## Kompetensi

Searching

1. Mahasiswa mampu menjelaskan mengenai algoritma *Searching*.
2. Mahasiswa mampu membuat dan mendeklarasikan struktur algoritma *Searching*.
3. Mahasiswa mampu menerapkan dan mengimplementasikan algoritma *Searching*.

## Alat Dan Bahan:

* + - PC/Leptop
    - Netbeans

## Ulasan Teori:

*Searching* merupakan proses untuk menemukan suatu data atau informasi dari sekumpulan data/informasi yang ada. Algoritma pencarian/*searching* algorithm merupakan algoritma yang menerima suatu kata kunci sebagai kriteria pencarian, dan dengan langkah-langkah tertentu akan mencari rekaman yang sesuai dengan kata kunci tersebut. Di dalam jobsheet ini akan diperkenalkan teknik pencarian data yaitu pencarian beruntun/sequential searching

## Pencarian Beruntun/Sequential Search

*Sequential Search* adalah teknik pencarian data dimana data dicari secara urut dari depan ke belakang atau dari awal sampai akhir. Kelebihan dari proses pencarian secara sequential ini jika data yang dicari terletak didepan, maka data akan ditemukan dengan cepat. Tetapi dibalik kelebihannya ini, teknik ini juga memiliki kekurangan. Pertama, jika data yang dicari terletak dibelakang atau paling akhir, maka akan membutuhkan waktu yang lama dalam proses pencariannya. Kedua, beban komputer akan semakin bertambah jika jumlah data dalam array sangat banyak.

Teknik pencarian data dari array yang paling mudah adalah dengan cara *sequential search,* dimana data dalam array dibaca 1 demi satu, diurutkan dari index terkecil ke index terbesar, maupun sebaliknya.

## Algoritma Pencarian Beruntun/Sequential Search

Algoritma pencarian berurutan dapat dituliskan sebagai berikut : 1. i ← 0

1. ditemukan ← false
2. Selama (tidak ditemukan) dan (i <= N) kerjakan baris 4
3. Jika (Data[i] = x) maka ditemukan ← true, jika tidak i ← i + 1
4. Jika (ditemukan) maka i adalah indeks dari data yang dicari, jika tidak data tidak ditemukan

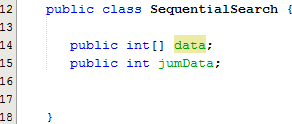
Contoh :

**int a[5] = {0,3,6,10,1}** (index array dimulai dari index ke 0) jika kita ingin mencari bilangan 6 dalam array tersebut, maka proses yang terjadi kita mencari

* 1. dari array index ke-0, yaitu 0, dicocokan dengan bilangan yang akan dicari, jika tidak sama, maka mencari ke index berikutnya
  2. pada array index ke-1, juga bukan bilangan yang dicari, maka kita mencari lagi pada index berikutnya
  3. pada array index ke-2, ternyata bilangan yang kita cari ada ditemukan, maka kita keluar dari looping pencarian.

## Langkah Praktikum:

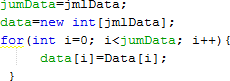
1. **Praktikum 1 (Pencarian Beruntun/Sequential Search )**
   1. Buatlah project baru pada Netbeans dengan nama “**TestSequentialSearch”**
   2. Buat class **“SequentialSearch”** , kemudian deklarasikan variabel berikut ini:



* 1. Buatlah konstruktor dengan parameter Data[] dan jmlData

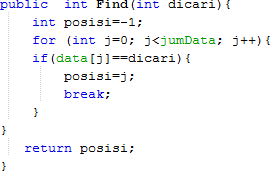


* 1. Isi konstruktor tersebut dengan kode berikut!



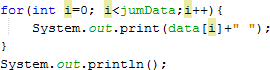
* 1. Buatlah method **“Find”** bertipe integer dengan parameter **“dicari”** bertipe integer.



* 1. Deklarasikan isi method Find dengan algoritma pencarian data menggunakan teknik sequential searching
  2. Buatlah method **“Tampilkan Data”** bertipe void



* 1. Deklarasikan isi dari method TampilData



* 1. Buatlah array pada class **“TestSequentialSearch”** dengan nama **data** kemudian isi array tersebut!
  2. Buatlah objek baru dengan nama **pencarian** yang merupakan instansiasi dari class SequentialSearch, kemudian isi parameternya!



