Jobsheet 4

Praktikum Algoritma & Struktur Data

Rafi Ody Prasetyo
(2341720180)

D-IV Teknik Informatika
Politeknik Negeri Malang
Semester 2
2024

Percobaan 1

Code:

Faktorial21.java

```java
public class Faktorial21 {

    public int nilai;

    int faktorialBF(int n)
    {
        int fakto = 1;
        for (int i = 1; i <= n; i++) {
            fakto = fakto * i;
        }
        return fakto;
    }

    int faktorialDC(int n)
    {
        if (n == 1) {
            return 1;
        }
        else
        {
            int fakto = n * faktorialDC(n - 1);
            return fakto;
        }
    }

}
```

MainFaktorial21.java

```java
public class MainFaktorial21 {

    Run | Debug
    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        System.out.println(x:"----------------------");
        System.out.print(s:"Masukkan jumlah elemen: ");
        int iJml = sc.nextInt();

        Faktorial21[] fk = new Faktorial21[10];
        for (int i = 0; i < iJml; i++) {
            fk[i] = new Faktorial21();
            System.out.print("Masukkan nilai data ke-"+(i+1)+" : ");
            fk[i].nilai = sc.nextInt();
        }

        System.out.println(x:"Hasil - Brute Force");
        for (int i = 0; i < iJml; i++) {
            System.out.println("Hasil perhitungan faktorial menggunakan brute force adalah "
            + fk[i].faktorialBF(fk[i].nilai));
        }

        System.out.println(x:"Hasil - Divide and Conquer");
        for (int i = 0; i < iJml; i++) {
            System.out.println("Hasil perhitungan faktorial menggunakan divide and conquer adalah "
            + fk[i].faktorialDC(fk[i].nilai));
        }

        sc.close();
    }

}
```

Output:

```
----------------------
Masukkan jumlah elemen: 3
Masukkan nilai data ke-1 : 5
Masukkan nilai data ke-2 : 8
Masukkan nilai data ke-3 : 3
Hasil - Brute Force
Hasil perhitungan faktorial menggunakan brute force adalah 120
Hasil perhitungan faktorial menggunakan brute force adalah 40320
Hasil perhitungan faktorial menggunakan brute force adalah 6
Hasil - Divide and Conquer
Hasil perhitungan faktorial menggunakan divide and conquer adalah 120
Hasil perhitungan faktorial menggunakan divide and conquer adalah 40320
Hasil perhitungan faktorial menggunakan divide and conquer adalah 6
```

Github: https://github.com/rafiody16/Praktikum-Algoritma-dan-Struktur-Data--smt-2-/tree/main/Jobsheet%204/BruteForceDivideConquer

Pertanyaan

1. **Pada base line Algoritma Divide Conquer untuk melakukan pencarian nilai faktorial, jelaskan perbedaan bagian kode pada penggunaan if dan else!**

   Jawab:

   if (n == 1) { return 1; } Ini adalah kondisi dasar atau base case dari algoritma rekursif. Ketika nilai n adalah 1, maka hasil faktorialnya adalah 1. Ini diperlukan agar rekursi berhenti dan tidak terjadi rekursi tak terbatas. ***Sedangkan else { int fakto = n * faktorialDC(n - 1); return fakto; }*** Pada bagian else ini, jika kondisi n bukan 1 (artinya n lebih besar dari 1), maka dilakukan rekursi. Pada rekursi ini, nilai faktorial untuk n dihitung dengan mengalikan n dengan hasil dari pemanggilan fungsi faktorialDC untuk n - 1. Dengan demikian, algoritma akan melakukan pemanggilan rekursif hingga mencapai base case ketika n adalah 1, dan kemudian mulai mengalikan kembali hasil rekursif tersebut sehingga menghasilkan nilai faktorial yang diinginkan.

2. **Apakah memungkinkan perulangan pada method faktorialBF() diubah selain menggunakan for? Buktikan!**

   Jawab:

   For dapat diubah dengan menggunakan perulangan WHILE.

