MODUL PEMBINAAN

OLIMPIADE SAINS BIDANG KOMPUTER

ALGORITMA DAN PEMROGRAMAN

Oleh

TIM PEMBINA OLIMPIADE KOMPUTER
ILMU KOMPUTER UNIVERSITAS UDAYANA
(disajikan untuk peserta pembinaan bidang komputer – OSN 2009)

PEMERINTAH DAERAH PROPINSI BALI
DINAS PENDIDIKAN PEMUDA DAN OLAHRAGA
2009

DAFTAR ISI

DASAR – DASAR PEMROGRAMAN

1. PENDAHULUAN

Dalam dunia pemrograman terdapat dua hal penting yang perlu diperhatikan yaitu belajar pemrograman dan belajar pengkodean. Dua hal ini pada dasarnya berbeda dimana pengkodean hanya salah satu langkah dalam pemrograman. Untuk meningkatkan kemampuan pemrograman maka harus *belajar pemrograman* sedangkan untuk meningkatkan kemampuan pengkodean harus *belajar bahasa pemrograman*. Materi yang dipelajari berbeda dan tujuannya pun berbeda.

Belajar pemrograman meliputi banyak hal, terutama adalah belajar memecahkan masalah atau merumuskan algoritma, mulai dari penentuan garis besar pemecahan sampai langkah-langkah rinci. Algoritma adalah strategi pemecahan masalah yang meliputi metode, dan sistematika pemecahan yang dituliskan dalam notasi yang disepakati. Oleh sebab itu belajar pemrograman ditekankan pada pemahaman masalah, analisis, dan melakukan sintesis untuk merumuskan solusinya.

Belajar bahasa pemrograman adalah belajar pemakaian suatu bahasa yang meliputi aturan sintaks, instruksi yang tersedia, dan cara pengoperasian kompilator bahasa pada mesin tertentu. Algoritma harus dituliskan kembali dalam bahasa pemrograman untuk dapat dieksekusi oleh komputer. Jika algoritma yang dikembangkan sudah cukup rinci, dan jika aturan bahasa pemrograman dikuasai, maka penerjemahannya ke dalam bahasa pemrograman akan sangat mudah. Penerjemahan akan berbeda untuk setiap bahasa pemrograman.

Oleh karena banyak bahasa pemrograman yang dapat digunakan dan pemilihannya akan sangat tergantung kepada masalah yang dipecahkan, maka belajar algoritma menjadi lebih penting daripada belajar bahasa pemrograman. Dokumen ini dimaksudkan untuk membantu dalam belajar algoritma dengan bantuan bahasa pemrograman yang memang tujuan awalnya adalah sebagai bahasa untuk pengajaran pemrogaman, yaitu pascal. Penekanan dokumen ini adalah pada algoritma tudak pada penggunaan bahasa pascal, meski bahasa pascal digunakan sebagai bantuan. Dengan demikian diharapkan kemampuan memecahkan masalah dapat dipelajari, tetapi sekaligus mendapat pengetahuan tentang bahasa pascal.

2. ALGORITMA

Algoritma sebagai langkah-langkah pemecahan masalah dapat dituliskan dalam beberapa cara, yaitu :

1. Uraian deskriptif
2. Bagan Alir (Flowchart)
3. Pseudocode

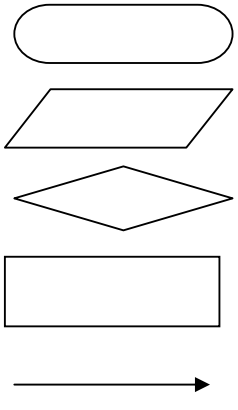
2.1 URAIAN DESKRIPTIF

Penulisan algoritma dengan uraian deskriptif adalah sebuah cara menggambarkan langkah-langkah pemecahan masalah dengan menggunakan bahasa yang biasa digunakan sehari-hari. Sebagai contoh kita akan membuat algoritma untuk merata-ratakan tiga buah bilangan, maka algoritmanya adalah :

1. Mulai
2. Baca Bilangan a,b,c
3. Jumlahkan ketiga bilangan tersebut
4. Bagi jumlah tersebut dengan 3
5. Tulis hasilnya
6. Selesai

2.2 BAGAN ALIR (FLOWCHART)

Flowchart adalah cara penulisan algoritma dengan menggunakan notasi grafik. Terdapat beberapa notasi dasar penggambaran flowchart, antara lain:



Mulai atau Selesai

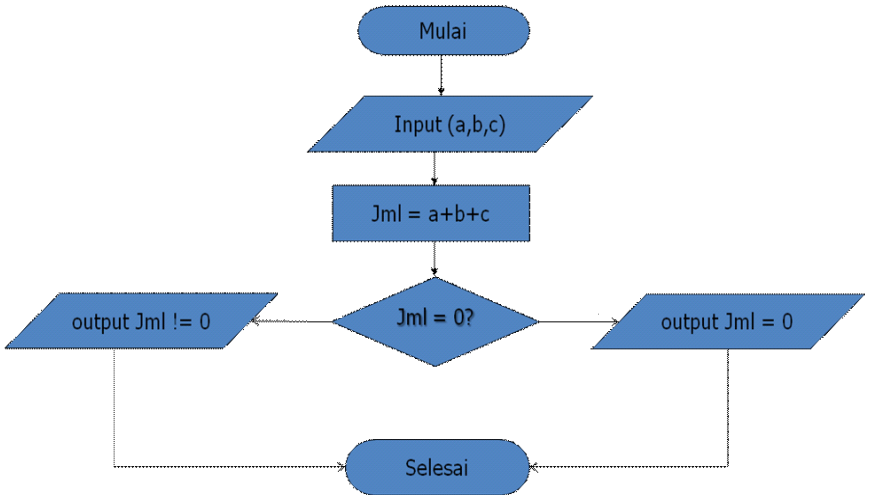
Input atau Output

Pengujian atau Percabangan

Proses

Aliran Tahapan

Contoh penggunaan flowchart ini adalah sebagai berikut :



Penulisan algoritma dengan menggunakan flowchart ini tidak banyak digunakan. Hal ini disebabkan oleh beberapa pertimbangan, antara lain :

1. Flowchart hanya cocok untuk masalah yang kecil. Untuk masalah yang besar, penggunaan flowchart menjadi tidak efisien.
2. Penggunaan flowchart memerlukan kemampuan menggambar yang baik sehingga sering kali penulisan algoritma dengan teknik ini membutuhkan waktu yang relatif lama.

Tidak

2.3 PSEUDOCODE

- **Pseudocode** adalah suatu cara penulisan algoritma agar ide dan logika dari algoritma dapat disampaikan/diekspersikan. Algoritma dalam bentuk pseudocode biasanya mengadopsi beberapa notasi pada sebuah bahasa pemrograman tertentu.
- **Pseudopascal** (alias **Peseudocode Pascal**) adalah pseudocode yang menggunakan (mengadopsi) beberapa notasi **Bahasa Pascal** berikut struktur penulisan programnya.

Prinsip dalam penulisan pseudocode adalah *"tuliskan seringkas-ringkasnya sejauh tidak mengurangi pengertian dari algoritma yang menjadi fokus pembahasan tersebut."* Dalam kaitannya dengan materi seleksi, pseudocode yang digunakan adalah pseudopascal berdasarkan kenyataan bahwa notasi-notasi Bahasa Pascal jauh lebih mudah dipahami daripada notasi bahasa pemrograman populer lainnya saat ini

terutama bagi pemula (misalnya bahasa C). Selain itu setiap notasi yang digunakan dijaga untuk selalu berpadanan dengan notasi yang ada dalam bahasa lain tersebut sehingga peserta yang lebih menguasai bahasa selain Pascal tetap dapat memahami ide dari algoritma terkait.

3. BAHASA PEMROGRAMAN PASCAL

Bahasa Pascal adalah bahasa pemrograman tingkat tinggi (*high-level language*) yang cukup populer, khususnya di Indonesia. Hal ini disebabkan bahasa Pascal lebih mudah dipahami dibandingkan dengan bahasa lainnya, seperti bahasa C, C++, *assembly*, dan bahasa tingkat tinggi lainnya. Selain itu, bahasa Pascal adalah bahasa pemrograman yang terstruktur dan seperti halnya dengan bahasa pemrograman tingkat tinggi lainnya, lebih mendekati bahasa manusia (dalam hal ini adalah bahasa Inggris sebagai bahasa Internasional) sehingga sangat cocok diterapkan dalam dunia pendidikan. Sebagai salah satu bahasa pemrograman komputer yang tergolong ke dalam kelompok bahasa pemrograman terstruktur (*structured programming*), struktur program Pascal relatif sederhana. Bangun suatu program Pascal akan mengikuti pola berikut:

```
1.  Judul Program
2.  Blok Program, terdiri dari:
    2.1     Bagian Deklarasi, meliputi (secara runut):
            Deklarasi Label
            Deklarasi Unit
            Deklarasi Konstanta
            Deklarasi Tipe
            Deklarasi Variabel
            Deklarasi Procedure dan atau Function
    2.2     Bagian Pernyataan, yang terdiri dari instruksi-
    instruksi pemrograman,
```

Dalam kasus-kasus pemrograman, bangun program Pascal di atas akan bisa berbentuk seperti berikut:

```
program Judul_Program_Pascal;                                    {Judul program}
uses  Crt;                                                     {Deklarasi unit CRT}
label Label_Saya;                                                {Deklarasi label}
const                                                          {Deklarasi konstanta}
      Bahasa = 'Pascal';
       Versi  = 7.0;
type  Nama = string [30];                                        {Deklarasi tipe}
var   Umur_Kamu : integer;                                     {Deklarasi variabel}
       Nama_Kamu : Nama;

procedure Baca_dan_Tulis;                                      {Deklarasi procedure}
begin
```

```
  write  ('Nama Kamu       : '); readln(Nama_Kamu);
  write  ('Usia Kamu       : '); readln(Usia_Kamu);
  writeln('Hallo, ', Nama_Kamu);
  writeln('Umur kamu sekarang ', Usia_Kamu, ' tahun');
end;

begin                                          {Awal instruksi-instruksi program induk}
  clrscr;
  writeln('Hallo, Saya Mr. Pascal');
  Baca_dan_Tulis;
  writeln('Kita sedang belajar Bahasa ', Bahasa, ' Versi ', Versi);
end.                                                    {Akhir instruksi program induk}
```

4. PENGULANGAN PADA PASCAL

Konsep pengulangan satu atau lebih pernyataan dalam suatu program merupakan salah satu konsep terpenting dalam pemrograman komputer. Pengulangan sekelompok pernyataan program komputer sering disebut dengan nama Looping. Adanya kemampuan dari bahasa-bahasa pemrograman komputer untuk mengerjakan satu atau lebih pernyataan-pernyataan program sebanyak jumlah yang ditentukan menyebabkan proses komputasi menjadi lebih efisien. Sebagai misal, untuk menghitung jumlah dari 100 bilangan asli yang pertama – yang ekspresinya dapat dituliskan dalam bentuk Jumlah = 1 + 2 + 3 + … + 100, tidaklah berarti pemrogram harus mengalokasikan 100 variabel memori untuk variabel di sebelah kanan tanda penugasan dan satu variabel untuk Jumlah. Pengulangan memungkinkan proses komputasi tersebut dilakukan secara efisien.

Bahasa Pascal mengenal 3 jenis pengulangan yaitu pengulangan `for`, pengulangan **`while`** dan pengulangan ***`repeat … until`***. Ketiga jenis pengulangan ini membutuhkan keberadaan ekspresi kondisional yang digunakan untuk mengetahui apakah sebuah blok pernyataan masih akan dieksekusi ataukah pengulangan berakhir. Selama ekspresi kondisional ini memberikan nilai True, blok pernyataan terus diulangi hingga ekspresi menghasilkan nilai False. Hal kedua yang perlu diperhatikan, ketiga jenis pengulangan ini – jika diperhatikan secara seksama – memiliki konsep yang berbeda dalam melakukan pengulangan blok program, seperti yang ditulis pada penjelasan berikut.