* 1. Lakukan pemanggilan method **TampilkanData** pada class SequentialSearch.



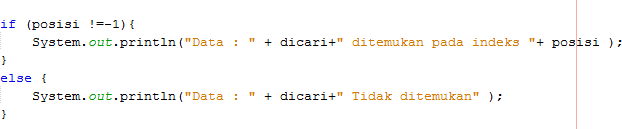
* 1. Deklarasikan variabel dicari bertpe integer dan isi nilainya dengan nilai yang ingin dicari



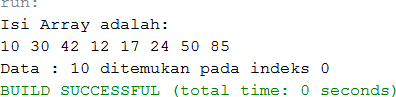
* 1. Deklarasikan variabel posisi bertipe integer yang memanggil method Find pada class SequentialSearch.



* 1. Lakukan pengecekan untuk kondisi data yang dicari.



* 1. Jalankan program, maka hasilnya adalah sebagai berikut:

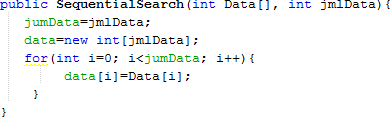


## Pertanyaan

* 1. Apa yang dimaksud dengan kode berikut ini:



* 1. Jelaskan fungsi kode program berikut ini!



* 1. Jelaskan fungsi **break** pada kode program dibawah ini!



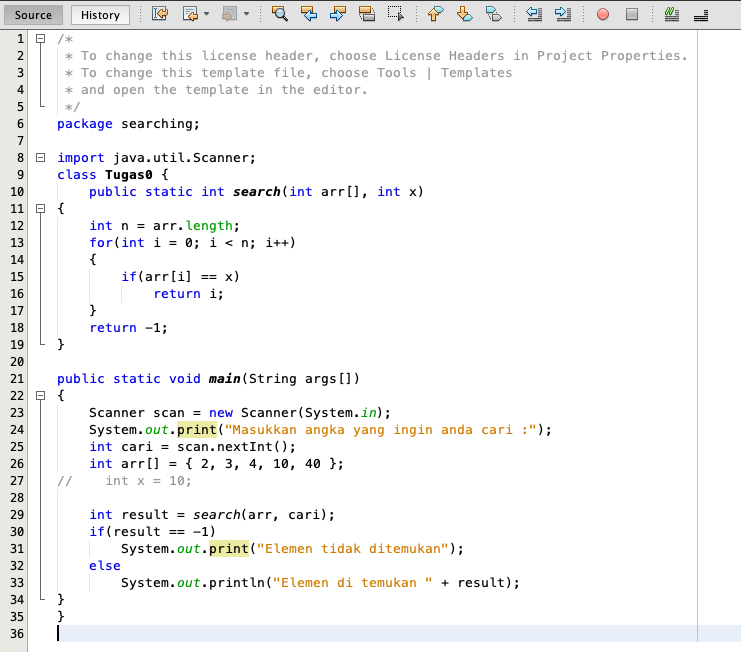
* 1. Modifikasilah program diatas yang mana data array dapat di inputkan atau bersifat dinamis!

## Tugas

1. Terdapat sebuah data array 2 dimensi sebagai berikut:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **indeks** | **0** | **1** | **2** | **3** | **4** |
| **0** | 45 | 78 | 7 | 200 | 80 |
| **1** | 90 | 1 | 17 | 100 | 50 |
| **2** | 21 | 2 | 40 | 18 | 65 |

