Telkom University

Minggu 11
Pengurutan

Algoritma Pemrograman – CII1F4

Fakultas Informatika

2021

Pertanyaan!

Bagaimana suatu program sederhana bisa mengurutkan data mahasiswa berikut ini? Seperti apakah algoritmanya?

No	Student ID	Name	GPA
1	113210689	Harith	1.56
2	113212624	Johnson	3.19
3	113211834	Kimmy	1.32
4	113212925	Chou	3.68
5	113210520	Grock	1.45
6	113210223	Lunox	1.89
7	113212819	Karrie	1.05
8	113211273	Aldous	2.46
9	113211643	Franco	1.60
10	113211992	Selena	3.50

No	Student ID	Name	GPA
1	113212925	Chou	3.68
2	113211992	Selena	3.50
3	113212624	Johnson	3.19
4	113211273	Aldous	2.46
5	113210223	Lunox	1.89
6	113211643	Franco	1.60
7	113210689	Harith	1.56
8	113210520	Grock	1.45
9	113211834	Kimmy	1.32
10	113212819	Karrie	1.05

Materi Perkuliahan

01

Definisi Pengurutan

Pengertian, Manfaat dan Properti Pengurutan

02

Jenis-Jenis Pengurutan

Ilustrasi beragam metode pengurutan

03

Metode Seleksi

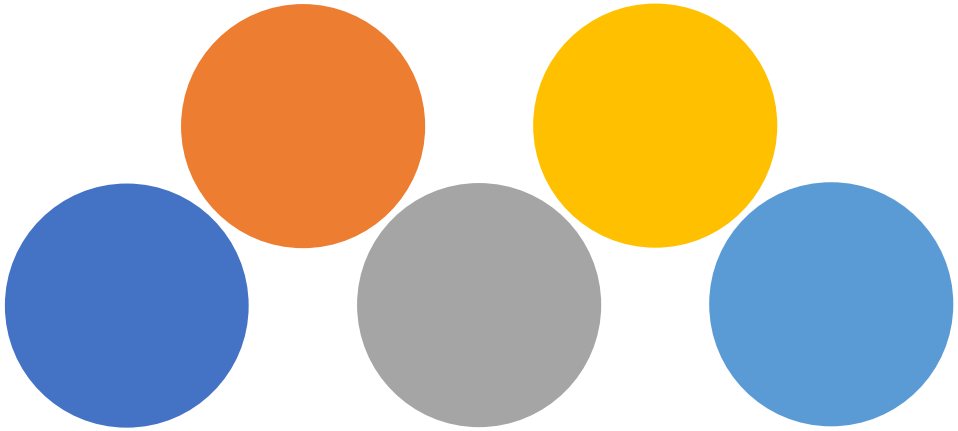
Memanfaatkan pencarian nilai ekstrim

04

Metode Insersi

Mencari posisi yang seharusnya

Definisi Pengurutan



Definisi Pengurutan

Definisi

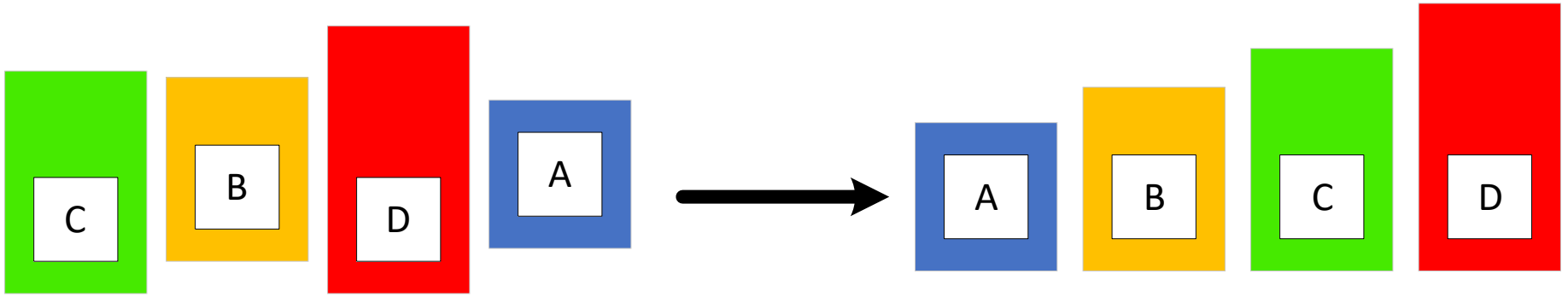
Pengurutan merupakan suatu proses untuk mengatur di mana posisi data yang seharusnya.