4.1 PENGULANGAN FOR

Pengulangan `for` pada Pascal akan memiliki salah satu dari 2 sintaks berikut:

```
for Variabel := First to Last do
    begin
      statement_1;
      statement_2;
      …
      statement_N;
    end;
```

atau

```
for Variabel := First downto Last do
    begin
      statement_1;
      statement_2;
      …
      statement_N;
    end;
```

Pada kedua sintaks di atas, tipe dari `Variabel`, `First` dan `Last` harus ordinal (integer atau char). Klausa **`to`** akan menyebabkan nilai dari `Variabel` bertambah 1 dan klausa **`downto`** menyebabkan nilai `Variabel` berkurang 1. Untuk mempermudah pemahaman Anda, misalkanlah ingin diketahui jumlah dari 10 bilangan asli yang pertama, yaitu:

```
Jumlah = 1 + 2 + … + 10
```

Seperti biasa, cobalah untuk membuat algoritma atau diagram alir dari permasalahan tersebut.

Berikut adalah *listing code* dengan Pascal:

```
program Hitung_Jumlah_Asli;
uses Crt;
var Jumlah, Pencacah: byte;

begin
  Clrscr;
  writeln('Menghitung Jumlah 10 Bilangan Asli yang Pertama');
  writeln('=============================================');
  Jumlah := 0;                                          {Inisialisasi Variabel}
  for Pencacah := 1 to 10 do                              {Memulai Pengulangan}
      Jumlah := Jumlah + Pencacah;
  writeln('1 + 2 + … + 10 = ', Jumlah);
  readkey;
end.
```

Memperhatikan bahwa hanya ada 1 instruksi yang diulangi setelah `do`, maka pasangan kata `begin … end` bisa dihilangkan. Namun jika terdapat lebih dari 1 instruksi, Anda harus meletakkan instruksi-instruksi tersebut di antara pasangan `begin … end`. Perhatikan kembali program di atas dan sintaks pada halaman sebelumnya!

Sintaks kedua akan digunakan untuk mengetahui nilai faktorial suatu bilangan. Seperti yang Anda ketahui, faktorial dari X (X > 1) didefinisikan sebagai X! = X (X-1) (X-2) … (1). Berikut adalah program untuk menghitung faktorial suatu bilangan.

```
program Faktorial_1;
uses Crt;
var Bilangan, Pencacah: byte;
    Faktorial         : longint;

begin
  Clrscr;
  writeln('Menghitung Nilai Faktorial Sebuah Bilangan');
  writeln('=========================================');
  write  ('Masukkan Bilangan yang Dihitung: '); readln(Bilangan);

  Faktorial := 1;                                          {Inisialisasi Variabel}
  for Pencacah := Bilangan downto 1 do                       {Memulai Pengulangan}
      Faktorial := Faktorial * Pencacah;
  writeln('Faktorial dari ', Bilangan, ' = ', Faktorial);
  readkey;
end.
```

Latihan Mandiri:

Dengan menggunakan pengulangan `for`, rancanglah program untuk menghitung jumlah dari deret berikut:

```
1, 9, 25, 49, 81, …
```

di mana banyak suku dari deret ditentukan pengguna program!

4.2 PENGULANGAN WHILE

Pengulangan `while` pada Pascal akan memiliki sintaks berikut:

```
while Logical_Expression do
    begin
      statement_1;
      statement_2;
      …
      statement_N;
    end;
```

Pada sintaks di atas, selama `Logical_Expression` bernilai `True`, maka kelompok instruksi pada blok `begin … end` akan tetap dieksekusi. Perhatikan contoh berikut:

```
program Faktorial_2;
uses Crt;
var Bilangan  : byte;
    Faktorial : longint;

begin
  Clrscr;
  writeln('Menghitung Nilai Faktorial Sebuah Bilangan');
  writeln('=========================================');
  write  ('Masukkan Bilangan yang Dihitung: '); readln(Bilangan);

  Faktorial := 1;                                          {Inisialisasi Variabel}
```

```
  while Bilangan > 1 do                                          {Memulai Pengulangan}
    begin
      Faktorial := Faktorial * Bilangan;
      dec(Bilangan);                                        {Mengurangi Nilai Bilangan}
    end;
  writeln('Faktorial dari ', Bilangan, ' = ', Faktorial);
  readkey;
end.
```

4.3 PENGULANGAN REPEAT … UNTIL

Pengulangan `repeat … until` pada Pascal akan memiliki sintaks berikut:

```
repeat
    begin
      statement_1;
      statement_2;
      …
      statement_N;
    end;
until Logical_Expression;
```

Pada sintaks di atas, kelompok instruksi yang terletak di antara pasangan `begin … end` akan dieksekusi `Logical_Expression` menghasilkan nilai `True`. Jika dicermati, maka setidak-tidaknya kelompok instruksi tersebut akan dieksekusi satu kali. Kenapa? Berikut adalah penerapan pengulangan tersebut yang digunakan untuk menghitung jumlah dari deret berikut:

Jumlah = $1^1 + 3^2 + 5^3 + 7^4 + \ldots$

```
program Jumlah_Deret;
uses Crt;
var Counter, Suku, Bilangan, Pangkat  : byte;
    Jumlah : longint;

begin
  Clrscr;
  writeln('Menghitung Jumlah dari Deret');
  writeln('============================');
  write  ('Masukkan Jumlah Suku Deret: '); readln(Suku);

  Jumlah   := 1;                                              {Inisialisasi Variabel}
  Bilangan := 1;
  Pangkat  := 0;
  repeat                                                        {Memulai Pengulangan}
    begin
      for Counter = 1 to Pangkat do
          Bilangan := Bilangan * Bilangan;
      Jumlah := Jumlah + Bilangan;
      inc(Bilangan, 2);                                {Menaikkan Nilai Bilangan dengan 2}
```

```
      inc(Pangkat);                                  {Menaikkan Nilai Pangkat dengan 1}
    end;
  until Bilangan > (2*Suku - 1);                                      {Akhir Pengulangan}

  writeln('Jumlah Deret',' = ', Jumlah);
  readkey;
end.
```

Latihan Mandiri:

1. Nilai dari Cos (X) bisa dihampiri dengan menggunakan Perluasan Deret Maclaurin yang dinyatakan dalam bentuk berikut:

   $\text{Cos}(X) = 1 - (x^2/2!) + (x^4/4!) - (x^6/6!) + (x^8/8!) - \ldots$

   Buatlah diagram alir untuk menyelesaikan persoalan di atas dengan masukan dari pengguna adalah nilai X yang dinyatakan dalam radian (2π rad = 360^0) dan jumlah suku yang digunakan untuk menghampirinya. Mengacu kepada diagram alir yang Anda buat, rancanglah program C++.

2. Buatlah diagram alir dan program C++ yang digunakan untuk mencetak Segitiga Phytagoras di mana 'ketinggian' segitiga ditentukan oleh pengguna program. Sebagai contoh, jika pengguna memasukkan ketinggian segitiga 5 satuan maka luaran dari program seperti berikut:

   ```
   1
   1 1
   1 2 1
   1 3 3 1
   1 4 6 4 1
   ```

5. PERCABANGAN PADA PASCAL

Selain konsep pengulangan satu atau lebih pernyataan dalam suatu program, konsep tentang penentuan aliran program (*program flow*) merupakan konsep pemrograman yang tidak kalah pentingnya. Percabangan aliran program – dalam teori-teori pemrograman – dikenal dengan nama *branching* atau *conditioning*.

Pascal 7.0 memiliki 2 bentuk pengaturan cabang program yaitu: **`if … then … else`** dan ***`case … of`***. Berikut adalah penjelasan tentang ketiga bentuk tersebut secara ringkas:

5.1 PERCABANGAN IF … THEN … ELSE

Percabangan `if … then … else` pada Pascal akan memiliki sintaks berikut:

```
if Expression then
   begin
      statement_11;
```

```
      statement_21;
      …
      statement_N1;
   end
else
   begin
      statement_12;
      statement_22;
      …
      statement_N2;
   end;
```

Pada sintaks di atas, seandainya hasil dari ekspresi logika Expression bernilai True, maka pernyataan-pernyataan pada kelompok begin … end yang pertamalah yang akan dieksekusi. Sebaliknya, begin … end yang kedua yang akan dilaksanakan. Untuk menambah pemahaman Anda, misalkanlah akan dicari akar-akar persamaan kuadrat $aX^2 + bX + c = 0$ dengan menggunakan 'Rumus ABC' sebagai berikut:

$$X_{1,2} = -b \pm \frac{\sqrt{b^2 - 4ac}}{2a}$$

Perhatikanlah algoritma untuk permasalahan di atas:

0. *Mulai*
1. *Baca Nilai A, B dan C*
2. *Hitung Diskriminan Persamaan = Diskriminan = b^2 – 4ac*
3. *Jika Diskriminan < 0, maka informasikan akar-akar persamaan IMAJINER; ke nomor 6*
4. *Jika Diskriminan = 0, maka informasikan akar-akar persamaan tunggal = X = –b; ke nomor 6*
5. *Hitung X1 = –b + (√Diskriminan)/(2a) dan X2 = –b - (√Diskriminan)/(2a)*
6. *Tanya pengguna, ada persamaan lain yang akan dihitung? Jika Ya; ke nomor 1*
7. *Selesai*

Perhatikanlah algoritma di atas! Saat ada pernyataan 'jika … maka' dalam algoritma, maka saatnya Anda menggunakan konsep percabangan dalam program. Perhatikan kode berikut dengan mengacu kepada algoritma tersebut.

```
program Cari_Akar;
uses Crt;
var A, B, C : real;
    Diskriminan, Akar_1, Akar_2 : real;
    Jawab : char;
begin
  repeat
    Clrscr;
    writeln('Mencari Akar Persamaan Kuadrat dari aX^2 + bX + c = 0');
    writeln('==================================================');
    write  ('Masukkan Koefisien Kuadrat Persamaan (A) : '); readln(A);
```

```
    write  ('Masukkan Koefisien Linier Persamaan  (B) : '); readln(B);
    write  ('Masukkan Konstanta Persamaan         (C) : '); readln(C);
    writeln;                                                {Mencetak Baris Kosong}
    Diskriminan := Sqr(B) - 4*A*C;                         {Menghitung Diskriminan}
    if Diskriminan < 0 then writeln('Akar-akar Persamaan Imajiner')
    else
       begin
        if Diskriminan = 0 then
           begin
             Akar_1 := -B;
             writeln('Akar-akar Persamaan Tunggal = X = ', Akar_1:7:2);
           end
        else
           begin
             Akar_1 := -B + Sqrt(Diskriminan)/(2*A);
             Akar_2 := -B - Sqrt(Diskriminan)/(2*A);
             writeln('Akar Persamaan I = X1 = ', Akar_1:7:2);
             writeln('Akar Persamaan I = X2 = ', Akar_2:7:2);
           end;
       end;
    writeln;                                                {Mencetak Baris Kosong}

    write  ('Ada Persamaan Lainnya [Y/T]');
    repeat
      read(Jawab);
    until Jawab in ['Y','y','T','t'];
  until Jawab in ['T','t'];
end.
```

5.2 PERCABANGAN CASE … OF

Saat 'cabang' alur program bertambah banyak, maka penggunaan `if … then … else` menjadi semakin rumit. Jika Anda menjumpai kondisi pemrograman di mana 'cabang' dari program lebih dari 3, maka disarankan untuk menggunakan `case … of` yang sintaksnya seperti berikut:

```
case Expression of
   case_1: statement_1;
   case_2: statement_2;
   …
   case_N: statement_N;
else
   statement_X;
end;
```

Misalkanlah terdapat 5 kemungkinan nilai yang diperoleh seorang murid yang mengambil mata pelajaran komputer. Kemungkinan nilai tersebut adalah A, B, C, D dan E. Jika untuk setiap huruf ada 'predikatnya' maka akan ada 5 kemungkinan predikat yang disandang murid. Perhatikan program berikut:

