menguji pekerjaan Anda yang mungkin berbeda dengan kompiler yang sering Anda gunakan selama ini. Selain itu anda mulai berlatih pemrograman dengan tingkat kesulitan mulai mendekati soal-soal di ON nanti.

Program anda harus dapat membaca string masukan yang berisi ekspresi aritmetika yang terdiri atas operator pangkat-kali-bagi-tambah-kurang dan menuliskan urutan pengerjaannya yang benar. Misalnya :

$$a - b + c/d * e/f \wedge g - h * j$$

Untuk menentukan urutan pengerjaannya dalam penulisannya operator-operator tersebut diberikan tingkat prioritas; pangkat paling tinggi, kemudian kali dan bagi pada prioritas yang sama, dan terakhir tambah dan kurang, pada prioritas yang sama. (Note : Dalam latihan ini tanda kurung atau operator lain belum diikutsertakan). Dengan adanya tingkat prioritas ini maka $f \wedge g$ harus dikerjakan sebelum e/f atau $g - h$. Jika prioritas sama sehingga mana yang di sebelah kiri akan dikerjakan lebih dahulu dari yang di sebelah kanan. Untuk contoh di atas c/d dikerjakan terlebih dahulu dari pada $d * e$. Dengan menggunakan nama variabel sementara xi untuk menerima hasil pengerjaan suatu operasi, maka salah satu urutan pengerjaan ekspresi tersebut adalah :

```
x1 = a - b  
x2 = c/d  
x3 = x2 * e  
x4 = f \wedge g  
x5 = x3/x4  
x6 = x1 + x5  
x7 = h * j  
x8 = x6 - x7
```

4.1 Masukan

Program itu harus membaca masukan dari file bernama EKSPRESI.IN. File ini akan berisikan satu baris teks ekspresi aritmetika dengan panjang < 256 karakter. Operator pangkat ditulis dengan simbol ' \wedge ', operator kali dengan simbol '*', operator bagi dengan simbol '/', operator tambah dengan simbol '+', dan operator kurang dengan simbol '-'. Operand-operand-nya sendiri adalah

menggunakan karakter huruf tunggal (a-Z, A-Z) untuk memudahkan anda membaca masukan. Dalam ekspresi tidak ada karakter spasi atau karakter lainnya selain huruf atau karakter simbol operator tersebut di atas.

4.2 Keluaran

Program harus menuliskan keluaran dalam file bernama EKSPRESI.OUT. Keluaran berisikan baris-baris operasi untuk mengerjakan ekspresi masukan yang dibantu oleh variabel-variabel sementara x_i . Agar keluaran menjadi unik maka urutan sedapat mungkin dari kiri ke kanan ekspresi kecuali kalau terkait dengan prioritas. Misalnya $a - b$ harus ditulis lebih dahulu dari c/d karena $a - b$ tidak bergantung hasil c/d . Variabel-variabel sementara x_i dituliskan sebagai karakter x dan bilangan i dengan i membesar dari baris pertama ke baris terakhir.

Contoh 1

EKSPRESI.IN

$$a - b + c/d$$

File.OUT

$$x1 = a - b$$

$$x2 = c/d$$

$$x3 = x1 + x2$$

Contoh 2

EKSPRESI.IN

$$c/d * e/f \wedge g$$

EKSPRESI.OUT

$$x1 = c/d$$

$$x2 = x1 * e$$

$$x3 = f \wedge g$$

$$x4 = x2/x3$$

Contoh 3

EKSPRESI.IN

$$a - b + c/d * e/f \wedge g - h * j$$

EKSPRESI.OUT

$$x1 = a - b$$

$$x2 = c/d$$

$$x3 = x2 * e$$

$$\begin{aligned}
 x4 &= f \wedge g \\
 x5 &= x3/x4 \\
 x6 &= x1 + x2 \\
 x7 &= h * j \\
 x8 &= x6 - x7
 \end{aligned}$$