Keterurutan

- Membesar, Ascending, A-Z
- Mengecil, Descending, Z-A

Ruang Lingkup

Pengurutan pada kumpulan data di dalam Array



Manfaat Pengurutan

Manfaat Pengurutan

Banyak **manfaat** yang diperoleh jika kita tahu data yang kita miliki dalam keadaan terurut.
Contoh masalah komputasi:

- Pencarian nilai ekstrim (min/max).

- Pencarian nilai tengah/median.

- Pencarian dengan binary search.

- Pencarian data pada peringkat tertentu.

Properti data terurut

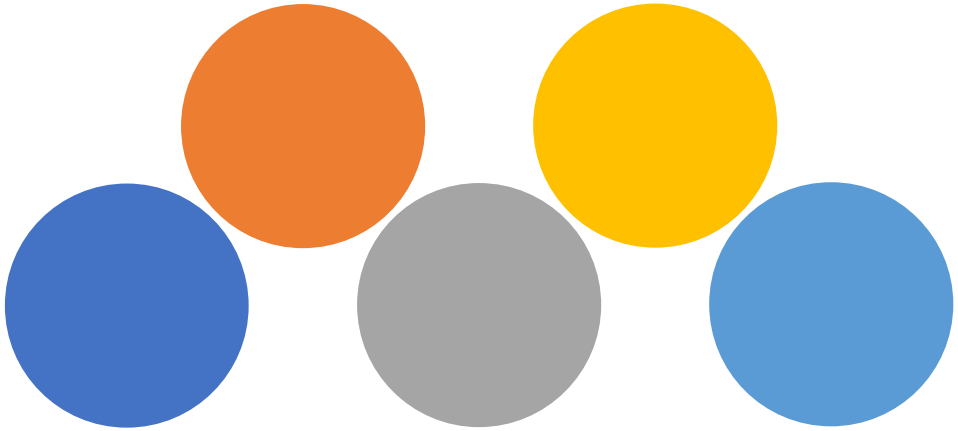
Misalkan data terurut membesar/ascending.

Data terbesar selalu berada pada indeks terbesar, begitu pula sebaliknya.

Untuk setiap data ke-i, maka seluruh data dengan indeks < i, juga mempunyai nilai ≤ data ke-i.

Selama data tidak bertukar tempat, maka keterurutan data tetap terjaga.

Jenis-Jenis Pengurutan



Berdasarkan Media Penyimpanan

Berdasarkan media penyimpanannya, pengurutan bisa dikelompokkan ke dalam dua kelompok, yaitu:

1. Pengurutan Internal

Pengurutan terhadap sekumpulan data yang disimpan dalam media internal (memori) komputer yang dapat diakses setiap elemennya secara langsung.

2. Pengurutan Eksternal

Pengurutan data yang disimpan dalam memori sekunder, biasanya data bervolume besar sehingga tidak mampu untuk dimuat semuanya dalam memori.

Pengurutan Internal

Di dalam pengurutan internal, terdapat berbagai variasi metode, mulai dari yang menggunakan ide yang paling sederhana hingga yang rumit sekalipun.

Tujuannya >> membuat data terurut

Perbedaan terletak pada waktu proses dan kapasitas memori yang digunakan*

*Performansi dan kompleksitas dari suatu algoritma akan dibahas pada mata kuliah **Analisis Kompleksitas Algoritma** dan **Strategi Algoritma**.

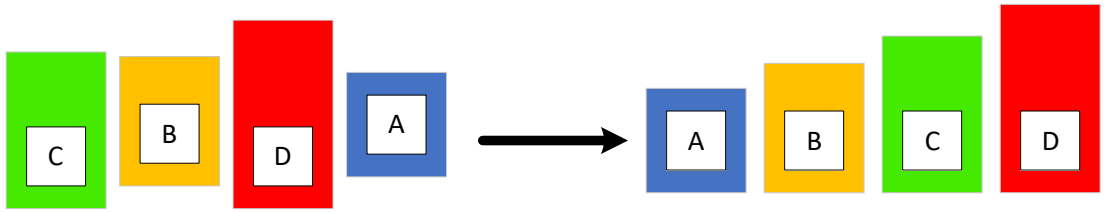
Pengurutan Internal

Berikut beberapa contoh algoritma pengurutan

Counting Sort	Maximum Sort	Insertion Sort	Bubble sort	Shaker sort
Heap Sort	Shell sort	Quick sort	Radix sort	dll

- Silahkan buka https://visualgo.net/en/sorting untuk melihat beberapa ilustrasi metode pengurutan.
- Untuk data dengan jumlah yang sangat banyak/besar, akan terlihat perbedaan waktu proses dari masing-masing algoritma.