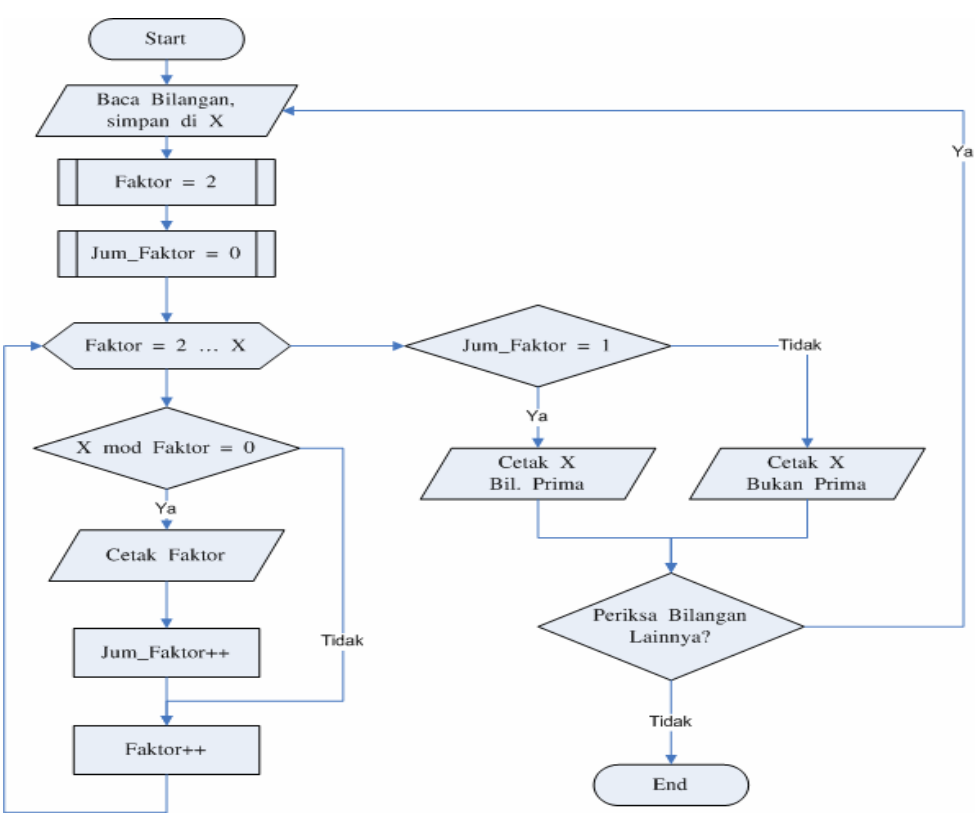
```
program Cari_Predikat;
uses Crt;
var Nilai    : char;
    Predikat : string;
    Nama     : string;
    NIS      : string[10];

begin
  writeln('Program untuk Mengetahui Predikat Kelulusan Siswa');
  writeln('-----------------------------------------------');
  write  ('Nama Siswa           : '); readln(Nama);
  write  ('Nomor Induk          : '); readln(NIS);
  write  ('Nilai yang Diperoleh : ');
  repeat
     read(Nilai);
  until UpCase(Nilai) in ['A'..'E'];

  case UpCase(Nilai) of                    {Fungsi UpCase untuk Menjadikan Huruf Kapital}
    'A' : Predikat := 'Genius!'
    'B' : Predikat := 'Sangat Berbakat!';
    'C' : Predikat := 'Cukup Berbakat!';
    'D' : Predikat := 'Kurang Berbakat!';
    'E' : Predikat := 'Tidak Berbakat!';
  end;
  writeln('Predikat Kelulusan   : ', Predikat);
  readkey;
end.
```

Latihan Mandiri:

1. Buatlah program Pascal untuk mengetahui apakah sebuah bilangan merupakan prima mengacu kepada diagram alir berikut:



2. Buatlah diagram alir dan program untuk menghitung deret berikut:

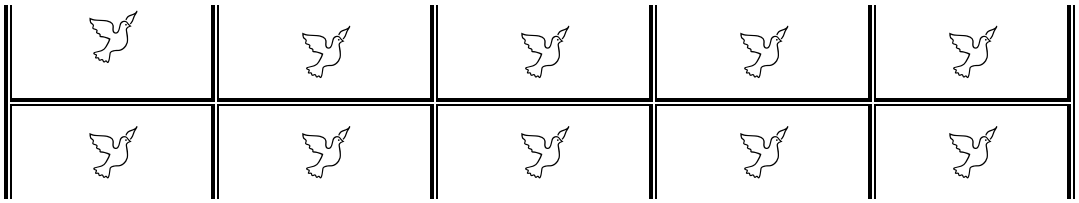
   Jumlah = 1(-2) + (-1)2 + 1(-2) + (-1)2 + 1(-2) + (-1)2 + …

   Sebagai masukan, jumlah suku dari deret!

3. Rancanglah program untuk mengetahui denda jika perpustakaan sekolah menerapkan aturan peminjaman koleksi perpustakaan sebagai berikut:
   - Koleksi yang bisa dipinjam hanyalah koleksi berupa buku-buku teks;
   - Jika koleksi yang dipinjam tergolong ke dalam kelompok langka (jumlah koleksi ≤ 3 eksemplar), maka lama peminjaman dibatasi maksimal 3 hari. Keterlambatan pengembalian untuk 4 hari pertama dikenakan denda Rp 1.000 per hari, 3 hari berikutnya dikenakan Rp. 2.500 per hari dan hari-hari berikutnya dikenakan denda Rp 5.000 per hari;
   - Jika koleksi yang dipinjam tidak tergolong ke dalam kelompok langka, maka lama peminjaman dibatasi maksimal 7 hari. Keterlambatan pengembalian dikenakan denda Rp. 500 per hari;
   - Jumlah koleksi yang bisa dipinjam untuk setiap peminjaman dibatasi maksimal 3 judul buku.

6. ARRAY

Array – sebagian buku-buku pemrograman menyebutkannya dengan nama Larik – adalah tipe data kompleks yang elemen-elemennya mempunyai tipe data yang sama. Bayangkanlah kandang merpati berikut:



Kandang di atas dapat dianalogikan sebagai suatu *array* yang berukuran 10 di mana burung yang bisa menempati setiap ‘lubang’ pada kandang adalah merpati! Perhatikan ukuran setiap ‘lubang’ – sama untuk ke-sepuluh ‘lubang’ pada kandang. Untuk membedakan sebuah ‘lubang’ dengan ‘lubang’ lainnya, dibutuhkan adanya pengenal ‘lubang’. Pengenal elemen *array* seringkali disebut dengan nama Indeks

dari *Array*, yang dinomori **dari 1** hingga jumlah dari seluruh elemen *array*. Pada kandang kita, pengenal lubang adalan 1, 2, …, 10.

Untuk mendefinisikan sebuah *array* pada Pascal, Anda bisa menggunakan salah satu dari cara berikut:

Cara I:

```
type Nilai = array [1..10] of real;
var  Nilai_Siswa : Nilai;
```

Cara II: **{Tidak Dianjurkan!!!}**

```
var  Nilai_Siswa : array [1..10] of real;
```

Meskipun kedua cara pendefinisian *array* di atas bisa dipilih, Penulis lebih menganjurkan untuk menggunakan cara pertama. Pemanfaatan *keyword* `type` untuk mendefinisikan tipe data kompleks yang diikuti *keyword* `var` untuk mendefinisikan sebuah variabel yang bertipe kompleks lebih banyak diikuti oleh para penulis buku-buku teks pemrograman Pascal.

Untuk mengakses elemen-elemen suatu *array*, maka dibutuhkan indeks dari *array*. Berbeda dengan **bahasa C atau C++** di mana indeks *array* diawali dengan angka **0** dan diakhiri dengan **n – 1** (n merupakan jumlah elemen *array*), pada **Pascal (seperti juga BASIC)** elemen *array* diawali dengan angka **1**. Sebagai misal, untuk mengakses elemen kedua dari variabel `Nilai_Siswa` pada contoh di atas bisa digunakan pernyataan `Nilai_Siswa[2]`. Perhatikan penggunaan pasangan **[ ]** untuk mengakses elemen *array*.

Misalkanlah ingin dirancang suatu program Pascal untuk mengetahui karakter di posisi ke – n dari suatu kata yang panjangnya dibatasi maksimal 15 huruf. Berikut adalah *listing code* dari program dimaksud:

```
{ Nama File Program : CekHuruf.PAS
  Tanggal File      : 14 Juli 2005
  Versi File        : 1.01 }

Program Cetak_Karakter;
Uses    Crt;
Const   MaxLen = 15;
Type    Kata = array [1 .. MaxLen] of Char;
Var     Kata_Diperiksa  : Kata;
```

```
        Kata_Diinput    : String[MaxLen];
        Counter, Posisi : Byte;

Begin
  Clrscr;
  Write('Masukkan Kata, Panjang Maksimal 15 Karakter  : ');
ReadLn(Kata_Diinput);
  For Counter := 1 to MaxLen do        {Menyalin Kata_Diinput ke Kata_Diperiksa}
      Kata_Diperiksa[Counter] := Kata_Diinput[Counter];

  Write('Tentukan Posisi Karakter yang Ingin Diketahui: '); ReadLn(Posisi);
  Write('Karakter pada Posisi ke - ', Posisi:2, ' adalah Huruf: ');
  WriteLn(Kata_Diperiksa[Posisi]);
  WriteLn;
  WriteLn('Gitu, lho!');
  ReadKey;
End.
(* End of File *)
```

Keluaran dari program di atas dapat berbentuk seperti berikut:

```
Masukkan Kata, Panjang Maksimal 15 Karakter  : Pascal Good
Tentukan Posisi Karakter yang Ingin Diketahui: 8
Karakter pada Posisi ke -  8 adalah Huruf: G
```

Program berikut digunakan untuk mencetak Daftar Nilai Pelajaran Teknologi Informasi dan Komunikasi di Sekolah Banyak Siswa, Denpasar. Perhatikan dengan baik kode program!

```
Program Cetak_Nilai_Siswa;
Uses    Crt;
Const   Max_Siswa = 40;
Type    Nama  = Array [1 .. Max_Siswa] of String [40];
        Nilai = Array [1 .. Max_Siswa] of Byte;
Var     Nama_Siswa  : Nama;
        Nilai_Siswa : Nilai;
        Counter, Nilai_Max, Nilai_Min : Byte;
        Total  : Word;
        Rataan : Real;
Begin
   WriteLn('Merekam Nama serta Nilai Siswa untuk Pelajaran TIK');
   WriteLn('-------------------------------------------------');
   for Counter := 1 to Max_Siswa Do
       Begin
          Clrscr;
          Write('Nama Siswa: '); ReadLn(Nama_Siswa [Counter]);
          Write('Nilainya  : '); ReadLn(Nilai_Siswa[Counter]);
       End;

   Nilai_Max := 0;                                    { Memroses Array Nilai_Siswa }
   Nilai_Min := 100;
   Total     := 0;
   for Counter := 1 to Max_Siswa Do
       Begin
```

```
         Total := Total + Nilai_Siswa[Counter];
         if Nilai_Max < Nilai_Siswa[Counter] then Nilai_Max := Nilai_Siswa[Counter];
         if Nilai_Min > Nilai_Siswa[Counter] then Nilai_Min := Nilai_Siswa[Counter];
       End;
   Rataan := Total/Max_Siswa;

   { Mencetak Nama dan Nilai Siswa }
   { ------------------------------------------------- }
   {       12345678901234567890123456789012345678901234567890 }
   {       0--------1---------2---------3---------4---------5 }
   Clrscr;
   WriteLn('=================================================');
   WriteLn('NAMA SISWA                              N I L A I');
   WriteLn('-------------------------------------------------');
   for Counter := 1 to Max_Siswa Do
       Begin
         Write(Nama_Siswa[Counter]);
         Write(Nilai_Siswa[Counter]:50-Length(Nama_Siswa[Counter]));
         WriteLn;
       End;
   WriteLn('-------------------------------------------------');
   WriteLn('Nilai Maksimum  : ',Nilai_Max:3);
   WriteLn('Nilai Minimum   : ',Nilai_Min:3);
   WriteLn('Nilai Rata-rata : ',Rataan:6:2);
   WriteLn('=================================================');
   ReadKey;
End.
```

Gambar berikut menunjukkan contoh luaran dari program di atas:

```
=================================================
NAMA SISWA                              N I L A I
-------------------------------------------------
Tina Toon Imut                                67
Gepeng Kerempeng                              52
Gendut Gembul Raharja                         86
Budi Badut                                    45
Sarah Sinetron                                32
-------------------------------------------------
Nilai Maksimum  :  86
Nilai Minimum   :  32
Nilai Rata-rata :  56.40
=================================================
```

Latihan Mandiri:

1. Palindrom merupakan kalimat yang jika dibaca dari belakang sama dengan dibaca dari depan. Sebagai misal, kalimat `kasur ini rusak` merupakan sebuah palindrom. Rancanglah program Pascal untuk mengetahui apakah sebuah kalimat merupakan palindrom atau bukan!