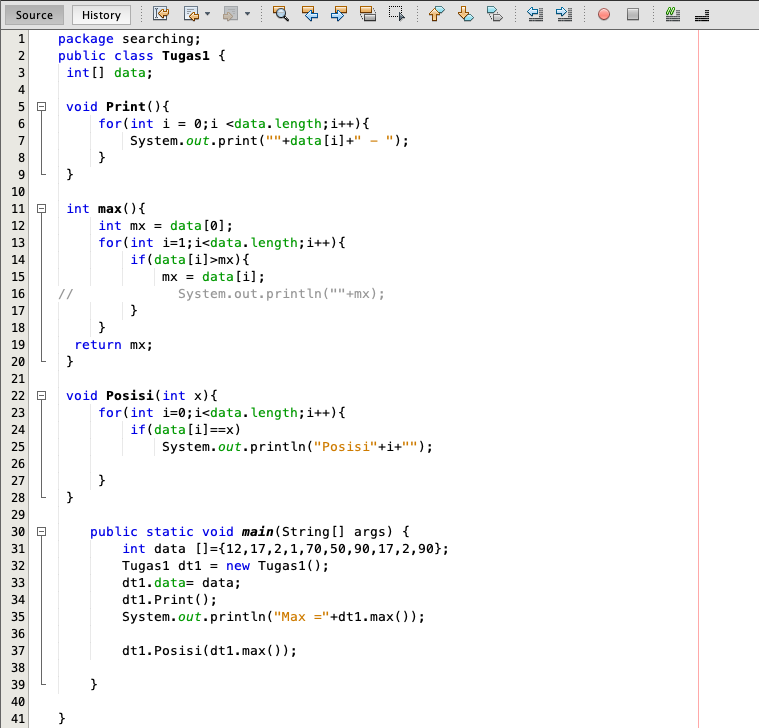
Berdasarkan data di atas buatlah program untuk melakukan pencarian data pada array dua dimensi tersebut, dimana data yang dicari di inputkan melalui keyboard!

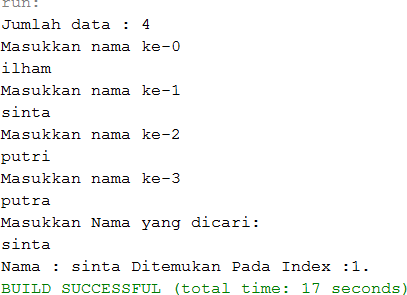


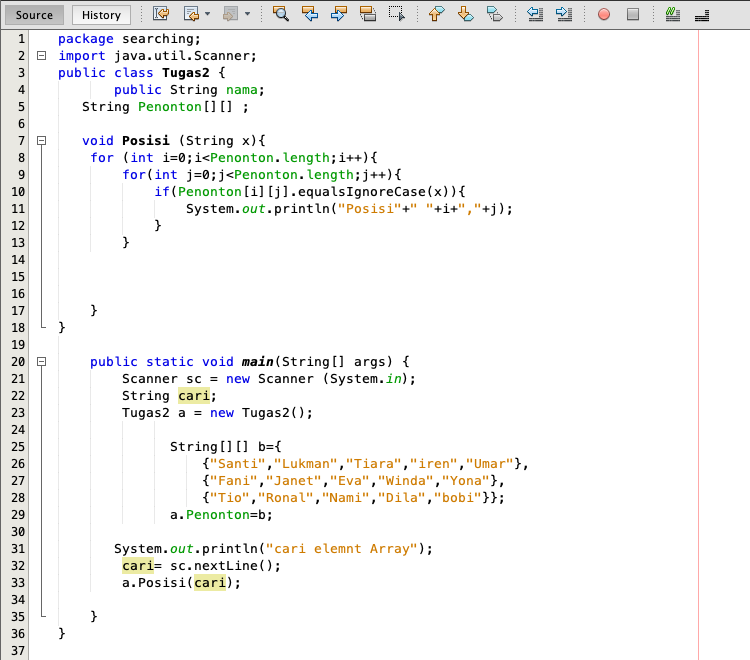
1. Terdapat sebuah data array 1 dimensi sebagai berikut

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| 12 | 17 | 2 | 1 | 70 | 50 | 90 | 17 | 2 | 90 |

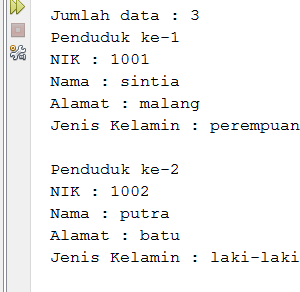
Buatlah program untuk mencari dan mencetak isi array yang nilainya terbesar, dan mencetak ada berapa buah nilai terbesar tersebut serta berada dilokasi mana saja nilai terbesar tersebut!