4.3 Contoh Penyelesaian

Berikut ini adalah contoh program untuk menyelesaikan masalah di atas.

```

program analisa_ekspresi_aljabar;
{ versi sabtu }
uses crt;
var ekspresi : string;
    batas_kiri,batas_kanan,proses,sekarang,i,j,jumlah_tanda,panjang : byte;
    tanda,karakter : array[1..255] of string[1];
    str_temp,suku_kiri,suku_kanan: string[2];
    prioritas : array[1..255] of byte;
    cari_prioritas, selesai : boolean;
    substitusi : string[6];
    jumlah_tanda_asli,nilai: byte;
    kode_error : integer;

begin
    clrscr;
    ekspresi := 'c/d*e/f^g';
    ekspresi := 'a-b+c/d*e/f^g-h*j';
    panjang := length(ekspresi);
    jumlah_tanda := 0;
    proses := 0;

{
    proses mencari jumlah operator dan operator apa saja
    yang ada beserta tingkatnya
}

    for i:=1 to panjang do
        begin
            karakter[i] := copy(ekspresi,i,1);
            if (karakter[i]='-') or (karakter[i]='+') then
                begin
                    inc(jumlah_tanda);
                    prioritas[jumlah_tanda] := 1;
                    tanda[jumlah_tanda] := karakter[i];
                end
            else
                if (karakter[i]='/') or (karakter[i]='*') then
                    begin
                        inc(jumlah_tanda);
                        prioritas[jumlah_tanda] := 2;
                    end
                end
            end
        end
    end

```

```

        tanda[jumlah_tanda] := karakter[i];
    end
else
    if (karakter[i]='^') then
        begin
            inc(jumlah_tanda);
            prioritas[jumlah_tanda] := 3;
            tanda[jumlah_tanda] := karakter[i];
        end;
    end;
end;
jumlah_tanda_asli := jumlah_tanda;

{ mencari operator mana yang akan dikerjakan terlebih dahulu }
cari_prioritas := false;
repeat
    sekarang := 1;
    for i:=1 to jumlah_tanda do
        begin
            if prioritas[sekarang+1]>prioritas[sekarang] then
                sekarang := sekarang+1
            else
                cari_prioritas:=true;
            end;
        end;
    until cari_prioritas;
    selesai := false;
repeat
    inc(proses);
    writeln('ekspresi : ',ekspresi);
    write('tanda ke : ',sekarang,' yang mau dikerjakan ');
    textcolor(yellow+blink);writeln(tanda[sekarang]);
    textcolor(white);
    batas_kiri := 2*sekarang-1;
    batas_kanan := 2*sekarang+1;
    writeln('batas kiri : ',batas_kiri,' batas kanan : ',batas_kanan);
    writeln('karakter batas kiri : ',karakter[batas_kiri],', ',
        'karakter batas kanan : ',karakter[batas_kanan]);
    suku_kiri := karakter[batas_kiri];
    { kalau suku kiri=1 ini artinya x1,
      kalau suku kiri=2 ini artinya x2, dan seterusnya }
    val(suku_kiri,nilai,kode_error);
    if (nilai>0) then suku_kiri:='x'+suku_kiri;

    suku_kanan := karakter[batas_kanan];
    val(suku_kanan,nilai,kode_error);
    if (nilai>0) then suku_kanan:='x'+suku_kanan;

    substitusi := suku_kiri+tanda[sekarang]+suku_kanan;

    str(proses,str_temp);
    writeln('----- substitusi x',str_temp,'=',

```

```

        substitusi,' -----');
ekspresi := '';
for i:=1 to batas_kiri-1 do ekspresi:=ekspresi+karakter[i];
ekspresi := ekspresi + str_temp;
for i:=batas_kanan+1 to panjang do ekspresi:=ekspresi+karakter[i];
writeln('ekspresi baru setelah direduksi : ',ekspresi);

{ proses seperti di awal kembali }
{
    proses mencari jumlah operator dan operator apa saja
    yang ada beserta tingkatnya
    panjang := length(ekspresi);
    jumlah_tanda := 0;

    for i:=1 to panjang do
        begin
            karakter[i] := copy(ekspresi,i,1);
            if (karakter[i]='-') or (karakter[i]='+') then
                begin
                    inc(jumlah_tanda);
                    prioritas[jumlah_tanda] := 1;
                    tanda[jumlah_tanda] := karakter[i];
                end
            else
                if (karakter[i]='/') or (karakter[i]='*') then
                    begin
                        inc(jumlah_tanda);
                        prioritas[jumlah_tanda] := 2;
                        tanda[jumlah_tanda] := karakter[i];
                    end
                else
                    if (karakter[i]='^') then
                        begin
                            inc(jumlah_tanda);
                            prioritas[jumlah_tanda] := 3;
                            tanda[jumlah_tanda] := karakter[i];
                        end;
                    end;
        end;

    { mencari operator mana yang akan dikerjakan terlebih dahulu }
    cari_prioritas := false;
    writeln('jumlah tanda ',jumlah_tanda);
    repeat
        sekarang := 1;
        for i:=1 to jumlah_tanda do
            begin
                if prioritas[sekarang+1]>prioritas[sekarang] then
                    sekarang := sekarang+1
                else
                    cari_prioritas:=true;
            end;
        end;
    until cari_prioritas;
}