Metode yang akan dipelajari



Secara umum ada 2 algoritma yang akan dipelajari:

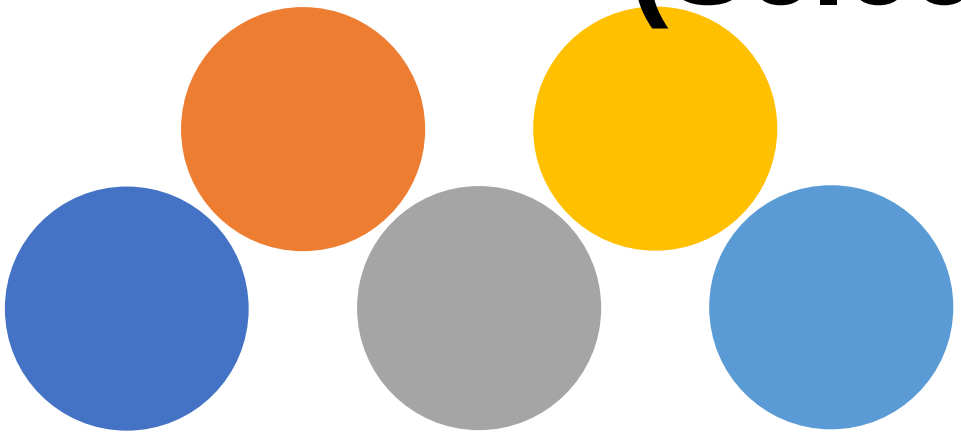
1. Pengurutan dengan metode seleksi (Selection Sort)
2. Pengurutan dengan metode insersi (Insertion Sort)

Mari kita mulai dengan contoh paling sederhana, yaitu dengan mengurutkan kumpulan bilangan.

Misalnya didefinisikan tipe data **`tabInt`** dengan maksimum elemen **`nMAX`**

```
kamus
    constant nMAX : integer = 1000
    type tabInt : array [0..nMAX-1] of integer
```

Metode Seleksi (Selection Sort)



Metode Seleksi

Ide dari pengurutan secara seleksi:

- Mencari posisi/indeks nilai maksimum/minimum pada array
- Kemudian menukarnya nilai maksimum/minimum dengan elemen terujung (kiri/kanan); elemen terujung ini "diisolasi" dan tidak diikut sertakan pada proses berikutnya.
- Proses diulang untuk sisa elemen Array.

SELECTION SORT

Ilustrasi Selection Sort

- Misalnya terdapat array of integer A

	0	1	2	3	4
A =	5	1	4	2	8

- Pengurutan secara **mengecil/descending**, artinya bilangan paling kiri adalah terbesar dan paling kanan adalah terkecil
- Attribut