2. Buatlah sebuah program Pascal untuk melakukan penjumlahan 2 buah vektor yang berukurun sama (Vektor merupakan sebuah besaran berarah yang dimensinya adalah n x 1 dengan n menyatakan banyaknya baris pada vektor dan 1 jumlah kolom).
3. Buatlah sebuah program Pascal untuk menjumlahkan, mengurangi dan mengalikan 2 buah matriks yang berikuran 4 x 4.

7. RECORD

Di bagian sebelumnya kita telah membahas tentang *array* – sebuah tipe data yang elemen-elemennya tersusun dari tipe yang sama. Ingatlah kembali analogi tentang kandang burung merpati. Meskipun demikian, tidak selamanya dalam kasus pemrograman Anda menghadapi tipe data yang seragam. Sebagai misal, Kebun Binatang Ragunan memiliki berbagai jenis binatang dengan ukuran kandang yang berbeda-beda disesuaikan dengan besar binatangnya.

Untuk mendefinisikan *record* – seperti halnya mendefinisikan *array* – Anda harus memanfaatkan *keyword* **Type** dan **Var** secara simultan. Misalkanlah seorang siswa memiliki 3 jenis informasi, yaitu: Nama Siswa, Nomor Induk Siswa dan Nilai Pelajara TIK . Perhatikan contoh pendefinisian *record* Siswa berikut:

```
Type  Siswa = Record
        Nama_Siswa : String[40];
        NIS        : string[ 7];
        Nilai_TIK  : Byte;
      End;
Var   Siswa_Saya   : Siswa;
```

Perhatikan modifikasi program sebelumnya dengan melibatkan *record* sebagai pengganti *array*:

```
Program Cetak_Nilai_Siswa;
Uses    WinCrt;
Const   Max_Siswa = 2;
Type    Siswa = Record
          Nama      : String [40];
          NIS       : String [ 7];
          Nilai_TIK : Byte;
        End;
Var     Siswa_Saya  : Array [1 .. Max_Siswa] of Siswa;
        Counter, Nilai_Max, Nilai_Min : Byte;
```

```
        Total  : Word;
        Rataan : Real;
Begin
   WriteLn('Merekam Nama serta Nilai Siswa untuk Pelajaran TIK');
   WriteLn('-------------------------------------------------');
   for Counter := 1 to Max_Siswa Do
       Begin
          Clrscr;
          Write('Nama Siswa: '); ReadLn(Siswa_Saya[Counter].Nama);
          Write('N I S     : '); ReadLn(Siswa_Saya[Counter].NIS);
          Write('Nilai TIK : '); ReadLn(Siswa_Saya[Counter].Nilai_TIK);
       End;
   Nilai_Max := 0;     { Memroses Array Nilai_Siswa }
   Nilai_Min := 100;
   Total     := 0;
   for Counter := 1 to Max_Siswa Do
       Begin
        Total := Total + Siswa_Saya[Counter].Nilai_TIK;
        if Nilai_Max < Siswa_Saya[Counter].Nilai_TIK then Nilai_Max := Siswa_Saya[Counter].Nilai_TIK;
        if Nilai_Min > Siswa_Saya[Counter].Nilai_TIK then Nilai_Min := Siswa_Saya[Counter].Nilai_TIK;
       End;
   Rataan := Total/Max_Siswa;
   { ------------------------------------------------------ }
   {        12345678901234567890123456789012345678901234567890 }
   {        0--------1---------2---------3---------4---------5 }
   Clrscr;
   WriteLn('=================================================');
   WriteLn('NAMA SISWA                                 NILAI');
   WriteLn('-------------------------------------------------');
   for Counter := 1 to Max_Siswa Do
       Begin
          Write(Siswa_Saya[Counter].Nama);
          Write(Siswa_Saya[Counter].Nilai_TIK:50-Length(Siswa_Saya[Counter].Nama));
          WriteLn;
       End;
   WriteLn('-------------------------------------------------');
   WriteLn('Nilai Maksimum  : ',Nilai_Max:3);
   WriteLn('Nilai Minimum   : ',Nilai_Min:3);
   WriteLn('Nilai Rata-rata : ',Rataan:6:2);
   WriteLn('=================================================');
   ReadKey;
End.
```

8. PROCEDURE

Prosedur/*procedure* adalah subprogram yang menerima masukan tetapi tidak mempunyai luaran secara langsung. Cara mendeklarasikan sebuah prosedur adalah sebagai berikut :

```
{ Nama Procedure adalah A }
procedure A;
begin
      statement_11;
      statement_21;
      …
      statement_N1;
end;
```

Perhatikan contoh-contoh berikut:

Contoh Procedure I

```
program Jumlah;
uses Crt;
var  Bil_1, Bil_2, Jumlah: Integer;

procedure Hitung_Jumlah;                                          { Awal Procedure }
begin
    Jumlah := Bil_1 + Bil_2;
end;                                                             { Akhir Procedure }

{ Awal Program Utama }
begin
    Clrscr;
    Write('Masukkan Bilangan Pertama: '); Readln(Bil_1);
    Write('Masukkan Bilangan Kedua  : '); Readln(Bil_2);

    Hitung_Jumlah; (* Pemanggilan Procedure *)

    Writeln;
    Writeln('Jumlah Kedua Bilangan    : ', Jumlah);
    ReadKey;
end.
```

Contoh Procedure II

```
Program Procedure_Aritmatika;
Uses Crt;
Var  Bil_1, Bil_2: Integer;

{Procedure Jumlah}
Procedure Jumlah(Bil_1, Bil_2: Integer);
Var Jumlah: Integer;
Begin
     Jumlah := Bil_1 + Bil_2;
     WriteLn('Jumlah Kedua Bilangan   : ', Jumlah);
End;


{Procedure Kurang}
Procedure Kurang(Bil_1, Bil_2: Integer);
Var Kurang: Integer;
Begin
     Kurang := Bil_1 - Bil_2;
     WriteLn('Selisih Kedua Bilangan  : ', Kurang);
End;

{Procedure Kali}
Procedure Kali(Bil_1, Bil_2: Integer);
Var Kali: Real;
Begin
     Kali := Bil_1 + Bil_2;
     WriteLn('Perkalian Kedua Bilangan: ', Kali: 7:2);
End;

{Procedure Bagi}
```

```
Procedure Bagi(Bil_1, Bil_2: Integer);
Var Bagi: Real;
Begin
     Bagi := Bil_1 / Bil_2;
     WriteLn('Pembagian Kedua Bilangan: ', Bagi: 7:2);
End;

{Program Utama}
{------------}
Begin
    Clrscr;
    Write('Masukkan Bilangan Pertama : '); ReadLn(Bil_1);
    Write('Masukkan Bilangan Kedua   : '); ReadLn(Bil_2);
    Write('------------------------------'); WriteLn;
    Jumlah(Bil_1, Bil_2);
    Kurang(Bil_1, Bil_2);
    Kali(Bil_1, Bil_2);
    Bagi(Bil_1, Bil_2);
    ReadKey;
End.
```

9. REKURSIF

9.1 DEFINISI REKURSIF

Salah satu keistimewaan yang dimiliki Pascal adalah bahwa Pascal bisa melakukan suatu proses yang disebut dengan proses **rekursif**. Rekursif berarti suatu proses yang bisa memanggil dirinya sendiri.

Dalam rekursif sebenarnya terkandung pengertian prosedur dan fungsi. Perbedaannya adalah bahwa rekursif bisa memanggil dirinya sendiri, tetapi prosedur dan fungsi harus dipanggil lewat pemanggil prosedur atau fungsi.

Contoh rekursif yang paling sederhana adalah :

- Menghitung nilai faktorial dari bilangan bulat positif.
- Mencari deret fibonacci dari suatu bilangan bulat.

Nilai faktorial, secara rekursif dapat ditulis sebagai berikut :

$0! = 1$

$N! = N \times (N - 1)!$ Untuk $N > 0$

Secara notasi pemrograman bisa ditulis sebagai berikut :

FAKTORIAL (0) = 1 (1)

FAKTORIAL (N) = N * FAKTORIAL (N – 1) (2)

Persamaan (2) diatas merupakan contoh hubungan rekurens yang berarti bahwa nilai suatu fungsi dengan argumen tertentu bisa dihitung dari fungsi yang sama dengan argumen yang lebih kecil. Persamaan (1) yang tidak bersifat rekursif disebut nilai awal. Bilangan fibonacci bisa didefinisikan berdasarkan deret integer tak berhingga sebagai berikut :

1 1 2 3 5 8 12 13 21 34 35 89 . . .

Dari deret diatas bisa dilihat bahwa bilangan ke N (N > 2) bisa dicari bilangan sebelumnya yang terdekat dengan bilangan ke N yaitu bilangan ke (N – 1) dan ke (N – 2). Sehingga fibo (N) bisa dihitung :

Fibo (N) = Fibo (N – 1) + Fibo (N – 2)

9.2 PROSES REKURSIF

Untuk memahami proses rekursif yang terjadi dalam sebuah fungsi rekursi, perhatikan contoh – contoh program dibawah ini :

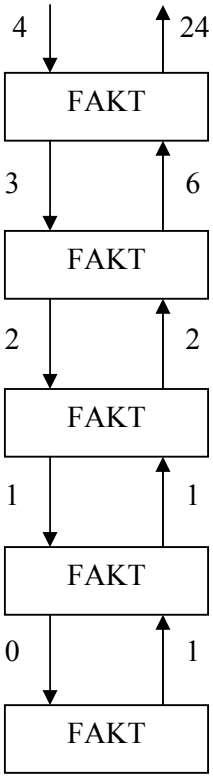
- Program mencari faktorial secara rekursif.

```
function FAKT ( N : integer ) : integer;
begin
    if  N = 0 then
      FAKT := 1
   Else
     FAKT := N * FAKT ( N – 1 )
end;
```

- Program deret fibonacci dengan cara rekursif.

```
function FIBO ( N : integer ) : integer;
begin
    if  ( N = 1 ) or ( N = 2 ) then
      FIBO := 1
    else
      FIBO ( N ) := FIBO ( N – 1 ) + FIBO ( N – 2 );
end;
```

Ilustrasi fungsi faktorial dengan proses rekursif :



Ilustrasi fungsi faktorial :

Fakt (6) = 6 * Fakt (5)

Fakt (5) = 5 * Fakt (4)

Fakt (4) = 4 * Fakt (3)

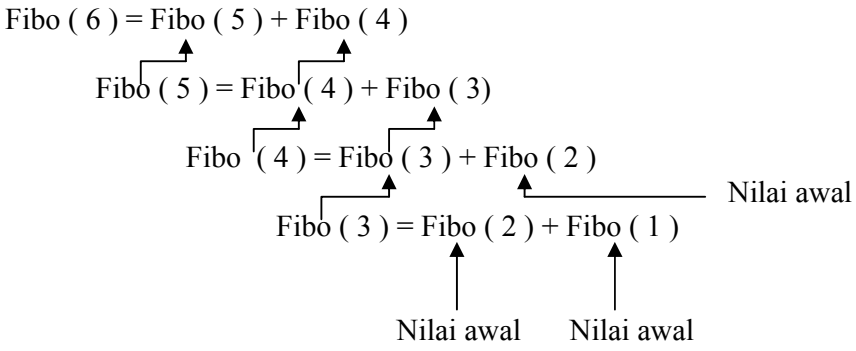
Fakt (3) = 3 * Fakt (2)

Fakt (2) = 2 * Fakt (1)

Fakt (1) = 1* Fakt (0)

Nilai awal

Ilustrasi fungsi Fibonacci :



Fibo (2) dan Fibo (1) = 1

Contoh :

1. N = 4

   Fibo (4) = Fibo (3) + Fibo (2)

   = Fibo (3) + 1

   = Fibo (2) + Fibo (1) + 1

   = 1 + 1 + 1

   = 3

SOAL ALGORITMA & PEMROGRAMAN

(buatlah algoritma untuk setiap kasus dibawah ini)

1. **Persoalan:** Misalkan diketahui tabel *A* yang berukuran *n* elemen sudah berisi nilai *integer*. Kita ingin menentukan nilai minimum dan nilai maksimum sekaligus di dalam tabel tersebut.