```



```

        end;
        if jumlah_tanda=1 then cari_prioritas:=true;
until cari_prioritas;
selesai := false;
if jumlah_tanda=1 then
begin
    str(jumlah_tanda_asli,str_temp);
    writeln('----- x',str_temp,'=x',karakter[1],karakter[2],'x',karakter[3])
    if jumlah_tanda=1 then selesai:=true;
end;
readln;
until selesai;
end.

```

4.4 Contoh hasil *running*

```

ekspresi : a-b+c/d*e/f^g-h*j
tanda ke : 1 yang mau dikerjakan -
batas kiri : 1 batas kanan : 3
karakter batas kiri : a karakter batas kanan : b
----- substitusi x1=a-b -----
ekspresi baru setelah direduksi : 1+c/d*e/f^g-h*j
jumlah tanda 7

```

```

ekspresi : 1+c/d*e/f^g-h*j
tanda ke : 2 yang mau dikerjakan /
batas kiri : 3 batas kanan : 5
karakter batas kiri : c karakter batas kanan : d
----- substitusi x2=c/d -----
ekspresi baru setelah direduksi : 1+2*e/f^g-h*j
jumlah tanda 6

```

```

ekspresi : 1+2*e/f^g-h*j
tanda ke : 2 yang mau dikerjakan *
batas kiri : 3 batas kanan : 5
karakter batas kiri : 2 karakter batas kanan : e
----- substitusi x3=x2*e -----
ekspresi baru setelah direduksi : 1+3/f^g-h*j
jumlah tanda 5

```

```

ekspresi : 1+3/f^g-h*j
tanda ke : 3 yang mau dikerjakan ^
batas kiri : 5 batas kanan : 7
karakter batas kiri : f karakter batas kanan : g
----- substitusi x4=f^g -----
ekspresi baru setelah direduksi : 1+3/4-h*j
jumlah tanda 4

```

```

ekspresi : 1+3/4-h*j
tanda ke : 2 yang mau dikerjakan /

```

```

batas kiri : 3 batas kanan : 5
karakter batas kiri : 3 karakter batas kanan : 4
----- substitusi x5=x3/x4 -----
ekspresi baru setelah direduksi : 1+5-h*j
jumlah tanda 3

ekspresi : 1+5-h*j
tanda ke : 1 yang mau dikerjakan +
batas kiri : 1 batas kanan : 3
karakter batas kiri : 1 karakter batas kanan : 5
----- substitusi x6=x1+x5 -----
ekspresi baru setelah direduksi : 6-h*j

jumlah tanda 2
ekspresi : 6-h*j
tanda ke : 2 yang mau dikerjakan *
batas kiri : 3 batas kanan : 5
karakter batas kiri : h karakter batas kanan : j
----- substitusi x7=h*j -----
ekspresi baru setelah direduksi : 6-7
jumlah tanda 1
----- x8=x6-x7