  i : indeks elemen array A

  N : jumlah data, pada contoh N = 5

  idx : indeks nilai ekstrim berada

  Pass : tahapan dalam satu siklus pencarian nilai ekstrim dan tukar

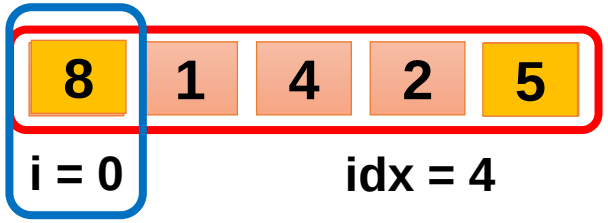
Ilustrasi Selection Sort

	0	1	2	3	4
A =	5	1	4	2	8

Pengurutan array A secara **descending**

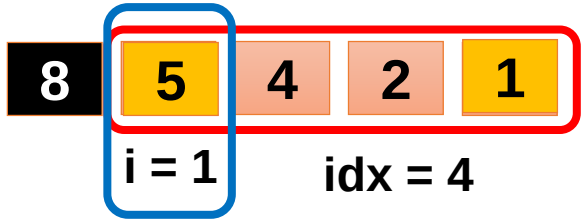
Nilai maksimum diletakkan diujung kiri

PASS = 1



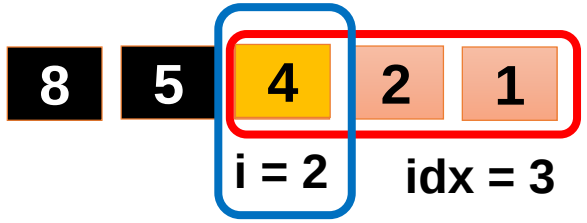
8 1 4 2 5

PASS = 2



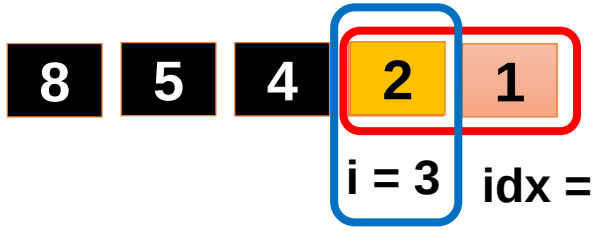
8 5 4 2 1

PASS = 3



8 5 4 2 1

PASS = 4



8 5 4 2 1

8 5 4 2 1

A terurut , sorting ini disebut juga **Maximum Sort**

Diskusi A

Berdasarkan ilustrasi sebelumnya dapat disimpulkan

1. Jumlah pass = N – 1
2. Setiap pass akan membagi array menjadi 2 bagian: (a) bagian terurut; (b) bagian tidak terurut.
3. Pencarian nilai ekstrim dimulai dari indeks ke-`pass-1`.
4. Pada proses tukar, nilai `A[idx]` ditukar dengan `A[pass-1]`
5. Bagaimana jika A telah terurut?
   Pengurutan tetap dilakukan, pada ilustrasi, array A telah terurut setelah pass 2.
   Alasannya :
   - Program tidak bisa melihat keterurutan data (tidak memiliki mata seperti manusia).
   - Program akan yakin data telah terurut apabila algoritma pengurutan telah selesai dijalankan.
6. Variasi Algoritma
   - Pencarian nilai maksimum dan minimum sangat bergantung pada nilai ujung yang dipilih, sebelah kiri atau kanan.
   - Pemilihan ini harus konsisten dari awal hingga akhir, dengan ujung kiri atau kanan saja.

Latihan Pemahaman 1

- Buatlah matrik berukuran 5x5 seperti gambar di bawah!
- Lakukan pengurutan array B dengan secara **ASCENDING**.
- Tuliskan isi array **setelah** melewati suatu Pass.

B	8	5	4	2	1
Pass 1					
Pass 2					
Pass 3					
Pass 4					

SOLUSI

B	8	5	4	2	1
Pass 1	1	5	4	2	8
Pass 2	1	2	4	5	8
Pass 3	1	2	4	5	8
Pass 4	1	2	4	5	8

Buatlah matrik berukuran 10x10 untuk mengurutkan 10 digit NIM (9 pass) anda (ascending/descending),

Latihan Pemahaman 2

Buatlah algoritma untuk menghasilkan susunan array setelah pass ke-3. Asumsi array telah terisi atau tersusun sedemikian rupa.

Poin Penting!

1. Pencarian nilai ekstrim dimulai dari indeks ke-`pass-1`.
2. Pada proses tukar, nilai `A[idx]` ditukar dengan `A[pass-1]`
3. `Algoritma dibangun berdasarkan illustrasi sebelumnya`

```
kamus
    constant nMAX : integer = 1000
    type tabInt : array [0..nMAX-1] of integer
    pass, idx, i, temp : integer
    A : tabInt
algoritma
    {asumsi A telah diisi pada baris ini}
    pass <- 3
    ...
    ...
```

Latihan Pemahaman 2

SOLUSI PASS ke-3

1. Pencarian nilai ekstrim dimulai dari indeks ke-pass-1.
2. Pada proses tukar, nilai `A[idx]` ditukar dengan `A[pass-1]`
3. Algoritma dibangun berdasarkan illustrasi sebelumnya

```
algoritma
    {asumsi A telah diisi pada baris ini sebanyak
     N nilai}
    pass <- 3
    {pencarian nilai maksimum}
    idx <- 2    {pass – 1}
    i <- 3      {pass}
    while i < N do
        if A[idx] < A[i] then
            idx <- I
        endif
        i <- i + 1
    endwhile
    {proses tukar}
    temp <- A[pass-1]
    A[pass-1] <- A[idx]
    A[idx] <- temp
```