   Before:

```
5       int faktorialBF(int n)
6       {
7           int fakto = 1;
8           for (int i = 1; i <= n; i++) {
9               fakto = fakto * i;
10          }
11          return fakto;
12      }
```

   After:

```
int faktorialBF(int n)
{
    int fakto = 1;
    int i = 1;

    while (i <= n) {
        fakto = fakto * i;
        i++;
    }
    return fakto;

}
```

Output:

```
Masukkan jumlah elemen: 3
Masukkan nilai data ke-1 : 4
Masukkan nilai data ke-2 : 2
Masukkan nilai data ke-3 : 6
Hasil - Brute Force
Hasil perhitungan faktorial menggunakan brute force adalah 24
Hasil perhitungan faktorial menggunakan brute force adalah 2
Hasil perhitungan faktorial menggunakan brute force adalah 720
Hasil - Divide and Conquer
Hasil perhitungan faktorial menggunakan divide and conquer adalah 24
Hasil perhitungan faktorial menggunakan divide and conquer adalah 2
Hasil perhitungan faktorial menggunakan divide and conquer adalah 720
```

3. **Jelaskan perbedaan antara fakto *= i; dan int fakto = n * faktorialDC(n-1); !**

   Jawab:

   fakto *= i; Digunakan untuk mengalikan nilai fakto dengan nilai i dan kemudian menetapkan hasilnya kembali ke variabel fakto. Sedangkan ***int fakto = n * faktorialDC(n-1);*** digunakan untuk melakukan perhitungan nilai faktorial menggunakan metode Divide and Conquer. Di sini, nilai faktorial dari n dihitung dengan mengalikan n dengan hasil dari pemanggilan rekursif faktorialDC(n - 1). Dalam rekursi ini, nilai faktorial untuk n - 1 dihitung secara rekursif hingga mencapai kondisi dasar (base case). Setelah itu, hasilnya dikalikan dengan n dan dikembalikan.

Percobaan 2

Code:

Pangkat21.java

```java
public class Pangkat21 {

    public int nilai, pangkat;


    int pangkatBF(int a, int n)
    {
        int hasil = 1;
        for (int i = 0; i < n; i++) {
            hasil *= a;
        }
        return hasil;
    }

    int pangkatDC(int a, int n)
    {
        if (n == 0) {
            return 1;
        }
        else
        {
            if (n % 2 == 1) {
                return (pangkatDC(a, n / 2) * pangkatDC(a, n / 2) * a);
            }
            else
            {
                return (pangkatDC(a, n / 2) * pangkatDC(a, n / 2));
            }
        }
    }
}
```

MainPangkat21.java

```java
public class MainPangkat21 {
    Run | Debug
    public static void main(String[] args) {

        Scanner sc = new Scanner (System.in);

        System.out.println(x:"================================");
        System.out.print(s:"Masukkan jumlah elemen yang dihitung: ");
        int elemen = sc.nextInt();

        Pangkat21 [] png = new Pangkat21[elemen];

        for (int i = 0; i < elemen; i++) {
            png [i] = new Pangkat21();
            System.out.print(s:"Masukkan nilai yang ingin dipangkatkan: ");
            int nilai = sc.nextInt();
            System.out.print(s:"Masukkan nilai pangkat: ");
            int pangkat = sc.nextInt();
            png[i].nilai = nilai;
            png[i].pangkat = pangkat;
            sc.nextLine();
        }

        System.out.println(x:"Hasil Pangkat - BRUTE FORCE");
        for (int i = 0; i < elemen; i++) {
            System.out.println("Hasil dari "
            + png[i].nilai +" pangkat "
            + png[i].pangkat +" adalah "
            + png[i].pangkatBF(png[i].nilai, png[i].pangkat));
        }

        System.out.println(x:"Hasil Pangkat - DIVIDE AND CONQUER");
        for (int i = 0; i < elemen; i++) {
            System.out.println("Hasil dari "
            + png[i].nilai +" pangkat "
            + png[i].pangkat +" adalah "
            + png[i].pangkatDC(png[i].nilai, png[i].pangkat));
        }

        sc.close();

    }

}
```

Output:

```
================================
Masukkan jumlah elemen yang dihitung: 2
Masukkan nilai yang ingin dipangkatkan: 4
Masukkan nilai pangkat: 2
Masukkan nilai yang ingin dipangkatkan: 3
Masukkan nilai pangkat: 5
Hasil Pangkat - BRUTE FORCE
Hasil dari 4 pangkat 2 adalah 16
Hasil dari 3 pangkat 5 adalah 243
Hasil Pangkat - DIVIDE AND CONQUER
Hasil dari 4 pangkat 2 adalah 16
Hasil dari 3 pangkat 5 adalah 243
```