```

5 MULTIPALINDROM

Palindrom adalah kata yang dapat dibaca sama saja baik dari kiri ke kanan ataupun dari kanan ke kiri. Suatu palindrom sedikitnya berisi satu huruf. Misalnya, "malam", "a" dan "ada" masing-masing adalah palindrom. Sebaliknya, setiap kata bukan merupakan palindrom dapat dianggap sebagai deretan sejumlah palindrom. Dengan kata lain, kata tersebut dapat dipecah-pecahkan ke dalam sejumlah palindrom. Jadi, setiap kata pada dasarnya dapat dipandang sebagai multipalindrom yang tersusun atas n palindrom, dengan $n > 0$. Untuk setiap kata terdapat sejumlah kemungkinan harga n . Dengan definisi itu maka setiap palindrom adalah multipalindrom dengan jumlah minimal $n=1$. Misalnya, kata "minimisasi" terdiri atas sedikitnya 2 palindrom yaitu "minim"- "isasi" (Red : ralat dan versi sebelumnya).

Buatlah suatu program dengan nama MULTIPAL.AS yang akan menghitung jumlah palindrom minimal dari suatu kata yang diberikan.

5.1 Format Masukan

File masukan adalah MULTIPAL.IN yang hanya berisi kata untuk dipecah-pecah ke dalam sejumlah palindrom. Karakter-karakter untuk membentuk kata adalah huruf kecil (a-z). Panjang dari kata tidak akan lebih dari 100 huruf.

5.2 Format Keluaran

Keluaran dituliskan dalam file MULTIPAL.OUT yang menyebutkan jumlah terkecil palindrom yang dapat dibuat.

Contoh-contoh :

MULTIPAL.IN	MULTIPAL.IN	MULTIPAL.IN
anaban	abaccbcb	anavolimilana
MULTIPAL.OUT	MULTIPAL.OUT	MULTIPAL.OUT
2	3	5

PENJELASAN CONTOH :

#1 a naban
#2 aba cc bcb
#3 ana v o limil ana

5.3 Contoh Penyelesaian

```
program mencari_multipaliandrom_pada_suatu_tulisan;
{ dikembangkan oleh dwi sakethi
      dwijim@unila.ac.id
      dwijim@gmail.com
      0816 403 432
      pada tanggal 18 agustus 2006 }

uses crt;

var file_data,file_hasil : text;
    tulisan_selesai,tulisan_asli,potongan_tulisan,tulisan : string;
    selesai : boolean;
    jumlah_paliandrom,panjang_tulisan_tetap,paliandrom : byte;
    mulai_tulisan_baru,jumlah_looping,batas_kanan : byte;

{
    fungsi ini memberikan nilai 1 jika kata yang dicek berupa
    paliandrom, jika kata yg dicek bukan merupakan paliandrom
    maka fungsi ini memberikan nilai 0
}
function cek_paliandrom(tulisan_yg_dicek : string):byte;
var hasil          : byte;
    ii,panjang_tulisan_x : byte;
    hasil_reverse    : string;
begin
    hasil          := 0;
    hasil_reverse  := '';
    panjang_tulisan_x := length(tulisan_yg_dicek);
    for ii:=panjang_tulisan_x downto 1 do
```

```

        begin
            hasil_reverse := hasil_reverse +
                                copy (tulisan_yg_dicek,ii,1);
        end;
    if tulisan_yg_dicek=hasil_reverse then
        hasil:=1;
        cek_paliandrom := hasil;
    end;

{ --- program utama --- }
begin

    clrscr;
    { membuka file data }
    assign(file_data,'multipal.in');
    reset(file_data);

    { membaca data tulisan }
    read(file_data,tulisan);
    writeln('tulisan asal : ',tulisan);

    { membuat file hasil }
    assign(file_hasil,'multipal.out');
    rewrite(file_hasil);

    { proses pencarian paliandrom dilakukan sampai batas
      akhir tulisan }
    selesai := false;
    batas_kanan := length(tulisan);
    panjang_tulisan_tetap := length(tulisan);
    potongan_tulisan := tulisan;
    tulisan_asli := tulisan;
    tulisan_selesai := '';
    jumlah_paliandrom := 0;
    repeat
        paliandrom :=cek_paliandrom(potongan_tulisan);
        writeln('asal : ',potongan_tulisan, ' cek : ',paliandrom);
        if paliandrom=0 then
            begin
                batas_kanan := batas_kanan - 1;
                potongan_tulisan := copy(potongan_tulisan,1,batas_kanan);
            end
        else
            begin
                writeln('paliandrom : ',potongan_tulisan);
                tulisan_selesai := tulisan_selesai + potongan_tulisan;
                mulai_tulisan_baru := length(potongan_tulisan)+1;
                potongan_tulisan := copy(tulisan,mulai_tulisan_baru,panjang_tulisan_tetap-mulai
                panjang_tulisan_tetap := length(potongan_tulisan);
                batas_kanan := length(potongan_tulisan);
            end
        end
    until selesai;
    writeln('paliandrom : ',tulisan_selesai);
end;