Algoritma Selection Sort 1

Sehingga algoritma lengkap dalam bentuk subprogram adalah

```
procedure mengurutkan(in/out A : tabInt, in N : integer)
{I.S. terdefinisi array A yang berisi N bilangan bulat
 F.S. array A terurut secara DESCENDING dengan menggunakan algoritma Selection Sort}
kamus
    pass, idx, i, temp : integer
algoritma
    pass <- 1
    while pass <= N-1 do
        idx <- pass-1
        i <- pass
        while i < N do
            if A[idx] < A[k] then
                idx <- i
            endif
            i <- i + 1
        endwhile
        temp <- A[pass-1]
        A[pass-1] <- A[idx]
        A[idx] <- temp
        pass <- pass + 1
    endwhile
Endprocedure
```

Untuk menampilkan isi Arraynya,

gunakan procedure untuk menampikan isi dari array

Algoritma Selection Sort 2

Variasi yang lain, apabila dipecah menjadi beberapa subprogram.

```
procedure mengurutkan(in/out A : tabInt, in N : integer)
{I.S. terdefinisi array A yang berisi N bilangan bulat
 F.S. array A terurut secara DESCENDING dengan menggunakan algoritma Selection Sort}

function maksPos(A : tabInt, N, pass : integer) → integer
{mengembalikan posisi/indeks pencarian nilai maksimum pada array A dari indeks ke-pass-1 s.d. ke-N-1}

procedure tukar(in/out A : tabInt, in i, j: integer)
{I.S. terdefinisi array A dan bilangan bulat i dan j
 F.S. terjadi pertukaran nilai pada array A di indeks ke-i dengan indeks ke-j}
```

Algoritma Selection Sort 2

Poin Penting!

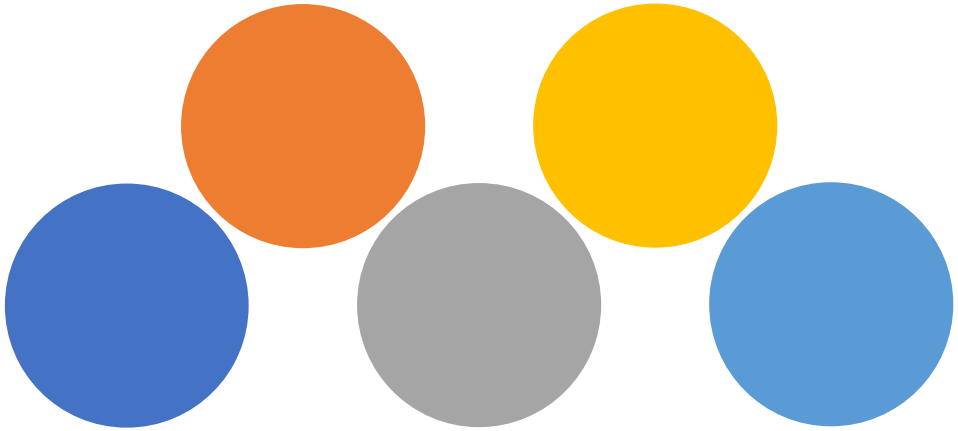
1. Jumlah Pass = N – 1 (Pass ke-1 s.d. ke N-1)
2. Pencarian nilai ekstrim dimulai dari indeks ke-pass-1.
3. Pada proses tukar, nilai A[idx] ditukar dengan A[pass-1]
4. Algoritma dibangun berdasarkan illustrasi sebelumnya

```
function maksPos(A : tabInt, N, pass : integer) → integer
{mengembalikan posisi/indeks pencarian nilai maksimum pada array A dari indeks ke-pass-1 s.d. ke-N-1}
kamus
    idx,i : integer
algoritma
    idx <- pass-1
    i <- pass
    while i < N do
        if A[idx] < A[i] then
            idx ← i
        endif
        i <- i + 1
    endfor
    return idx
Endfunction
```

```
procedure tukar(in/out A : tabInt, in i, j: integer)
{I.S. terdefinisi array A dan bilangan bulat i dan j
 F.S. terjadi pertukaran nilai pada array A di indeks ke-i dan j}
kamus
    temp : integer
algoritma
    temp <- A[i]
    A[i] <- A[j]
    A[j] <- temp
endprocedure

procedure mengurutkan(in/out A : tabInt, in N : integer)
{I.S. terdefinisi array A yang berisi N bilangan bulat
 F.S. array A terurut secara DESCENDING dengan menggunakan algoritma Selection Sort}
kamus
    pass, idx : integer
algoritma
    for pass <- 1 to N-1 do
        idx <- maksPos(A, N, pass)
        tukar(A,idx, pass-1)
    endfor
endprocedure
```