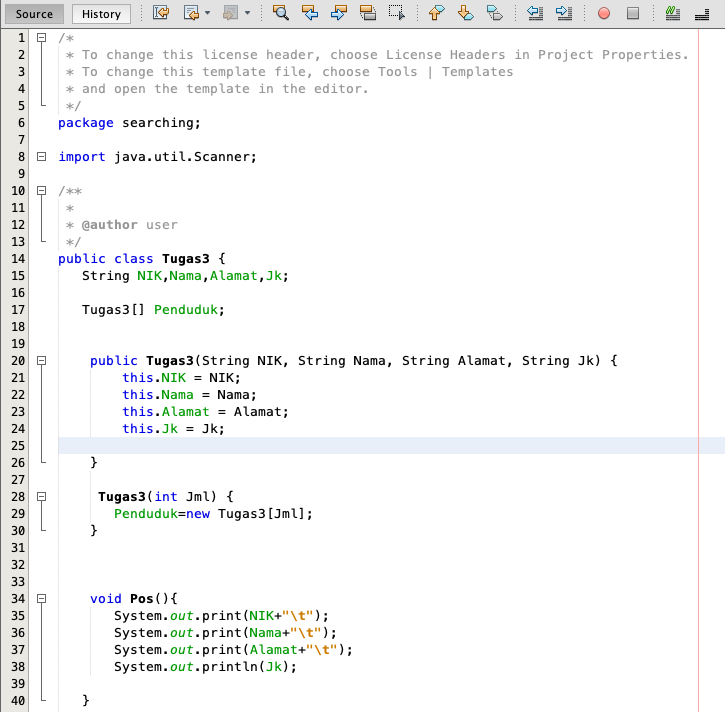


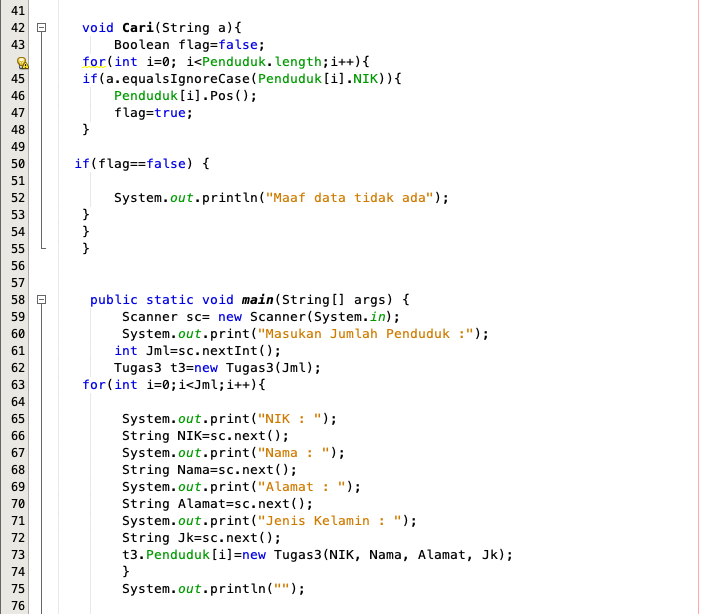
1. Buatlah program untuk pencaria data nama yang mana jumlah panjang data berupa inputan dan data nama tersebut di inputkan menggunakan keyboard!

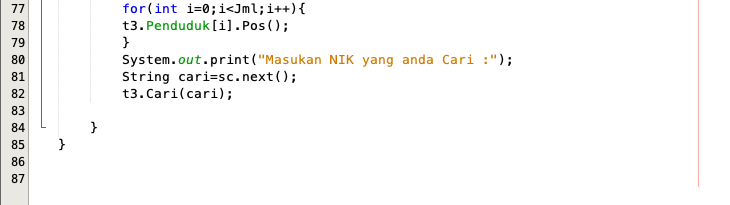


1. Buatlah program untuk melakukan pencarian data penduduk berdasarkan NIK yang mana data penduduk tersebut terdiri dari NIK, nama, alamat, dan jenis kelamin. Data penduduk dan jumlah data penduduk di inputkan melalui keyboard.









Penduduk ke-3

NIK : 1003

Nama : budi Alamat : malang

Jenis Kelamin : laki-laki

Data Kependudukan

{ Nn { NI K { NRNR { RLRNR'£ { IN I S KE LRN IN



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 1 | l | 1001 | l | sintia malang lperempuan l |
| l | 2 | l | 1002 | l | putra batu llaki-laki l |
| l | 3 | l | 1003 | l | budi malang llaki-laki l |



# t Cari Data Pendudukt Masukkan NIK : 1002 ndata di temukan...



{ Nn { NI K { NRNR { RLRNR'£ { IN I S KE LRN IN



# l 2 11002 l putra l batu llaki-laki l



BUILD SUCCESSFUL (total time: 1 minute 42 seconds)