```

```

        tulisan := potongan_tulisan;
        if tulisan<>' ' then
            jumlah_paliandrom := jumlah_paliandrom + 1;
        readln;
    end;
    writeln('tulisan kontrol : ',tulisan_selesai);
    if tulisan_selesai=tulisan_asli then
        selesai := true;
    until selesai;

    writeln('jumlah paliandrom : ',jumlah_paliandrom);
    readln;

    { tulisan hasil ke file output }
    writeln(file_hasil,jumlah_paliandrom);
    { menutup kembali file yg telah diakses }
    close(file_hasil);
    close(file_data);
end.

```

5.4 Hasil Program

```

cek : 0
tulisan kontrol :
asal : anavolimilana cek : 0
tulisan kontrol :
asal : anavolimilan cek : 0
tulisan kontrol :
asal : anavolimila cek : 0
tulisan kontrol :
asal : anavolimil cek : 0
tulisan kontrol :
asal : anavolimi cek : 0
tulisan kontrol :
asal : anavolim cek : 0
tulisan kontrol :
asal : anavoli cek : 0
tulisan kontrol :
asal : anavol cek : 0
tulisan kontrol :
asal : anavo cek : 0
tulisan kontrol :
asal : anav cek : 0
tulisan kontrol :
asal : ana cek : 1
paliandrom : ana

tulisan kontrol : ana
asal : volimilana
cek : 0

```

tulisan kontrol : ana
asal : volimilana cek : 0
tulisan kontrol : ana
asal : volimilan cek : 0
tulisan kontrol : ana
asal : volimila cek : 0
tulisan kontrol : ana
asal : volimil cek : 0
tulisan kontrol : ana
asal : volimi cek : 0
tulisan kontrol : ana
asal : volim cek : 0
tulisan kontrol : ana
asal : voli cek : 0
tulisan kontrol : ana
asal : vol cek : 0
tulisan kontrol : ana
asal : vo cek : 0
tulisan kontrol : ana
asal : v cek : 1
paliandrom : v

tulisan kontrol : anav
asal : olimilana

cek : 0
tulisan kontrol : anav
asal : olimilana cek : 0
tulisan kontrol : anav
asal : olimilan cek : 0
tulisan kontrol : anav
asal : olimila cek : 0
tulisan kontrol : anav
asal : olimil cek : 0
tulisan kontrol : anav
asal : olimi cek : 0
tulisan kontrol : anav
asal : olim cek : 0
tulisan kontrol : anav
asal : oli cek : 0
tulisan kontrol : anav
asal : ol cek : 0
tulisan kontrol : anav
asal : o cek : 1
paliandrom : o

tulisan kontrol : anavo
asal : limilana

cek : 0

```

tulisan kontrol : anavo
asal : limilana cek : 0
tulisan kontrol : anavo
asal : limilan cek : 0
tulisan kontrol : anavo
asal : limila cek : 0
tulisan kontrol : anavo
asal : limil cek : 1
paliandrom : limil

tulisan kontrol : anavolimil
asal : ana

cek : 0
tulisan kontrol : anavolimil
asal : ana cek : 1
paliandrom : ana

tulisan kontrol : anavolimilana
asal :

cek : 1
paliandrom :

tulisan kontrol : anavolimilana

jumlah paliandrom : 5

```

6 Menghitung Jumlah Huruf

Masalah mencari jumlah huruf pada suatu kata atau kalimat. Soal yang lebih jelas, mudah-mudahan kapan-kapan akan ditulis di sini.