2. **Persoalan**: Diberikan himpunan titik, *P*, yang terdiri dari *n* buah titik, (x_i, y_i), pada bidang 2-D. Tentukan jarak terdekat antara dua buah titik di dalam himpunan *P*.

3. **Persoalan**: Diberikan uang senilai *A*. Tukar *A* dengan koin-koin uang yang ada. Berapa jumlah minimum koin yang diperlukan untuk penukaran tersebut?

   Contoh: tersedia koin-koin 1, 5, 10, dan 25
   Uang senilai 32 dapat ditukar dengan cara berikut:
   32 = 1 + 1 + … + 1 (32 koin)
   32 = 5 + 5 + 5 + 5 + 10 + 1 + 1 (7 koin)
   32 = 10 + 10 + 10 + 1 + 1 (5 koin)
   … dan seterusnya

   Minimum: 32 = 25 + 5 + 1 + 1 hanya 4 koin

4. **Persoalan**: Diberikan
   teks (*text*), yaitu (*long*) *string* yang panjangnya *n* karakter
   pattern, yaitu *string* dengan panjang *m* karakter ($m < n$) yang akan dicari di dalam teks.
   Carilah lokasi pertama di dalam teks yang bersesuaian dengan *pattern*.

5. **Persoalan :** Misalkan tabel *A* berisi elemen-elemen berikut:
   8 1 4 6 9 3 5 7
   Urutkan data diatas

6. **Persoalan :** Misalkan tabel *A* berisi elemen-elemen berikut:
   8 1 4 5 9 3 5 7
   Tentukan sebuah bilangan ada di urutan ke berapa saja (misal : 5 ada di urutan 4 dan 7)

7. **Persoalan :** Misalkan $a \in R$ dan *n* adalah bilangan bulat tidak negatif:
   $a^n = a \times a \times \ldots \times a$ (sebanyak *n* kali) , jika $n > 0$
   $= 1$, jika $n = 0$
8. **Persoalan:** Misalkan bilangan bulat *X* dan *Y* yang panjangnya *n* angka (masalah ini dapat dirampatkan sebagai *n byte* atau *n word*):
   $X = x_1x_2x_3 \ldots x_n$
   $Y = y_1y_2y_3 \ldots y_n$

   Hitunglah hasil kali *X* dengan *Y*.

Contoh 4.8. Misalkan,

$X = 1234$ $(n = 4)$
$Y = 5678$ $(n = 4)$

9. **Persoalan** : Terdapat barisan berikut
   1 3 9 5 25 7 49 9 81 ... n
   Buatlah algoritma untuk menanpilkan baris diatas dengan batas n

10. **Persoalan** : Misal diinput 11 buah bilangan :
    2 5 7 4 5 8 5 4 2 7 4
    Buat algoritma untuk mencari modus dari n bilangan yang diinput
    Untuk kasus diatas, modus adalah 4 dan 5

SOAL PILIHAN

Algoritma dengan pseudopascal berikut dimaksudkan untuk menjumlahkan bilangan bilangan pada suatu array tabeldata hanya pada elemen array bernomor indeks kelipatan 3 (yaitu: 3, 6, 9, dst...) sampai dengan elemen ke 30 dan mencetak hasilnya ke layar. Diketahui, array tabeldata berindeks dari 1 sampai dengan 40.

```
sum := 0;
 // inisialisasi i
while i < hargabatas do
begin
   sum := sum + tabeldata[i];
   //increment i
end;
writeln(sum);
```

SOAL

1. Berapakah harga untuk menginisialisasi i pada baris "// inisialisasi i" agar algoritma bekerja seperti yang diharapkan?
   A. 3
   B. 0
   C. 1
   D. 2
   E. 4
2. Perintah manakah yang harus diberikan menggantikan "// increment i" ?
   A. i := i + 3
   B. i := i + 1
   C. i := i - 1
   D. i := 3
   E. i := i + 2
3. Berapakah variabel hargabatas seharusnya diberi harga agar algoritma bekerja seperti yang diharapkan?
   A. 31
   B. 25

C. 30
D. 35
E. 40

4. Apa yang akan dicetak oleh algoritma jika setiap elemen array berisi harga yang sama dengan nomor indeksnya jika algoritma dituliskan sebagaimana yang seharusnya?
   A. 165
   B. 135
   C. 145
   D. 176
   E. 30

DESKRIPSI

berikut ini struktur **if-then-else**
if (a and not (not c and not b)) or not ((c and b) or not a)
then writeln('merah')
else writeln('putih');

SOAL

1. Agar algoritma itu selalu menuliskan 'merah' maka kondisi yang tepat adalah
   A. a dan c keduanya harus true sementara b tidak penting
   B. b berharga true yang lain tidak penting
   C. a berharga true yang lain tidak penting
   D. c berharga true yang lain tidak penting
   E. b berharga false yang lain tidak penting

2. Pemeriksaan ekspresi lojik (di antara notasi **if** dan **then**) tersebut bisa digantikan dengan ekspresi berikut
   A. (a and (c or b)) or (not (c and b) and a)
   B. ((a <> c) or (a = b) or b)
   C. ((a = c) and (a <> b) and not b)
   D. (a and c or b) or (not c or not b and a)
   E. a and not b and not c

3. Jika a berharga false maka
   A. algoritma selalu menuliskan 'putih' apapun harga b dan c
   B. algoritma selalu menuliskan 'merah' apapun harga b dan c
   C. algoritma selalu menuliskan 'putih' jika salah satu lainnya true
   D. algoritma selalu menuliskan 'putih' b dan c true
   E. algoritma selalu menuliskan 'putih' b dan c false

DESKRIPSI

Perhatikan program berikut:

// array X berisi n bilangan
dengan index dari 1 s.d. n

```
// fungsi Max(a,b) adalah mencari bilangan terbesar dari a atau b
ts := 0;
ms := 0;
for i := 1 to n do
begin
  ts := Max(0, X[i] + ts);
  ms := Max(ts, ms);
end;
writeln(ms);
```

SOAL

1. Jika array berisi harga-harga sebagai berikut: 1, -3, 4, -2, -1, 6 (berarti juga n berharga 6) maka algoritma akan mencetak harga?
   A. 7
   B. 4
   C. 2
   D. 6
   E. -6
2. Jika array berisi harga-harga sebagai berikut: 1, -1, 1, -1, 1, -1, 1 (berarti juga n berharga 7) maka algoritma akan mencetak harga?
   A. 1
   B. -1
   C. 4
   D. -3
   E. 7
3. Jika fungsi Max(a,b) diimplementasikan manakah yang anda pilih paling benar dan paling efisien menurut waktu eksekusi?
   A. if (a > b) then Max := a else Max := b;
   B. if (a < b) then Max := a else Max := b;
   C. Max := b; if (a > b) then Max := a;
   D. Max := a; if (a > b) then Max := b;
   E. if ((a - b) > 0) then Max := a else Max := b;
4. Jika jumlah data adalah N maka berapakalikah fungsi Max(a,b) akan dipanggil selama algoritma itu dijalankan?
   A. $2N$
   B. N
   C. $N/2$
   D. N^2
   E. $\log(N)$

DESKRIPSI

Perhatikan algoritma berikut.

```
procedure Bingo(t);
begin
  if (t < 2) then
    writeln('Bingo!')
  else
```

```
  begin
    Bingo(t-1);
    Bingo(t-2)
  end;
end;
```

SOAL

1. Berapa kalikah 'Bingo!' dituliskan jika procedure tersebut dipanggil dengan Bingo(6)?
   A. 13
   B. 8
   C. 6
   D. 1
   E. 2

DESKRIPSI

Perhatikan algoritma berikut.

```
procedure Boo(t: integer);
begin
  if (t > 0) then
  begin
    for i := 1 to t do writeln(\'Boo!\');
   Boo(t div 2); // t dibagi 2 dan dibulatkan ke bawah
  end;
end;
```

SOAL

1. Berapa kalikah 'Boo!' dituliskan jika procedure tersebut dipanggil dengan Boo(k) dimana k adalah suatu bilangan 2^N?
   A. 2^{N+1}-1
   B. 2^N-1
   C. 2^{N+1}
   D. $2N$
   E. N
2. Pemanggilan Boo(1000) menghasilkan pencetakan "Boo!" sebanyak?
   A. 1993 baris
   B. 2000 baris
   C. 1000 baris
   D. 500 baris
   E. 10 baris
3. Untuk menghasilkan pencetakan "Boo!" sebanyak 200 kali memerlukan pemanggilan dengan?
   A. Boo(102)
   B. Boo(200)
   C. Boo(100)
   D. Boo(16)
   E. Boo(1000)

DESKRIPSI

Suatu array X berindeks dari 1 s.d. 10 dan setiap elemennya berisi huruf-huruf berurutan dari 'a' sampai 'j'.

SOAL

1. Suatu algoritma bekerja pada array tersebut sbb

```
for i := 1 to 10 do
swap(X[i],X[10-i+1]; // prosedur swap menukarkan
            // kedua isi elemen array tsb
for i := 1 to 10 do write(X[i]);
```

   Hasil yang dicetak adalah

   A. abcdefghij
   B. jihgfedcba
   C. ebacdhfgij
   D. fghijabcde
   E. cdefghijab

2. Suatu algoritma bekerja pada array tersebut sbb

```
procedure lagi(a: integer; b: integer);
var t: integer;
begin
   t := (a+b) div 2;
   if (a <= b) then begin
      write(X[t]);
      lagi (a,t-1);
      lagi (t+1,b);
   end
end;
```

   dengan pemanggilan:

   lagi(1,10);

   Hasil yang akan dicetak adalah

   A. ebacdhfgij
   B. abcdefghij
   C. jihgfedcba
   D. fghijabcde
   E. cdefghijab

3. Suatu algoritma bekerja pada array tersebut sbb

```
for i := 2 to 9 do
  swap(X[i-1],X[i+1]; // prosedur swap menukarkan kedua
                // isi elemen array tsb
```

```
for i := 1 to 10 do write(X[i]);
```

Hasil yang akan dicetak adalah

A. cdefghijab
B. ebacdhfgij
C. abcdefghij
D. jihgfedcba
E. fghijabcde

SOAL

1. Penggalan deklarasi program sbb :

```
type ar = array[1..10] of byte ;
var a, ar ; I, b : byte ;
procedure baca( var x : ar ) ;
begin
        for i := 1 to 10 do readln(x[i]) ;
end ;
function jumlah( x : ar ) : byte ;
var j : byte ;
begin
        j := 0 ;
        for i := 1 to 10 do
        j := j + x[i] mod i ;
end ;
begin
        baca( a ) ; b := jumlah(a) ; writeln( b ) ;
end ;
```

Bila data yang dibaca berurutan sbb : 1, 2, 3, 4 ,5 ,6 ,7 ,8, 9, 10,
maka hasil output yang dicetak :

A. 10
B. 55
C. 0
D. 15
E. salah semua

2. Masih berhubungan dengan No. 6 di atas. Bila data yang dibaca secara berurutan sbb: 10,9,8,7,6,5,4,3,2,1, maka output yang dicetak adalah :

A. 10
B. 22
C. 20
D. 18
E. Salah semua

3. Penggalan program pascal sbb:

```
function fakto( n : byte ) : longint ;
begin
    if n = 0 then fakto := 0
    else fakto := n * fakto(n-1) ;
end ;
```

function di atas bila dipanggil dengan statemen y := fakto(5) ; maka y berharga :
A. 0
B. 5
C. 120
D. 250
E. salah semua

4. Masih berhubungan dengan penggalan program no. 8 di atas, bila statement
fakto := n * fakto(n-1)
diganti dengan
fakto := n + fakto(n-1)
dan dipaggil dengan
y := fakto (6)
maka y berharga :
A. 21
B. 24
C. 6
D. 2
E. salah semua

5. Penggalan program pascal sbb:
```
type kal = string[20] ;
procedure pqr( x : kal , n : byte) ;
begin
    if n > 0 then
            begin write(x[n]); pqr(x,n-1) ; end ;
end;
```
Bila dipanggil procedure dengan pqr('TOKI97',6) ;
maka output yang dicetak adalah:
A. TOKI97
B. 79IKOT
C. TOKI
D. OKIT
E. Salah semua

