Back End Development 1

Sesi 7

Algorithm

Algorithm - Sesi 7

Introduction

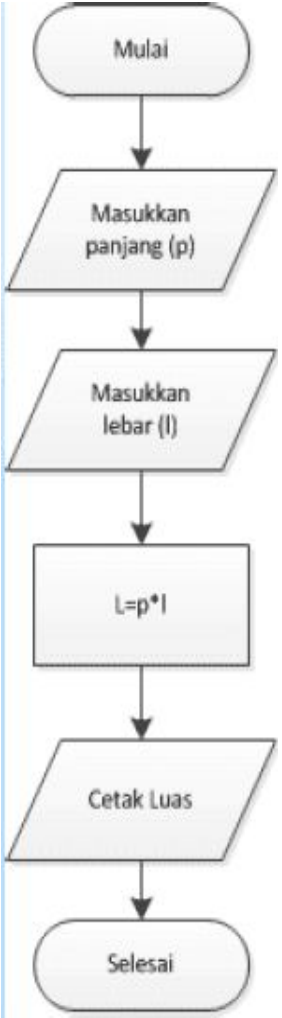
Walaupun algoritma bisa dibilang jantung ilmu komputer atau informatika, tetapi jangan beranggapan bahwa algoritma selalu identik dengan ilmu komputer saja.

Dalam kehidupan sehari-hari, terdapat banyak proses yang dinyatakan dalam suatu algoritma. Misal cara memasak mie, cara membuat kue, dan lainnya.

Jika kita buat algoritma memasak mie akan seperti di bawah ini:

- Siapkan 1 bungkus mie instan, 400 ml air (2 gelas), panci, mangkok, sendok, dan garpu
- Masukkan 400 ml air ke dalam panci
- Masak air
- Tunggu hingga mendidih
- Masukkan mie ke dalam panci yang sudah berisi air mendidih
- Tunggu dan aduk hingga 3 menit
- Jika sudah matang masukkan bumbu
- Aduk hingga rata
- Sajikan mie

atau




```
public class BelajarJava{
        public static void main(String[] args){
                System.out.println("Selamat Belajar Java");
        }
}
```

Selection Sort Algorithm

Sorting adalah salah satu algoritma penting yang ada di bidang pemrograman.

Sorting sering kali dimanfaatkan untuk mempermudah mendapatkan informasi tertentu secara cepat,

contohnya adalah ketika kita ingin mencari sebuah nama di dalam daftar kontak, kita cenderung akan melihat huruf pertama dari nama yang ingin kita cari dan kita akan membuka halaman dari daftar kontak yang merujuk ke halaman yang berisi kumpulan nama-nama yang diawali huruf tersebut.

Hal tersebut bisa kita lakukan karna daftar kontak tersebut sudah di-sorting berdasarkan alfabet.

Proses Selection Sort (Ascending)

Iterasi 1 :

13 9 15 2 3 1 → (Apakah 13 nilai yang paling kecil?)

1 9 15 2 3 13 → (Tidak. 13 ditukar dengan 1)

Iterasi 2 :

1 9 15 2 3 13 → (Apakah 9 nilai yang paling kecil?)

1 2 15 9 3 13 → (Tidak. 9 ditukar dengan 2)

Iterasi 3 :

1 2 15 9 3 13 → (Apakah 15 nilai yang paling kecil?)

1 2 3 9 15 13 → (Tidak. 15 ditukar dengan 3)

Iterasi 4 :

1 2 3 9 15 13 → (Apakah 9 nilai yang paling kecil?)

1 2 3 9 15 13 → (Iya)

Data Setelah di sorting ialah sebagai berikut :
Data : 1 2 3 9 13 15

Penjelasan Algoritma Selection Sort:

- Jumlah Iterasi untuk Selection Sort ialah berjumlah sebesar Jumlah Data – 1.

Untuk kasus diatas, Jumlah Datanya ialah 6. Maka, jumlah Iterasinya ialah sebesar 6 – 1 = 5.

- Proses pertukaran Data dimulai dari Data Pertama sampai Data Terakhir dengan cara membandingkan Data ke-n dan cari nilai yang paling kecil di sisi kanan nilai n.