6.1 Contoh Penyelesain

Contoh penyelesaiannya seperti berikut :

```

{ -----
  program untuk menghitung jumlah huruf dan jenisnya
  dibuat dengan bahasa pascal
  diproses dengan sistem operasiGNU Linux Ubuntu
  compiler free pascal
  dwi sakethi http://dwijim.staff.unila.ac.id

  nama file : hitung-jumlah-huruf.pas
  ----- }

uses crt;

```

```

{ karena ada perintah cetak ke layar }

var kalimat : string;
    huruf_ke, huruf_ke_isi : byte;
    jumlah_huruf_ke : array[1..100] of byte;
    isi_huruf_ke      : array [1..100] of string;

{ prosedur mencetak identitas pembuat program }
procedure identitas_pembuat;
begin
    textcolor(yellow+blink);
    gotoxy(1,24);write('dwi sakethi http://dwijim.staff.unila.ac.id');
    textcolor(white);
end;

{ menentukan ke mana ular akan bergerak }
procedure masukan_kalimat;
begin
    gotoxy(1,1);write('masukan kalimatnya :');
    readln(kalimat);
end;

{ menentukan ke mana ular akan bergerak }
procedure hitung_huruf;
var panjang_kalimat : byte;
    huruf_sekarang : string;
    jumlah_huruf_yang_ada : byte;
    ada_huruf : boolean;
begin
    jumlah_huruf_yang_ada := 1;
    panjang_kalimat := length(kalimat);
    writeln('panjang kalimat : ',panjang_kalimat);
    for huruf_ke:=1 to panjang_kalimat do
        begin
            huruf_sekarang :=copy(kalimat,huruf_ke,1);
            { dari kata atau kalimat diambil per huruf selain spasi}

            if huruf_sekarang <> ' ' then
                begin
                    ada_huruf := false;
                    for huruf_ke_isi:=1 to jumlah_huruf_yang_ada do
                        begin
                            if huruf_ke = 1 then
                                begin
                                    { buat array yang berisi huruf-huruf yang ditemukan
                                      sampai dengan proses ini,
                                      untuk huruf pertama, pasti jadi elemen pertama }
                                    isi_huruf_ke[1] := huruf_sekarang;
                                    jumlah_huruf_ke[1] := 1;
                                    ada_huruf := true;

```



```

        end
    else
        begin
            if isi_huruf_ke[huruf_ke_isi] = huruf_sekarang then
                begin
                    inc(jumlah_huruf_ke[huruf_ke_isi]);
                    ada_huruf := true;
                end;
            end
        end; { akhir looping for huruf_ke_isi }

        { jika huruf yang sedang diproses tidak ada di antara salah
          satu dari array huruf yang sudah ada maka
          ini berarti huruf baru dan jumlahnya pasti 1 }
        if (ada_huruf = false) then
            begin
                inc(jumlah_huruf_yang_ada);
                isi_huruf_ke[jumlah_huruf_yang_ada] := huruf_sekarang;
                jumlah_huruf_ke[jumlah_huruf_yang_ada] := 1;
            end; { akhir if ada_huruf = false }
        end; { akhir if huruf_sekarang }
    end; { akhir looping kalimat }

    for huruf_ke:=1 to jumlah_huruf_yang_ada do
        begin
            writeln(isi_huruf_ke[huruf_ke], ' : ', jumlah_huruf_ke[huruf_ke]);
        end;
    end;

    { ----- program utama ----- }
begin
    clrscr;
    identitas_pembuat;
    masukan_kalimat;
    hitung_huruf;
end.

```

6.2 Contoh Keluaran

Contoh hasilnya seperti berikut :

```

dwijim@dwijim-desktop:~/Documents/olimpiade$ ppc386 hitung-jumlah-huruf.pas
Free Pascal Compiler version 2.2.2-8 [2009/01/08] for i386
Copyright (c) 1993-2008 by Florian Klaempfl
Target OS: Linux for i386
Compiling hitung-jumlah-huruf.pas
Linking hitung-jumlah-huruf
99 lines compiled, 0.1 sec
dwijim@dwijim-desktop:~/Documents/olimpiade$
masukan kalimatnya :buku tamu

```

```
panjang kalimat : 9
b : 1
u : 3
k : 1
t : 1
a : 1
m : 1
dwijim@dwijim-desktop:~/Documents/olimpiade$
```

7 Variasi Masalah Bilangan Prima

Soal ini hanya merupakan variasi dari masalah mencari bilangan prima. Jadi jika Anda sudah bisa mencari bilangan prima maka ini bisa menjadi latihan berikutnya.