6. Bila kata : string[20] = 'kasur nababan rusak' ; procedure pqr no. 10 di atas dipanggil dengan statemen pqr(kata, length(kata)); maka output yang dihasilkan adalah :
A. rusak kasur nababan
B. nababan kasur rusak
C. kasur nababan rusak
D. rusak nababan kasur
E. salah semua

7. Penggalan program pascal sbb:
```
x := 0 ; y := 0 ;
while x < 10 do
begin x := x + 1 ; y := y + x ; end ;
writeln( x , y ) ;
```
Berapa harga x yang dicetak ?
A. 9

B. 10
C. 11
D. 14
E. salah semua

8. Perhatikan program di bawah ini:

```
Var
J:integer;
Begin
    J:=0;
    I:=25;
    While I>0 do
    Begin
            Dec(I);
            J:=j+I mod 5; {baris ke 9}
            I:=I shr 1;
    End;
    Write(j);
End.
```

Berapa nilai akhir variabel J yang dicetak di layar?
A. 10
B. 0
C. 7
D. 40
E. 55

9.
```
var i,j : integer;
begin
    j := 1;
    for i := 1 to 5 do
    begin
            writeln(i,' ',j);
            j := i-1;
    end;
end.
```

Output dari program di atas adalah

A.	B.	C.	D.
1 1	1 1	1 0	1 1
2 1	2 2	2 1	2 0
1 3	3 3	3 2	3 1
4 1	4 4	4 3	4 2
1 5	5 5	5 4	5 3

10. Bagaimana keluaran program di bawah ini?

```
Var a,b:integer;
Begin
    For a:=1 to 5 do
    begin
    b:=10 – a;
    write(a,b);
    end;
End.
```

A. 1 9 2 8 3 7 4 6 5 5
B.
1 9
2 8
3 7
4 6
5 5
C. 1928374655
D.
1
9
2
8
3
7
4
6
5
5
E. 1, 9, 2, 8, 3, 7, 4, 6, 5, 5

11. Apa output dari program di bawah ini?

```
Uses crt;
var I,j:integer;
begin
    I:=0;
    while I<5 do
    for j:=1 to 8 do
    I:=I+j;
    writeln(I,'-',J);
end.
```

A. 5 – 3
B. 5 – 8
C. 6 – 3
D. 3 – 3
E. 36 – 8

12. Perhatikan penggalan program berikut

```
Var
i,j: Shortint;
Begin
    for i:=1 to 200 do
    Inc(j);
end.
```

Program diatas akan menghasilkan:
A. Nilai j = 200;
B. Nilai j = 127;
C. Nilai j = 129;
D. Terjadi error (code 76).
E. Program tidak dapat berhenti

13. Manakah identifier yang benar untuk menamakan nama program :
A. prog-a

B. prog_a
C. prog+a
D. prog//a

14. Manakah ekspresi yang tidak dapat dilakukan oleh tipe data real
    A. a := a + b
    B. a := a - b
    C. a := a * b
    D. a := a mod b

15. Tipe data di bawah ini mana yang tidak dapat melakukan operasi aritmatika
    A. integer
    B. byte
    C. real
    D. Boolean

16. Yang memiliki hierarchi paling tinggi di antara operasi logika AND, OR dan NOT adalah
    A. AND
    B. NOT
    C. OR
    D. semua sama

17. Yang paling rendah dari operasi logika AND, OR dan NOT adalah :
    A. AND
    B. NOT
    C. OR
    D. semua sama

18. Manakah ekpresi yang tidak dapat dilakukan oleh tipe data integer
    A. x := x + 5
    B. y := y - z
    C. k := k / 5
    D. r := r * s

19. Manakah yang salah dari deklarasi di bawah ini
    A. program a; const k=10;
    B. program abc ; const p:6;
    C. program pqr ; const r:=8;
    D. program satu; const x>y;

20. Manakah yang benar dari deklarasi di bawah ini
    A. program abc; var x, y : real;
    B. program pqr; var x,y=byte;
    C. program xyz ; var x:=boolean;
    D. program klm; var z,y=char;

21. Manakah yang salah dari deklarasi di bawa ini :
    A. var a = array[1..5] of integer ;
    B. var a : array[0..5] of byte ;
    C. var a : array[2..6] of byte ;

D. var a : array[1..5] of real ;

22. Manakah deklarasi yang salah di bawah ini :
    A. type x = array[1..10] of byte ;
    B. type y : array[1..10] of integer ;
    C. type z = array[0..9] of real ;
    D. type w = array[3..10] of integer ;

23. Manakah yang salah dari deklarasi di bawah ini
    A. function abc(a: byte): real
    B. function xyz(x: byte) : byte ;
    C. function klm(k,l : byte) : real ;
    D. function stu(s: real) ;

24. Mana yang salah dari deklarasi di bawah ini
    A. procedure abc ;
    B. procedure abc(a: byte) ;
    C. procedure xyz(var x : integer) ;
    D. procedure klm(a, b : real) : real ;

25. Penggalan deklarasi program sbb :

```
type ar = array[1..10] of byte ;
var a, ar ; I, b : byte ;
procedure baca( var x : ar ) ;
begin
for i := 1 to 10 do readln(x[i]) ;
end ;
function jumlah( x : ar ) : byte ;
var j : byte ;
begin
j := 0 ;
for i := 1 to 10 do j := j + x[i] mod i ;
end ;
begin
baca( a ) ; b := jumlah(a) ; writeln( b ) ;
end ;
```

    Bila data yang dibaca berurutan sbb : 1, 2, 3, 4 ,5 ,6 ,7 ,8, 9, 10,
    maka hasil output yang dicetak :
    A. 10
    B. 55
    C. 0
    D. salah semua

26. Masih berhubungan dengan No. 13 di atas. Bila data yang dibaca secara berurutan sebagai berikut: 10,9,8,7,6,5,4,3,2,1 maka output yang dicetak adalah:
    A. 10
    B. 22
    C. 20
    D. salah semua

27. Masih berhubungan dengan soal No. 13 di atas. Bila judul procedure baca diubah dari procedure baca(var x : ar) ; menjadi procedure baca (x : ar), dan menggunakan data pada soal no. 14 di atas, maka output yang dicetak adalah
    A. 10
    B. 22
    C. 0
    D. salah semua

28. Penggalan program pascal dideklarasikan sbb :
    type ari = array[1..5] of byte ;
    var x : ari ; i , j : byte ;
    .....
    for i := 1 to 5 do x[i] := 10 div i ;
    j := 0 ; for i := 5 downto 1 do j := j + x[i] ;
    writeln(j) ;
    output yang dicetak dari penggalan program di atas adalah :
    A. 5
    B. 22
    C. 22.8
    D. salah semua

29. Masih menggunakan soal no. 16 di atas, bila array diubah dari 5 menjadi 10, begitupun juga pada for i : 1 to 10 do dan for i := 10 downto 1 do, maka hasil outputnya sebagai berikut :
    A. 28.3
    B. 28
    C. 27
    D. salah semua

30. Penggalan program pascal sbb :
    y := 0 ;
    for i := 1 to 5 do
    for j := 5 downto i do
    y := y + i ;
    writeln(y) ;
    hasil output yang dicetak adalah :
    A. 15
    B. 20
    C. 35
    D. salah semua

31. Masih menggunakan penggalan program no. 18 di atas, bila statement y := y + i; diganti dengan y := y + j; maka output yang dicetak adalah :
    A. 55
    B. 45
    C. 35
    D. salah semua

32. Masih menggunakan penggalan program no. 18 di atas bila statement y := y + i; diganti dengan y := y + 1; maka output yang dicetak adalah :

A. 5
B. 10
C. 15
D. salah semua

33. Penggalan program pascal sbb:
```
function fakto( n : byte ) : longint ;
begin
    if n = 0 then fakto := 0
    else fakto := n * fakto(n-1) ;
end ;
```
function di atas bila dipanggil dengan statemen
y := fakto(5) ; maka y berharga :
A. 0
B. 5
C. 120
D. salah semua

34. Masih berhubungan dengan penggalan program no. 21 di atas, bila statemen fakto := n * fakto(n-1) diganti dengan fakto := n + fakto(n-1) dan dipaggil dengan y := fakto (6) maka y berharga :
A. 21
B. 24
C. 6
D. salah semua

35. Berikut ini penggalan program pascal :
```
function abc( a, b : byte ) : longint ;
begin
    if b = 0 then abc := 1
    else abc := a * abc(a, b-1) ;
end ;
```
Bila fucntion abc di atas dipanggil dengan
x := abc(5,3) ;
maka x berharga :
A. 15
B. 125
C. 1
D. salah semua

36. Masih berhubungan dengan function no. 23. di atas, fucntion abc dipanggil dengan statemen x := abc (3, 5) maka x berharga :
A. 81
B. 15
C. 243
D. salah semua

37. Masih berhubungan dengan function abc soal no. 23 di atas, bila statemen abc := a * abc(a, b-1) diganti dengan abc := a + abc(a,b-1) dan dipanggil dengan x := abc(3,4) maka x berharga :
A. 13

B. 12
C. 10
D. salah semua

38. Penggalan program pascal sbb:
```
type kal = string[20] ;
procedure pqr( x : kal , n : byte) ;
begin
    if n > 0 then
    begin write(x[n]); pqr(x,n-1) ; end ;
end;
```
Bila dipanggil procedure dengan pqr('TOKI97',6) ;
maka output yang dicetak adalah:
A. TOKI97
B. 79IKOT
C. TOKI
D. Salah semua

39. Bila kata : string[20] = 'kasur nababan rusak' ; procedure pqr no. 26 di atas dipanggil dengan statemen pqr(kata, length(kata)); maka output yang dihasilkan adalah :
A. rusak kasur nababan
B. nababan kasur rusak
C. kasur nababan rusak
D. salah semua

40. Penggalan program pascal sbb:
```
x := 0 ; y := 0 ;
while x < 10 do
begin x := x + 1 ; y := y + x ; end ;
writeln( x , y ) ;
```
Berapa harga x yang dicetak ?
A. 9
B. 10
C. 11
D. salah semua

41. Masih berhubungan dengan soal no. 28 di atas, berpa harga y yang dicetak ?
A. 55
B. 45
C. 36
D. salah semua

42. Penggalan program pascal sbb :
```
a := 10 ; b := 0 ;
repeat b := b + 1 ; a := a - b ; until a > b ;
writeln ( a, b ) ;
```
Berapa harga a yang dicetak ?
A. 10
B. 1
C. 9

D. salah semua

43. Masih berhubugan dengan no. 30 di atas, berapa harga b yang dicetak ?
    A. 1
    B. 10
    C. 9
    D. salah semua

44. Deklarasi yang tepat adalah
    A. var A: String;
    B. type A = record;
    C. type MagicNumber: Integer;
    D. Jawaban (a) – (c) benar semua

45. Pernyataan manakah yang boleh dipakai untuk mendeklarasikan variabel dalam Pascal?
    A. var A, B;
    B. var A: Real, B: Integer;
    C. var A; B: Integer;
    D. var A: Pointer; B: Integer;

46. Bagaimana cara mendeklarasikan konstanta bertipe?
    A. var A: Integer const = 100;
    B. const A: Integer = 100;
    C. const A = 100;
    D. var A: Integer = 100;

47. Carilah deklarasi yang tidak diperbolehkan:
    A. const A = 14; B = A * 2.5;
    B. const E1 = 'Division by zero';
       E2 = 'Overflow';
       E3 = 'Invalid argument';
       ErrMsg: array[1..3] of String = (E1, E2, E3);
    C. const A = 123; B = Chr(A);
    D. Semua jawaban salah

48. Deklarasi mana yang salah dalam bahasa Pascal?
    A. var A: String[70];
    B. var A = String;
    C. var A: String[1024];
    D. var A: String[1..75];

49. Bagaimana cara mendeklarasikan array dua dimensi?
    a. var A: array[1..10, 1..10] of Char;
    b. var B: array[1 to 10, 1 to 10] of Char;
    c. var C: matrix[1..10, 1..10] of Char;
    d. var D: array[10, 10] of Char;

50. Deklarasi manakah yang benar?
    A. type Anggota = record

```
    Nama: String[40];
    NomorAnggota: Word;
    End;
```