Pertanyaan

1. **Jelaskan mengenai perbedaan 2 method yang dibuat yaitu PangkatBF() dan PangkatDC()!**

   Jawab:

   Method PangkatBF() menggunakan loop *for* untuk mengalikan bilangan dasar sebanyak kali sesuai dengan pangkat yang diinginkan. Pada setiap iterasi, bilangan dasar akan dikalikan dengan dirinya sendiri sebanyak n kali. Sedangkan method PangkatDC() bekerja dengan cara membagi masalah menjadi dua bagian yang lebih kecil dan menyelesaikan setiap bagian secara terpisah, kemudian menggabungkan hasilnya. Jika pangkat (n) adalah bilangan genap, maka a^n = (a^(n/2)) * (a^(n/2)). Jika pangkat (n) adalah bilangan ganjil, maka a^n = a * (a^((n-1)/2)) * (a^((n-1)/2)).

2. **Apakah tahap combine sudah termasuk dalam kode tersebut?Tunjukkan!**

   Jawab:

   Tahap combine dalam algoritma divide-and-conquer (bagi dan taklukkan) menggabungkan solusi dari sub-masalah untuk menghasilkan solusi akhir. Pada kode pangkatDC, tahap combine terjadi di bagian else.

```
if (n % 2 == 1) {
    return (pangkatDC(a, n / 2) * pangkatDC(a, n / 2) * a);
}
else
{
    return (pangkatDC(a, n / 2) * pangkatDC(a, n / 2));
}
```

3. **Modifikasi kode program tersebut, anggap proses pengisian atribut dilakukan dengan konstruktor.**

4. **Tambahkan menu agar salah satu method yang terpilih saja yang akan dijalankan menggunakan switch-case!**

   Jawab:

   Before:

```
public class Pangkat21 {

    public int nilai, pangkat;


    int pangkatBF(int a, int n)
    {
        int hasil = 1;
        for (int i = 0; i < n; i++) {
            hasil *= a;
        }
        return hasil;
    }

```

After:

```
public class Pangkat21 {

    public int nilai, pangkat;

    public Pangkat21 (int nilai, int pangkat)
    {
        this.nilai = nilai;
        this.pangkat = pangkat;
    }

```

Before:

```
for (int i = 0; i < elemen; i++) {
    png [i] = new Pangkat21();
    System.out.print(s:"Masukkan nilai yang ingin dipangkatkan: ");
    int nilai = sc.nextInt();
    System.out.print(s:"Masukkan nilai pangkat: ");
    int pangkat = sc.nextInt();
    png[i].nilai = nilai;
    png[i].pangkat = pangkat;
    System.out.println();
    sc.nextLine();
}
```

After:

```
for (int i = 0; i < elemen; i++) {
    System.out.print(s:"Masukkan nilai yang ingin dipangkatkan: ");
    int nilai = sc.nextInt();
    System.out.print(s:"Masukkan nilai pangkat: ");
    int pangkat = sc.nextInt();
    png [i] = new Pangkat21(nilai, pangkat);
    System.out.println();
    sc.nextLine();
}
```

Before

```
System.out.println(x:"Hasil Pangkat - BRUTE FORCE");
for (int i = 0; i < elemen; i++) {
    System.out.println("Hasil dari "
    + png[i].nilai +" pangkat "
    + png[i].pangkat +" adalah "
    + png[i].pangkatBF(png[i].nilai, png[i].pangkat));
}

System.out.println(x:"Hasil Pangkat - DIVIDE AND CONQUER");
for (int i = 0; i < elemen; i++) {
    System.out.println("Hasil dari "
    + png[i].nilai +" pangkat "
    + png[i].pangkat +" adalah "
    + png[i].pangkatDC(png[i].nilai, png[i].pangkat));
}
```

After

```
System.out.println(x:"Main Menu");
System.out.println(x:"1. Hasil Pangkat BF\n2. Hasil Pangkat DC");
System.out.print(s:"Pilih: ");
int pilih = sc.nextInt();

switch (pilih) {
    case 1:
        System.out.println(x:"Hasil Pangkat - BRUTE FORCE");
        for (int i = 0; i < elemen; i++) {
            System.out.println("Hasil dari "
            + png[i].nilai +" pangkat "
            + png[i].pangkat +" adalah "
            + png[i].pangkatBF(png[i].nilai, png[i].pangkat));
        }
        break;

    case 2:
        System.out.println(x:"Hasil Pangkat - DIVIDE AND CONQUER");
        for (int i = 0; i < elemen; i++) {
            System.out.println("Hasil dari "
            + png[i].nilai +" pangkat "
            + png[i].pangkat +" adalah "
            + png[i].pangkatDC(png[i].nilai, png[i].pangkat));
        }
        break;

    default:
        break;
}
```