- Keterangan bahwa nilai Data yang sudah di tukar(nilai yang paling kecil) tidak akan dibandingkan lagi untuk proses iterasi berikutnya. Berikut ilustrasi lengkapnya untuk kasus diatas.

```java
package Sesi6;

import java.util.Scanner;

public class SelectionSort
{
    Run | Debug
    public static void main(String[] args)
    {
        //    Buat Objek Scanner
        Scanner scan = new Scanner(System.in);

        //    Input jumlah Data
        System.out.print("Masukkan jumlah Data : ");    int jlh_data = scan.nextInt();

        //    Input nilai tiap Data
        int[] data = new int[jlh_data];        //    Array untuk nilai tiap Data
        System.out.println();
        for(int x = 0; x < jlh_data; x++)
        {
            System.out.print("Input nilai Data ke-"+(x+1)+" : ");
            data[x] = scan.nextInt();
        }

        //    Tampilkan Data Sebelum di sorting
        System.out.println();
        System.out.print("Data Sebelum di Sorting : ");
        for(int x = 0; x < jlh_data; x++)
            System.out.print(data[x]+" ");
```

```
25          System.out.println();
26          System.out.print("Data Sebelum di Sorting : ");
27          for(int x = 0; x < jlh_data; x++)
28              System.out.print(data[x]+" ");
29
30          //    Proses Selection Sort
31          System.out.println("\n\nProses Selection Sort");
32          for(int x = 0; x < jlh_data-1; x++)
33          {
34              System.out.println("Iterasi ke-"+(x+1)+" : ");
35              for(int y = 0; y < jlh_data; y++)
36                  System.out.print(data[y]+" ");
37
38              System.out.println("   Apakah Data "+data[x]+" sudah benar pada urutannya?");
39
40              boolean tukar = false;
41              int index = 0;
42              int min = data[x];
43              String pesan = "   Tidak Ada Pertukaran";
44              for(int y = x+1; y < jlh_data; y++)
45              {
46                  if(min > data[y])
47                  {
48                      tukar = true;
49                      index = y;
50                      min = data[y];
51
52                  }
53              }
```

```
55              if(tukar == true)
56              {
57                  //    Pertukaran Data
58                  pesan = "   Data "+data[x]+" ditukar dengan Data "+data[index];
59                  int temp = data[x];
60                  data[x] = data[index];
61                  data[index] = temp;
62              }
63
64              for(int y = 0; y < jlh_data; y++)
65                  System.out.print(data[y]+" ");
66
67              System.out.println(pesan+"\n");
68          }
69
70          //    Tampilkan Data Setelah di Sorting
71          System.out.print("Data Setelah di sorting : ");
72          for(int x = 0; x < jlh_data; x++)
73              System.out.print(data[x]+" ");
74      }
75  }
```

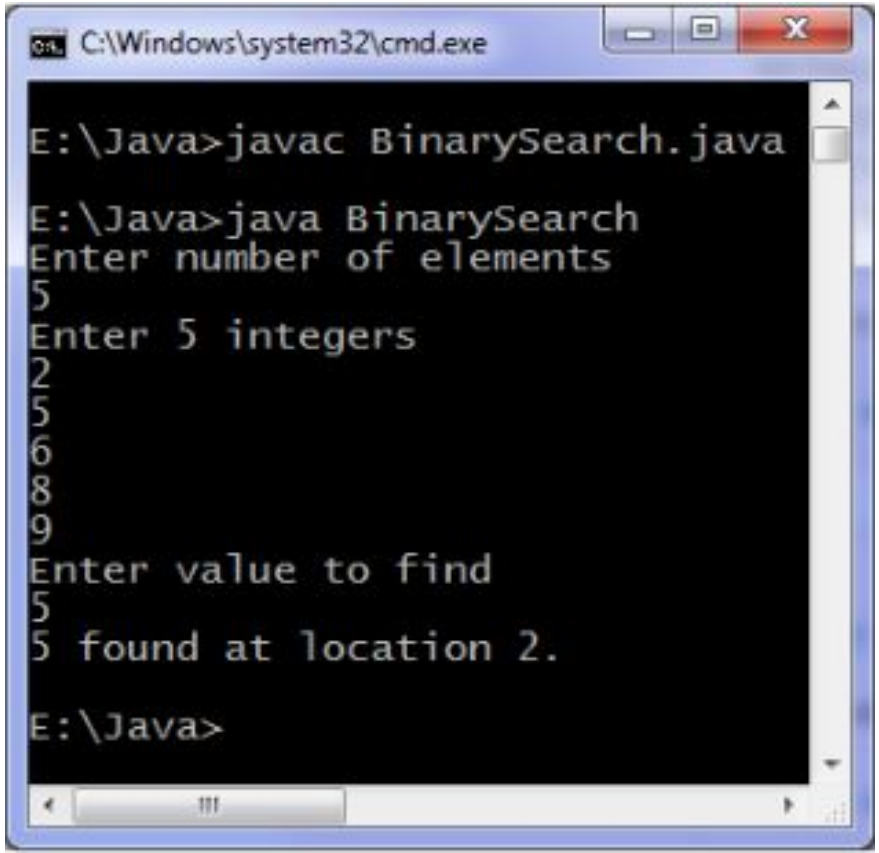
Algorithm - Sesi 6

Binary Search Algorithm

Binary search (pencarian biner) adalah algoritme pencarian yang paling populer. Algoritme ini relatif efisien sehingga menjadikannya salah satu teknik yang paling banyak digunakan untuk menyelesaikan beragam persoalan.