B. type Anggota: record
   Nama = String;
   NomorAnggota = Word;
   End;

C. type Anggota := record
   Nama := String[20];
   NomorAnggota := Word;
   End;

D. type Anggota = record
   Nama: String[];
   NomorAnggota: Word;
   end;

51. Deklarasi manakah yang dapat diterima?
    A. var F: File of Text;
    B. var F: File of Integer;
    C. var F: File[1..10] of Integer;
    D. var F: Text of Char;

52. Deklarasi manakah yang salah?
    A. type Rec := record
       X, Y: Real;
       end;
       var F: File of Rec;
    B. var F: File;
    C. type A = array[1..10] of Byte;
       var F: File of A;
    D. var F: File of array;

53. Deklarasi manakah yang salah?
    A. var E: record X, Y: Integer end;
    B. var F: File of set of Char;
    C. var G: record
       F: File;
       U: Integer;
       end;
    D. Tidak ada deklarasi yang salah

54. Deklarasi manakah yang salah?
    A. var A: set of Real;
    B. var B: array[1..10,1..10] of record
       Flag: set of Byte;
       R: Real;
       end;
    C. var C: set of (Nasi, Roti, Bakmi, Bakpau);
    D. Tidak ada deklarasi yang salah

55. Deklarasi manakah yang benar?

A. var R: array[True..False] of String;
B. type P = set of Byte;
   var R: array[P] of String;
C. var Q: set of String;
D. Salah semua

56. Manakah yang salah?
A. function Gamma(I: Real): Real;
B. function Gamma(var I: Real): Real;
C. function Gamma(var I): Real;
D. function Gamma(I: Real);

57. Penulisan program Pascal yang dibenarkan adalah

A program Sederhana;
```
begin
var C: Word;
C := 275 * 400 + 5;
Writeln(C);
end.
```

B{ program Sederhana; }
```
var W: Word;
begin
W := 61224;
Writeln(W - 50000);
end.
```

C program Sederhana;
```
var A: String;
begin
A := ' tahun lalu belum ada komputer';
A := 200 + A;
Writeln(A);
end.
```

D program Matematika;
```
var Jari-jari: Real;
begin
Write('Jari-jari lingkaran = ');
Readln(Jari-jari);
Writeln('Keliling=',2 * Pi * Jari-jari);
end.
```

58. Tipe data manakah yang dapat memuat bilangan pecahan?
A. Byte
B. Integer
C. Boolean
D. Real

59. Manakah yang *salah*:

A var X: Integer;
```
begin
for X := 1 to 25 do Writeln(X);
end.
```

B.Var X: Char;
begin
for X := '1' to '9' do Writeln(X);
end.

C. var X: LongInt;
begin
for X := -50 downto -400 do Writeln(X);
end.

D. var X: Real;
begin
for X := 0 to 5 do Writeln(X);
end.

60. Tipe data manakah yang sanggup memuat nilai *numerik* sebesar 75000?
    A. Integer
    B. Real
    C. Word
    D. String

61. Berapakah jangkauan tipe data Word?
    A. 0…65536
    B. 0…65535
    C. -32768…32767
    D. Salah semua

62. Turbo Pascal menyediakan tipe data enumerasi yang didefinisikan dengan menyebutkan nama-nama elemennya oleh pemakai. Bila tipe data Nilai dapat bernilai Jelek, Kurang, Cukup, dan Baik, bagaimanakah cara menulisnya?
    A. type Nilai = (Jelek, Kurang, Cukup, Baik);
    B. type Nilai = Jelek, Kurang, Cukup, Baik;
    C. type Nilai = [Jelek, Kurang, Cukup, Baik];
    D. Salah semua.

63. Lihat kembali definisi Nilai di atas. Pernyataan manakah yang tidak benar?
    A. var Ket: array[Jelek..Baik] of Nilai;
    B. var Ket: array[Jelek..Baik] of String;
    C. var Ket: Jelek;
    D. Salah semua

64. Misalkan N adalah variabel bertipe Nilai (lihat soal di atas), maka operasi manakah yang tidak dibenarkan untuk N?
    A. Inc(N);
    B. N := N + Jelek;
    C. Writeln(Ord(N));
    D. if N >= Cukup then Writeln('Lulus');

65. Berikut ini adalah deklarasi sebuah program:
    type IsiRoti = (Coklat, Pisang, Nanas);
    var Isi: IsiRoti;
    Manakah yang salah di antara pernyataan-pernyataan berikut?
    A. if Isi in IsiRoti then

```
    Writeln('Nah, ini baru enak.');
B. if Isi in [Coklat, Nanas] then
    Writeln('Saya suka itu!');
C. if 'Coklat' in Isi then
    Writeln('Suka roti coklat juga ya?');
D. if IsiRoti = [] then
    Writeln('Harus pilih salah satu!');
```

66. Dalam potongan program berikut,

```
var Z1, Z2, Z3, Z4: set of Byte;
begin
   Z1 := [1, 2, 6, 8, 9];
   Z2 := [3, 6, 8, 12, 14];
   Z3 := Z1 + Z2;
   Z4 := Z1 * Z2;
end.
```

Berapakah nilai akhir Z3 dan Z4?

A. Z3 = [6, 7, 8]
   Z4 = [1,2,3,4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]
B. Z3 = [6, 8]
   Z4 = [1, 2, 3, 6, 8, 9, 12, 14]
C. Z3 = [1, 2, 3, 6, 6, 8, 8, 9, 12, 14]
   Z4 = [6, 6, 8, 8]
D. Salah semua

67. Manakah yang salah bila

```
var Size: array[1..150] of Real;
R: Real;
I: Integer;
```

A. Size[I] := I;
B. Size[R] := 15.5;
C. Size[I] := R;
D. Tidak ada yang salah

Deklarasi variabel berikut ini digunakan untuk menjawab soal-soal berikut ini:

```
var A: Integer;
B: Boolean;
C: Integer;
D: Real;
E: Char;
F: Byte;
```

68. Dengan deklarasi variabel di atas, ekspresi manakah yang benar dalam bahasa Pascal?

A. B := A + C > D;
B. if A then C := C + A;
C. A := (B + C) * 7;
D. semua ekspresi salah

69. Lihat kembali deklarasi di atas. Ekspresi manakah yang tidak sah dalam bahasa Pascal?

A. C := A and F;
B. D := A + C / 10;
C. if E = F then Writeln('Nilainya sama.');
D. semua ekspresi valid

70. Lihat kembali deklarasi di atas. Ekspresi mana yang dibenarkan dalam bahasa Pascal?
    A. A := D div 10;
    B. if B and (F > 5) then Writeln('Ulangi!');
    C. C := A / 10;
    D. F := Ord(E + E);

71. Lihat kembali deklarasi di atas. Ekspresi mana yang tidak benar?
    A. Inc(E);
    B. Inc(D);
    C. D := D + 1;
    D. E := Succ(E);

72. Deklarasi mana yang salah dalam bahasa Pascal?
    A. var A: record
       Nama, Alamat: String;
       end;
    B. type InfoRec = record
       Nama, Alamat: String;
       end;
       var A: InfoRec;
    C. var A: InfoRec = record
       Nama, Alamat: String;
       end;
    D. Semua deklarasi di atas salah

Soal Membuat Program

Contoh Soal 1. Cari Maksimum
(Soal ini pernah diberikan dalam OSN 2003, Balikpapan)
Pembahasan oleh : Ilham Kurnia, Alumni TOKI 2003

Nama Program	: CARI.PAS
Batas *Run-time*	: 1 detik / test case
Nama File Masukan	: CARI.IN
Nama File Keluaran	: CARI.OUT

Buatlah sebuah program yang menghasilkan keluaran berupa sebuah program dalam Pascal standar yang mencari maksimum dari *N* (1 ? *N* ?13) bilangan tanpa menggunakan *loop*. Agar keluaran mudah dibaca dan dimengerti, maka aturan-aturan berikut harus dipenuhi:

1. Keluaran diawali dengan sebuah baris berisi "program Maksimum (input, output);" (tanpa tanda kutip).

2. Baris kedua berisi pendeklarasian variabel-variabel dengan tipe integer yang dibutuhkan. Nama variabel berasal dari *N* huruf pertama pada abjad (semuanya huruf kecil). Berikan tepat satu spasi antara var dengan variabel pertama, koma dengan variabel berikutnya, variabel terakhir dengan :, dan : dengan integer.
3. Baris ketiga berisi "begin".
4. Baris keempat berisi sebuah perintah read, yang harus membaca nilai semua variabel. Berikan tepat satu spasi antara koma dengan variabel berikutnya.
5. Baris-baris berikutnya berisi komparasi (bila diperlukan) antara variabel-variabel sehingga secara pasti ditemukan variabel yang menyimpan nilai tertinggi. Komparasi dilakukan dengan menggunakan perintah "if then else". Kondisi dari setiap perintah if adalah sebuah pertidaksamaan lebih besar (>). Bagian perintah "if then" dan "else" masing-masing berada pada baris yang berbeda. Urutan komparasi adalah sesuai dengan abjad. Jadi, komparasi variabel b dengan variabel d tidak boleh mendahului komparasi variabel b dengan variabel e dan sebagainya.
6. Variabel yang secara alfabetis muncul lebih dahulu harus menempati posisi sebelum tanda > pada setiap pengecekan kondisi. Beri tepat satu spasi antara perintah if dengan variabel pertama, variabel pertama dengan >, > dengan variabel kedua, dan variabel kedua dengan perintah then.
2. Berilah indentasi (jorokan) sebanyak 2 spasi pada baris-baris yang merupakan bagian perintah yang dilakukan bila kondisi yang ada terpenuhi (setelah if then, sebelum else). Beri pula indentasi dengan yang sama besar pada baris-baris yang merupakan bagian perintah yang dilakukan (setelah else) bila kondisi yang ada bernilai salah.
3. Apabila komparasi yang dilakukan telah dapat memastikan variabel mana yang menyimpan nilai terbesar, maka tulis isi perintah tersebut dengan menggunakan perintah "writeln(...)", misalnya writeln(d).
4. Harus ada tepat 2*N*-1 perintah writeln.
5. Hanya ada tepat tiga karakter titik koma (;) pada keluaran.
6. Akhiri keluaran dengan "end.".
7. Keluaran harus dapat di*compile* dengan baik.

Lihat contoh yang diberikan untuk mempermudah Anda mengerti mengenai aturan-aturan di atas.

FORMAT MASUKAN (Nama File: CARI.IN**)**
Masukan terdiri dari satu baris yang berisi bilangan bulat *N*.

CONTOH MASUKAN
3

FORMAT KELUARAN (Nama File: CARI.OUT**)**
Keluaran yang dihasilkan harus sesuai dengan ketentuan yang tertulis di deskripsi soal.

CONTOH KELUARAN
```
program Maksimum (input, output);
var a, b, c : integer;
begin
read(a, b, c);
if a > b then
```

```
if a > c then
writeln(a)
else
writeln(c)
else
if b > c then
writeln(b)
else
writeln(c)
end.
```

Catatan
Perhatikan baik-baik penggunaan spasi.

Contoh Soal 2. Menghitung Perulangan
(Soal ini pernah diberikan dalam OSN 2003, Balikpapan)
oleh : Ilham Kurnia, Alumni TOKI 2003

Nama Program	: HITUNG.PAS
Batas *Run-time*	: 1 detik / test case
Nama File Masukan	: HITUNG.IN
Nama File Keluaran	: HITUNG.OUT

Diberikan dua buah untaian huruf (*string*), hitung berapa kali string kedua muncul sebagai bagian dari string pertama. Asumsikan bahwa tiap kemunculan dari string kedua pada string pertama boleh saling menimpa (*overlap*). Panjang string pertama maksimal 10000, sementara panjang string kedua maksimal 200. String didefinisikan sebagai untaian karakter-karakter denga n kode ASCII 32 – 127 yang dibatasi oleh karakter-karakter dengan kode ASCII yang tidak termasuk dalam jangkauan 32 – 127 tersebut.

FORMAT MASUKAN (Nama File: HITUNG.IN**)**
Masukan terdiri dari dua baris. Baris pertama berisi string pertama, sementara baris kedua berisi string kedua.

CONTOH MASUKAN
```
abcdefabcghiabcabcjklmnlabcw
abc
```

FORMAT KELUARAN (Nama File: HITUNG.OUT**)**
Keluaran hanya terdiri dari sebuah baris berisi sebuah bilangan bulat yang menyatakan banyak kemunculan string kedua pada string pertama.