Output:

```
================================
Masukkan jumlah elemen yang dihitung: 2
Masukkan nilai yang ingin dipangkatkan: 2
Masukkan nilai pangkat: 4

Masukkan nilai yang ingin dipangkatkan: 3
Masukkan nilai pangkat: 5

Main Menu
1. Hasil Pangkat BF
2. Hasil Pangkat DC
Pilih: 2
Hasil Pangkat - DIVIDE AND CONQUER
Hasil dari 2 pangkat 4 adalah 16
Hasil dari 3 pangkat 5 adalah 243
```

```
================================
Masukkan jumlah elemen yang dihitung: 2
Masukkan nilai yang ingin dipangkatkan: 3
Masukkan nilai pangkat: 3

Masukkan nilai yang ingin dipangkatkan: 4
Masukkan nilai pangkat: 6

Main Menu
1. Hasil Pangkat BF
2. Hasil Pangkat DC
Pilih: 1
Hasil Pangkat - BRUTE FORCE
Hasil dari 3 pangkat 3 adalah 27
Hasil dari 4 pangkat 6 adalah 4096
```

Percobaan 3

Code:

Sum21.java

```java
public class Sum21 {
    public int elemen;
    public double keuntungan[], total;

    public Sum21(int elemen)
    {
        this.elemen = elemen;
        this.keuntungan = new double[elemen];
        this.total = 0;
    }

    public double totalBF(double arr[])
    {
        for (int i = 0; i < elemen; i++) {
            total = total + arr[i];
        }
        return total;
    }

    public double totalDC(double arr[], int l, int r)
    {
        if (l == r) {
            return arr[l];
        }
        else if (l < r)
        {
            int mid = (l + r) / 2;
            double lsum = totalDC(arr, l, mid - 1);
            double rsum = totalDC(arr, mid + 1, r);
            return lsum + rsum + arr[mid];
        }
        return 0;
    }
```

MainSum21.java

```java
import java.util.Scanner;

public class MainSum21 {

    Run | Debug
    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println(x:"============================");
        System.out.println(x:"Program Menghitung Keuntungan total (Satuan Juta. Misal 5.9)");
        System.out.print(s:"Masukkan Jumlah Bulan: ");
        int elm = sc.nextInt();

        Sum21 sm = new Sum21(elm);
        System.out.println(x:"============================");
        for (int i = 0; i < sm.elemen; i++) {
            System.out.print("Masukkan untung bulan ke - "+ (i+1) +" : ");
            sm.keuntungan[i] = sc.nextDouble();
        }

        System.out.println(x:"============================");
        System.out.println(x:"Algoritma Brute Force");
        System.out.println("Total keuntungan perusahaan selama "+ sm.elemen +" bulan adalah "+ sm.totalBF(sm.keuntungan));
        System.out.println(x:"============================");
        System.out.println(x:"Algoritma Divide Conquer");
        System.out.println("Total keuntungan perusahaan selama "+ sm.elemen +" bulan adalah "+ sm.totalDC(sm.keuntungan, l:0, sm.elemen-1));

        sc.close();
    }

}
```

Output:

```
============================
Program Menghitung Keuntungan total (Satuan Juta. Misal 5.9)
Masukkan Jumlah Bulan: 5
============================
Masukkan untung bulan ke - 1 : 8.5
Masukkan untung bulan ke - 2 : 9.54
Masukkan untung bulan ke - 3 : 7.2
Masukkan untung bulan ke - 4 : 9.1
Masukkan untung bulan ke - 5 : 6
============================
Algoritma Brute Force
Total keuntungan perusahaan selama 5 bulan adalah 40.339999999999996
============================
Algoritma Divide Conquer
Total keuntungan perusahaan selama 5 bulan adalah 40.34
```

Github: https://github.com/rafiody16/Praktikum-Algoritma-dan-Struktur-Data--smt-2-/tree/main/Jobsheet%204/BruteForceDivideConquer

Pertanyaan

1. **Mengapa terdapat formulasi return value berikut? Jelaskan!**

```
return lsum+rsum+arr[mid];
```

Jawab:

Formulasi return lsum + rsum + arr[mid]; digunakan untuk menggabungkan hasil dari submasalah yang lebih kecil menjadi hasil akhir dari pemecahan masalah yang lebih besar dalam pendekatan Divide and Conquer.