Binary search hanya berfungsi pada kumpulan elemen yang runtut. Dengan kata lain, agar dapat menggunakan binary search, kumpulan elemen tersebut harus diurutkan terlebih dahulu.

Saat binary search digunakan untuk melakukan operasi pada kumpulan yang telah diurutkan, jumlah iterasi selalu dapat dikurangi tergantung dengan nilai yang dicari.


C:\Windows\system32\cmd.exe
E:\Java>javac BinarySearch.java
E:\Java>java BinarySearch
Enter number of elements
5
Enter 5 integers
2
5
6
8
9
Enter value to find
5
5 found at location 2.
E:\Java>

```java
import java.util.Scanner;

class BinarySearch
{
    public static void main(String args[])
    {
        int c, first, last, middle, n, search, array[];     Scanner in = new Scanner(System.in);

        System.out.println("Enter number of elements");

        n = in.nextInt();
        array = new int[n];

        System.out.println("Enter " + n + " integers");
        for (c = 0; c < n; c++)
        array[c] = in.nextInt();

        System.out.println("Enter value to find");
        search = in.nextInt();
        first  = 0;
        last   = n - 1;
        middle = (first + last)/2;
        while( first <= last )
        {
            if ( array[middle] < search )
                first = middle + 1;
            else if ( array[middle] == search )
            {
                System.out.println(search + " found at location " + (middle + 1) + ".");
                break;
            }
            else
                last = middle - 1;
                middle = (first + last)/2;
        }
        if ( first > last )
            System.out.println(search + " is not present in the list.\n");
    }
}
```

HACKTIV8

Case 2

key < 30

	bawah							tengah							atas
index	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]
list	5	8	12	15	17	23	26	30	34	38	42	54	64	78	81

key > 15

	bawah			tengah				atas
index	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
list	5	8	12	15	17	23	26	30

key < 23

	bawah		tengah		atas
index	[3]	[4]	[5]	[6]	[7]
list	15	17	23	26	30

key == 26

	bawah	tengah	atas
index	[5]	[6]	[7]
list	23	26	30

Pada contoh di atas anda lihat bahwa target atau key yang dicari adalah nilai 26, dan data sudah diurutkan terlebih dahulu. Binary bekerja dengan memberikan nilai bawah, nilai tengah dan nilai atas dari array tersebut, kemudian setelah setiap perbandingan, porsi pencarian terus berkurang setengahnya, sampai dengan nilai key ditemukan.

Jika jumlah elemen array adalah z, maka setiap setelah perbandingan, akan menjadi z/2 elemen tersisa yang akan digunakan dalam pencarian, kemudian setelah perbandingan kedua, akan menjadi (z/2)/2 elemen tersisa yang digunakan dalam pencarian. Maka setelah y kali perbandingan , maka akan tersisa z/2y.

```
public class Binary {
    public static int pencarianBinary(int[] list, int key) {
     int bawah = 0; int atas = list.length - 1;

     while (atas >= bawah) {
       int tengah = (bawah + atas) / 2;

       if (key < list[tengah])
           atas = tengah - 1;
       else
           if (key == list[tengah])
         return tengah;

       else
         bawah = tengah + 1;
     }
       return -1; // tidak ditemukan

    }

    public static void main(String args []){
       int myArray [] = {5, 8, 12, 15, 17, 23, 26, 30, 34, 38, 42, 54, 64, 78, 81};
       int key1 = 26; int key2 = 78;
       int key3 = 8; int key4 = 39;
       int i = Binary.pencarianBinary(myArray, key1);
       int j = Binary.pencarianBinary(myArray, key2);
       int k = Binary.pencarianBinary(myArray, key3);
       int l = Binary.pencarianBinary(myArray, key4);

         System.out.println("Key " + key1 + " index " + i );
         System.out.println("Key " + key2 + " index " + j );
         System.out.println("Key " + key3 + " index " + k );
         System.out.println("Key " + key4 + " index " + l );
    }
}
```

Output:

Key 26 index 6

Key 78 index 13

Key 8 index 1

Key 39 index -1

Nilai 39 tidak terdapat pada myArray, oleh karena itu mengembalikan nilai -1.