Latihan Soal



Soal 1. Mahasiswa

Apabila didefinisikan sebuah kamus seperti berikut ini, buatlah procedure untuk mengurutkan data mahasiswa (student) berdasarkan ipk (GPA) secara **ascending**, seperti contoh table yang diberikan.

```
kamus
    constant nMAX : integer = 2022
    type student < name, sid : string ; gpa : real >
    type tabMhs : array [0..nMAX-1] of student
```

No	Student ID	Name	GPA
1	113210689	Harith	1.56
2	113212624	Johnson	3.19
3	113211834	Kimmy	1.32
4	113212925	Chou	3.68
5	113210520	Grock	1.45
6	113210223	Lunox	1.89
7	113212819	Karrie	1.05
8	113211273	Aldous	2.46
9	113211643	Franco	1.60
10	113211992	Selena	3.50

No	Student ID	Name	GPA
1	113212819	Karrie	1.05
2	113211834	Kimmy	1.32
3	113210520	Grock	1.45
4	113210689	Harith	1.56
5	113211643	Franco	1.60
6	113210223	Lunox	1.89
7	113211273	Aldous	2.46
8	113212624	Johnson	3.19
9	113211992	Selena	3.50
10	113212925	Chou	3.68

Soal 2. Lengkapi Program

- Lengkapi program untuk berikut ini! Pahami program utama yang diberikan

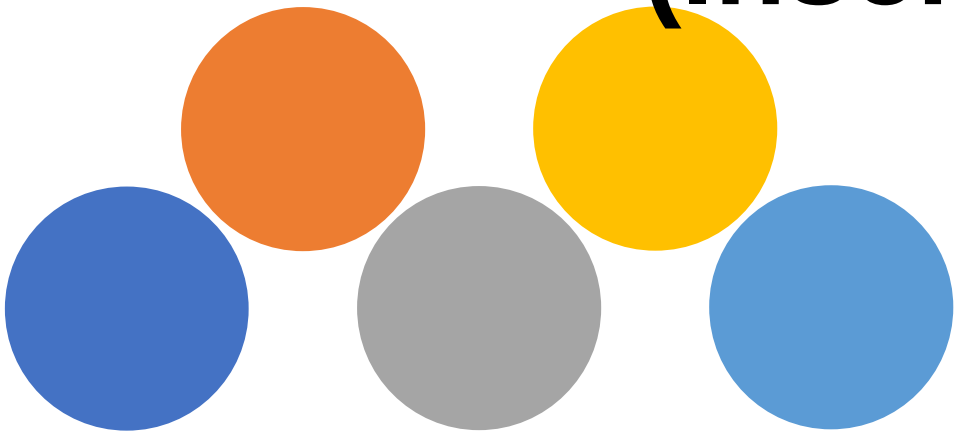
```
program SortAndSimilarityCheck
kamus
    constant MAX : integer = 30
    type IntArray : <
        tabInt : array [1..MAX] of integer,
        N : integer >
    array1, array2: IntArray
algoritma
    inputArray(array1)
    inputArray(array2)
    sort(array1)
    sort(array2)
    print(isSimilar(array1, array2))
endprogram
```

```
procedure inputArray(in/out T: IntArray )
{I.S. Input sudah siap pada piranti masukan
Process: Baca semua integer dari input, sampai ditemukan 0
F.S. array T berisi N elemen, integer 0 terakhir tidak
termasuk}

procedure sortArray(in/out T: IntArray )
{I.S. array T berisi N elemen, 0 < N ≤ MAX
F.S. T terurut dari kecil ke besar menggunakan Selection sort}

function isSimilar(T1, T2: IntArray ) -> boolean
{Mengembalikan true jika T1 dan T2 sama banyaknya dan elemen
pada kedua array dengan indek yang sama juga mempunyai nilai
yang sama}
```

Metode Insersi (Insertion Sort)



Bersambung…

TERIMA KASIH