2. **Kenapa dibutuhkan variable mid pada method TotalDC()!**

Jawab:

Variabel mid pada metode totalDC() dibutuhkan untuk menentukan titik tengah dari rentang array yang sedang diproses. Pada pendekatan Divide and Conquer, array dibagi menjadi dua bagian yang lebih kecil untuk memecahkan masalah menjadi submasalah yang lebih sederhana.

3. **Program perhitungan keuntungan suatu perusahaan ini hanya untuk satu perusahaan saja. Bagaimana cara menghitung sekaligus keuntungan beberapa bulan untuk beberapa perusahaan.(Setiap perusahaan bisa saja memiliki jumlah bulan berbeda-beda)? Buktikan dengan program!**

Jawab:

Code:

```
        System.out.println(x:"==============================");
        System.out.println(x:"Program Menghitung Keuntungan total (Satuan Juta. Misal 5.9)");
        System.out.print(s:"Masukkan jumlah perusahaan yang akan dihitung: ");
        int prs = sc.nextInt();
        Sum21[] arr = new Sum21[prs];

        for (int x = 0; x < prs; x++) {
            System.out.println("Perhitungan keuntungan perusahaan ke - "+ (x + 1));
            System.out.print(s:"Masukkan Jumlah Bulan: ");
            int elm = sc.nextInt();
            arr[x] = new Sum21(elm);

            System.out.println(x:"==============================");
            for (int i = 0; i < arr[x].elemen; i++) {
                System.out.print("Masukkan untung bulan ke - "+ (i+1) +" : ");
                arr[x].keuntungan[i] = sc.nextDouble();
            }

        }

        System.out.println();
```

```
        for (int y = 0; y < arr.length; y++) {
            System.out.println("Perhitungan Perusahaan "+ (y + 1));
            System.out.println(x:"==============================");
            System.out.println(x:"Algoritma Brute Force");
            System.out.println("Total keuntungan perusahaan selama "+ arr[y].elemen +" bulan adalah "
            + arr[y].totalBF(arr[y].keuntungan));
            System.out.println(x:"==============================");
            System.out.println(x:"Algoritma Divide Conquer");
            System.out.println("Total keuntungan perusahaan selama "+ arr[y].elemen +" bulan adalah "
            + arr[y].totalDC(arr[y].keuntungan, l:0, arr[y].elemen-1));
        }

        sc.close();
    }

}
```

Output:

```
==============================
Program Menghitung Keuntungan total (Satuan Juta. Misal 5.9)
Masukkan jumlah perusahaan yang akan dihitung: 2
Perhitungan keuntungan perusahaan ke - 1
Masukkan Jumlah Bulan: 2
==============================
Masukkan untung bulan ke - 1 : 2.1
Masukkan untung bulan ke - 2 : 3.3
Perhitungan keuntungan perusahaan ke - 2
Masukkan Jumlah Bulan: 3
==============================
Masukkan untung bulan ke - 1 : 3.3
Masukkan untung bulan ke - 2 : 4.9
Masukkan untung bulan ke - 3 : 5

Perhitungan Perusahaan 1
==============================
Algoritma Brute Force
Total keuntungan perusahaan selama 2 bulan adalah 5.4
==============================
Algoritma Divide Conquer
Total keuntungan perusahaan selama 2 bulan adalah 5.4
Perhitungan Perusahaan 2
==============================
Algoritma Brute Force
Total keuntungan perusahaan selama 3 bulan adalah 13.2
==============================
Algoritma Divide Conquer
Total keuntungan perusahaan selama 3 bulan adalah 13.200000000000001
```

Latihan Praktikum

1. **Sebuah showroom memiliki daftar mobil dengan data sesuai tabel di bawah ini**

merk	tipe	tahun	top_acceleration	top_power
BMW	M2 Coupe	2016	6816	728
Ford	Fiesta ST	2014	3921	575
Nissan	370Z	2009	4360	657
Subaru	BRZ	2014	4058	609
Subaru	Impreza WRX STI	2013	6255	703
Toyota	AE86 Trueno	1986	3700	553
Toyota	86/GT86	2014	4180	609
Volkswagen	Golf GTI	2014	4180	631