7.1 Contoh Penyelesain

Contoh penyelesaiannya seperti berikut :

```
{ -----
program untuk mencetak bilangan prima sebanyak n buah
dan merupakan bilangan prima ke-i dari deretan
bilangan prima
dibuat dengan bahasa pascal
diproses dengan sistem operasiGNU Linux Ubuntu
compiler free pascal
dwi sakethi http://dwijim.staff.unila.ac.id

nama file : prima-k.pas
----- }

uses crt;
{ karena ada perintah cetak ke layar }

var banyaknya_bilangan : byte;
    bilangan_prima_ke   : array[1..100] of byte;
    bilangan_prima      : longint;
    ketemu_prima        : longint;

{ prosedur mencetak identitas pembuat program }
procedure identitas_pembuat;
begin
    textcolor(yellow+blink);
    gotoxy(1,24);write('dwi sakethi http://dwijim.staff.unila.ac.id');
    textcolor(white);
end;

{ memasukkan data banyaknya bilangan prima yang dicari
dan urutan masing-masing bilangan prima }
procedure masukan_data;
```

```

var bilangan_ke : byte;
begin
  gotoxy(1,1);write('Berapa banyak bilangan prima :');
  readln(banyaknya_bilangan);
  for bilangan_ke:=1 to banyaknya_bilangan do
  begin
    write('Bilangan prima ke : ');
    readln(bilangan_prima_ke[bilangan_ke]);
  end;
end;

{ mengecek apakah suatu bilangan termasuk bilangan prima atau bukan}
function cek_prima_apa_bukan(bilangan_ini:longint):boolean;
var bilangan_sekarang : longint;
begin
  cek_prima_apa_bukan:=TRUE;
  for bilangan_sekarang:=2 to bilangan_ini-1 do
  begin
    if (bilangan_ini mod bilangan_sekarang)=0 then
    begin
      cek_prima_apa_bukan:=FALSE;
      exit;
    end;
  end;
end;

{ ----- program utama ----- }
{ mencetak bilangan prima }
procedure mencetak_hasil;
var bilangan_ke : byte;
    urutan      : byte;
begin
  for bilangan_ke:=1 to banyaknya_bilangan do
  begin
    write('Bilangan prima ke : ',bilangan_prima_ke[bilangan_ke],' adalah : ');
    urutan := 0;
    bilangan_prima := 2;
    repeat
      if cek_prima_apa_bukan(bilangan_prima)=TRUE then
      begin
        inc(urutan);
        ketemu_prima := bilangan_prima;
        { jika suatu bilangan termasuk bilangan prima maka
          urutan bertambah yang tadinya 0 menjadi 1 dst
          kemudian ditandai juga bahwa bilangan itu adalah
          bilangan prima }
      end;
      inc(bilangan_prima);
    until urutan=bilangan_prima_ke[bilangan_ke];
    writeln(ketemu_prima);
  end;
end;

```

```

    end;
end;

begin
    clrscr;
    identitas_pembuat;
    masukan_data;
    mencetak_hasil;
    writeln;
end.

```

7.2 Contoh Keluaran

Contoh hasilnya seperti berikut :

```

dwijim@dwijim-desktop:~/Documents/olimpiade$ ppc386 prima-k.pas
Free Pascal Compiler version 2.2.2-8 [2009/01/08] for i386
Copyright (c) 1993-2008 by Florian Klaempfl
Target OS: Linux for i386
Compiling prima-k.pas
Linking prima-k
92 lines compiled, 0.1 sec
dwijim@dwijim-desktop:~/Documents/olimpiade$
Berapa banyak bilangan prima :3
Bilangan prima ke : 4
Bilangan prima ke : 1
Bilangan prima ke : 8
Bilangan prima ke : 4 adalah : 7
Bilangan prima ke : 1 adalah : 2
Bilangan prima ke : 8 adalah : 19

dwijim@dwijim-desktop:~/Documents/olimpiade$

```