CONTOH KELUARAN
```
5
```

Contoh Soal 3. Perhiasan
(Soal ini pernah diberikan dalam OSN 2003, Balikpapan)
oleh : Ilham Kurnia, Alumni TOKI 2003

Nama Program	: **HIAS.PAS**
Batas *Run-time*	: **3 detik / test case**
Nama File Masukan	: **HIAS.IN**

Nama File Keluaran : **HIAS.OUT**

"Be Jeweled" adalah sebuah permainan populer di komputer palm. Ada 81 batu mulia yang disusun sebagai matriks 9 × 9. Tujuan dari permainan ini adalah untuk memperoleh skor setinggi-tingginya dari hasil sekali penukaran dua batu mulia yang bersebelahan pada suatu susunan. Dua batu mulia yang bersebelahan dapat ditukarkan apabila setelah pertukaran, ada beberapa batu mulia yang dapat dihitung nilainya.

Penilaian dilakukan dengan cara sebagai berikut:

1. Bila ada tiga atau lebih batu mulia dari jenis yang sama berjejer secara horisontal, maka batu-batu mulia tersebut akan dihitung.
2. Bila ada tiga atau lebih batu mulia dari jenis yang sama berjejer secara vertikal, maka batu-batu mulia tersebut akan dihitung.
3. Bila ada batu mulia yang memenuhi syarat 1 dan 2, maka batu mulia tersebut hanya dihitung sekali saja.
4. Skor yang diperoleh = 5 * *T* ^ (banyak batu mulia yang terhitung – 3), dimana *T* adalah tingkat kesulitan dari susunan/matriks batu-batu mulia (1 ? *T* ? 50).

A	B	C	D	E	F
B	D	R	P	R	A
C	P	P	R	P	B
D	B	C	P	B	C
E	A	B	P	A	D
F	D	F	E	F	E

Agar mudah, semua batu mulia akan direpresentasikan sebagai huruf besar. Ambil contoh submatriks (dengan tingkat kesulitan 3) berikut:

Tukar pasangan yang diarsir.

A	B	C	D	E	F
B	D	R	R	R	A
C	P	P	P	P	B
D	B	C	P	B	C
E	A	B	P	A	D
F	D	F	E	F	E

Batu mulia yang diarsir masuk dalam hitungan.
Untuk penukaran pasangan ini, skor yang diperoleh adalah $5 \times 3^{9-3} = 3645$

Tugas Anda adalah untuk membuat program yang mencari semua langkah terbaik (yaitu yang dapat menghasilkan skor tertinggi) dari sebuah matriks dengan ketentuan tertera di atas. Untuk memperjelas keluaran, maka hanya 2 macam pertukaran yang diperbolehkan: pertukaran dengan batu mulia sebelah kanan, atau pertukaran dengan batu mulia sebelah bawah.

FORMAT MASUKAN (Nama File: HIAS.IN**)**
Masukan terdiri dari 10 baris. Baris pertama berisi bilangan bulat *T*, sementara pada sembilan baris lainnya, setiap barisnya terdiri atas tepat 9 karakter. Setiap karakter melambangkan sebuah jenis batu mulia. Anda dapat berasumsi bahwa pa da masukan

tidak ada tiga atau lebih batu mulia yang telah berjejer secara horisontal maupun vertikal.

CONTOH MASUKAN
3
AAJXXEXXS
ASAJAXJEX
EJEXAXSJX
SAJXEJSJA
SXESJEJEX
AESJSSEXA
EJEEAAJSJ
EEAEXJASJ
AEXASSXJS

FORMAT KELUARAN (Nama File: HIAS.OUT**)**
Apabila tidak ditemukan satupun langkah pertukaran, maka pada keluaran hanya tulis "tidak ada" pada baris pertama. Bila ada langkah pertukaran, pada baris pertama tulis skor maksimum yang dapat diperoleh untuk matriks pada masukan. Pada baris -baris selanjutnya, tulis semua langkah terbaik yang ada, terurut membesar berdasarkan nomor baris, kemudian nomor kolom, dan terakhir arah (bawah terlebih dahulu). Formatnya adalah *Nomor_baris Nomor_kolom* bawah/kanan. Setiap angka dipisahkan oleh tepat satu spasi.

CONTOH KELUARAN
135
6 2 bawah
7 1 kanan
9 8 kanan

CONTOH MASUKAN 2
1
ABCDEFGHI
IHGFXDCBA
ABCDEFGHI
IHGFXDCBA
ABCDEFGHI
IHGFXDCBA
ABCDEFGHI
IHGFXDCBA
ABCDEFGHI

CONTOH KELUARAN 2
tidak ada

Catatan
Ada kemungkinan variabel dengan tipe integer, atau cardinal tidak dapat memuat dengan baik hasil keluaran untuk sejumlah kecil test case.

Contoh Soal 4. Nilai Terbesar
Pembahasan : Fajran Iman Rusadi, Alumni TOKI

Source Code	: MAXROUTE.PAS/C/CPP
Input	: MAXROUTE.IN
Output	: MAXROUTE.OUT
Waktu Eksekusi	: 2 detik

Perhatikan segitiga di bawah ini. Tulis sebuah program yang menghitung nilai terbesar dari angka-angka yang ada dalam jalur yang dimulai dari atas sampai ke bawah segitiga. Setiap langkah dalam jalur dapat secara diagonal ke bawah kiri atau kanan.

7
3 8
8 1 0
2 7 4 4
4 5 2 6 5

Jalur yang menghasilkan jumlah terbesar adalah 7-3-8-7-5 yang menghasilkan 30.

FORMAT INPUT
Baris pertama berisi sebuah bilangan bulat R (1 <= R <= 100) yang menyatakan ukuran segitiga. Baris ke dua sampai R+1 berisi bilangan-bilangan bulat (0 <= N <= 99) pada setiap baris segitiga.

CONTOH INPUT (File: MAXROUTE.IN)
5
7
3 8
8 1 0
2 7 4 4
4 5 2 6 5

FORMAT OUTPUT
Output hanya 1 baris yang terdiri dari 1 integer M, merupakan jumlah terbesar yang dapat dibuat dari spesifikasi di atas.

CONTOH OUTPUT (File: MAXROUTE.OUT)
30

Contoh Soal 5. Ekspresi Aritmatika
Pembahasan : Fajran Iman Rusadi, Alumni TOKI

Source Code	: EKSPRESI.PAS/C/CPP
Input	: EKSPRESI.IN
Output	: EKSPRESI.OUT
Waktu Eksekusi	: 1 detik

Program anda harus dapat membaca string masukan yang berisi ekspresi aritmetika yang terdiri atas operator ^, *, /, +, -, dan menuliskan urutan pengerjaannya yang benar. Misalnya:
a-b+c/d*e/f^g-h*j

Untuk menentukan urutan pengerjaan dalam penulisannya, operator -operator tersebut diberikan tingkat prioritas: ^ paling tinggi, kemudian * dan / pada prioritas yang sama, dan terakhir + dan – pada prioritas yang sama. Dengan adanya tingkatan prioritas ini maka f^g harus dikerjakan sebelum e/f atau g-h. Jika prioritas sama, maka yang disebelah 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 pengerjaaan suatu operasi, maka salah satu urutan pengerjaan ekspresi tersebut adalah:

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

FORMAT INPUT

Masukan terdiri dari satu baris teks ekspresi aritmetika dengan panjang < 256 karakter. Operator pangkat ditulis dengan simbol '^', 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.

FORMAT OUTPUT

Keluaran berisikan baris-baris operasi untuk mengerjakan ekspresi masukan yang dibantu oleh variabel-variabel sementara **xi**. 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 begantung pada hasil c/d. Variabel-variabel sementara **xi** dituliskan sebagai karakter **x** dan bilangan **i** dengan **i**
membesar dari baris pertama ke baris terakhir.

CONTOH INPUT (File: EKSPRESI.IN)

a-b+c/d

CONTOH OUTPUT (File: EKSPRESI.OUT)

x1=a-b
x2=c/d
x3=x1+x2

Contoh 2, EKSPRESI.IN

c/d*e/f^g

Contoh 2, EKSPRESI.OUT

x1=c/d
x2=x1*e

x3=f^g
x4=x2/x3

Contoh 3, EKSPRESI.IN
a-b+c/d*e/f^g-h*j

Contoh 3, EKSPRESI.OUT
x1=a-b
x2=c/d
x3=x2*e
x4=f^g
x5=x3/x4
x6=x1+x5
x7=h*j
x8=x6-x7

Contoh Soal 6. Keluarga Vito / *Vito's Family*

Keluarga gangster terkenal, Vito Deadstone, akan pindah ke Jakarta. Dia mempunyai keluarga yang sangat banyak di sana. Sejak itu, dia berkeinginan untuk mengunjungi sanak keluarganya lebih sering, dia ingin memilih rumah yang paling dekat. Vito ingin meminimalisasi jarak dari semua sanak keluarganya itu. Dia mengirimimu surat kaleng untuk memecahkan persoalannya.

The famous gangster Vito Deadstone is moving to New York. He has a very big family there, all of them living on Lamafia Avenue. Since he will visit all his relatives very often, he wants to find a house close to them. Indeed, Vito wants to minimize the total distance to all of his relatives and has blackmailed you to write a program that solves his problem.

Format Masukan

Masukan terdiri dari beberapa uji kasus. Baris pertama terdiri dari banyaknya uji kasus. Untuk tiap uji, kamu akan diberikan bilangan integer yang berkaitan dengan *r (0 < r < 500)* dan nomor jalan (juga dalam integer) *s1, s2, ..., si, ..., sr* dimana mereka tinggal (0 < *si* < 30.000).
Catatan : sanak keluarga tersebut dapat tinggal pada nomor jalan yang sama.

The input consists of several test cases. The first line contains the number of test cases. For each test case you will be given the integer number of relatives r (0 < r < 500) and the street numbers (also integers) s1, s2, . . . , si, . . . , sr where they live (0 < si < 30, 000).
Note that several relatives might live at the same street number.

Format Keluaran

Untuk setiap uji kasus, program harus menulis minimal jumlah dari jarak-jarak optimal dari rumah Vito ke setiap rumah sanak keluarganya. Jarak antara dua nomor jalan *si* dan *sj* adalah *dij* = |*si* – *sj*|.

For each test case, your program must write the minimal sum of distances from the optimal Vito's house to each one of his relatives. The distance between two street numbers si and sj is dij = |si − sj |.

Contoh Masukan
2
2 2 4
3 2 4 6

Contoh Keluaran
2
4

Contoh Soal 7. Soal-soal Dasar

a. Sebuah fungsi Fibonaci

```
function FIBO(N: Integer) : Integer;

begin
        if (N = 1) or (N = 2) then
                FIBO := 1;
        else
                FIBO(N) := FIBO(N – 1) + FIBO(N – 2);
end;
```

Jika pada fungsi tersebut di atas diberikan nilai N = 12 dan dieksekusi, maka akan dihasilkan nilai FIBO = ?

b. Sebuah program dengan fungsi xxx

```
Uses CRT;
Const Max = 10;

Var Gerakan,
        banyak : Integer;

Prosedur xxx(var c_gerak: integer; cacah : integer;
A, B, C : Char);
Begin
        If cacah > 0 then
        Begin
                xxx(c_gerak, cacah-1, a, b, c);
                c_gerak := succ(c_gerak);
                xxx(c_gerak, cacah-1, b, a, c);
        End;
End;

Begin
        Write('banyak : '); readln(banyak);
        xxx(gerakan, banyak, 'A', 'B', 'C');
End.
```

Berapa kalikah terjadinya perpindahan jika banyak dimasukkan nilai 5?

c. Prosedur Buble Sort

```
Procedure buble_sort(var A: larik, N: integer);
Var i, j : integer;

Begin
        For i:=1 to n-1 do
                For j:=1 to n – i do
                        If A[j] > A[j+1] then
                                Swap(A[j], a[j+1]);
End;
```

Jika diberikan nilai A adalah larik sebanyak 15 dan nilai N adalah 15, berapa kalikah maksimal terjadinya perulangan untuk j?

DAFTAR PUSTAKA

Cormen, Thomas H and Leiserson,Charles E. *Introduction to Algorithms.* McGraw-Hill. London. 2002.

Suryana Setiawan. *Pseudopascal (Versi Olimpiade Sains Bidang Informatika/ Komputer)*. Jakarta. 2006