Tentukan:

A. top_acceleration tertinggi menggunakan Divide and Conquer!

B. top_acceleration terendah menggunakan Divide and Conquer!

C. Rata-rata top_power dari seluruh mobil menggunakan Brute Force!

Jawab:

Code:

Showroom21.java

```java
public class Showroom21 {

    public int low, max;

    public int upTopAcc(int arrAcc[], int l, int r)
    {
        if(l == r)
        {
            return arrAcc[l];
        }
        else if (r == l - 1)
        {
            if (arrAcc[r] > arrAcc[l])
            {
                return arrAcc[r];
            }
            else
            {
                return arrAcc[l];
            }
        }
        else{
            int mid = (l + r) / 2;
            int lsum = upTopAcc(arrAcc, l, mid - 1);
            int rsum = upTopAcc(arrAcc, mid + 1, r);
```

```
            if (lsum > rsum)
            {
                return lsum;
            }
            else
            {
                return rsum;
            }
        }
    }

    public int minTopAcc(int arrAcc[], int l, int r)
    {
        if(l == r)
        {
            return arrAcc[l];
        }
        else if (r == l - 1)
        {
            if (arrAcc[r] < arrAcc[l])
            {
                return arrAcc[r];
            }
            else
            {
                return arrAcc[l];
            }
        }
```

```
        else{
            int mid = (l + r) / 2;
            int lsum = minTopAcc(arrAcc, l, mid);
            int rsum = minTopAcc(arrAcc, mid + 1, r);

            if (lsum < rsum)
            {
                return lsum;
            }
            else
            {
                return rsum;
            }
        }
    }

    public double avgPower (int arrPwr[])
    {
        int sum = 0;
        int n = arrPwr.length;

        for (int i = 0; i < n; i++) {
            sum += arrPwr[i];
        }

        return (double) sum / n;
    }
}
```

MainShowroom21.java

```java
public class MainShowroom21 {

    Run | Debug
    public static void main(String[] args) {
        int[] arrAcc = { 6816, 3921, 4360, 4058, 6255, 3700, 4180, 4180 };
        int[] arrPwr = { 728, 575, 657, 609, 703, 553, 609, 631 };

        System.out.println("Merk                    Tipe    Top Acceleration   Top Power\n" +
        "BMW M2 Coupe            2016    6816               728\n" +
        "Ford Fiesta ST          2014    3921               575\n" +
        "Nissan 370Z             2009    4360               657\n" +
        "Subaru BRZ              2014    4058               609\n" +
        "Subaru Impreza WRX STI  2013    6255               703\n" +
        "Toyota AE86 Trueno      1986    3700               553\n" +
        "Toyota 86/GT86          2014    4180               609\n" +
        "Volkswagen Golf GTI     2014    4180               631\n");

        Showroom21 rslt = new Showroom21();

        int maxAcc = rslt.upTopAcc(arrAcc, 1:0, arrAcc.length - 1);
        System.out.println("Nilai Top Acceleration Terbesar: "+ maxAcc);
        int minAcc = rslt.minTopAcc(arrAcc, 1:0, arrAcc.length - 1);
        System.out.println("Nilai Top Acceleration Terkecil: "+ minAcc);
        double avgPower = rslt.avgPower(arrPwr);
        System.out.println("Rata2 Top Power: "+avgPower);
    }

}
```

Output:

```
Merk                    Tipe    Top Acceleration   Top Power
BMW M2 Coupe            2016    6816               728
Ford Fiesta ST          2014    3921               575
Nissan 370Z             2009    4360               657
Subaru BRZ              2014    4058               609
Subaru Impreza WRX STI  2013    6255               703
Toyota AE86 Trueno      1986    3700               553
Toyota 86/GT86          2014    4180               609
Volkswagen Golf GTI     2014    4180               631

Nilai Top Acceleration Terbesar: 6816
Nilai Top Acceleration Terkecil: 3700
Rata2 Top Power: 633.125
```

Github: https://github.com/rafiody16/Praktikum-Algoritma-dan-Struktur-Data--smt-2-/tree/main/Jobsheet%204/Latihan