JOBSHEET IV
BRUTE FORCE DAN DIVIDE CONQUER

Nama: Rizqi Bagus Andrean
Kelas: TI-1D
Absen: 25

4.1 Tujuan Praktikum

Setelah melakukan materi praktikum ini, mahasiswa mampu:

1. Mahasiswa mampu membuat algoritma bruteforce dan divide-conquer
2. Mahasiswa mampu menerapkan penggunaan algorima bruteforce dan divide-conquer

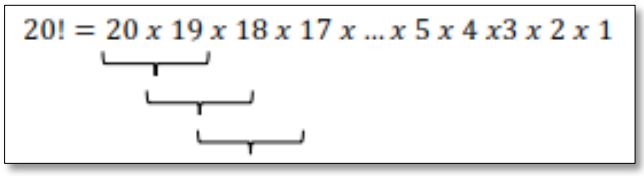
4.2 Menghitung Nilai Faktorial dengan Algoritma Brute Force dan Divide and Conquer

Perhatikan Diagram Class berikut ini :

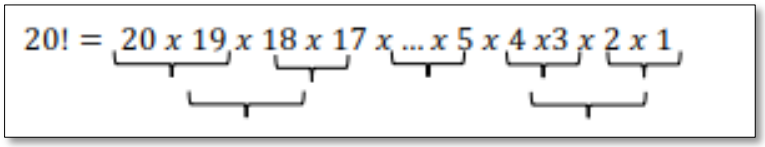
Faktorial
nilai: int
faktorialBF(): int faktorialDC(): int

Berdasarkan diagram class di atas, akan dibuat program class dalam Java. Untuk menghitung nilai faktorial suatu angka menggunakan 2 jenis algoritma, Brute Force dan Divide and Conquer. Jika digambarkan terdapat perbedaan proses perhitungan 2 jenis algoritma tersebut sebagai berikut :

Tahapan pencarian nilai faktorial dengan algoritma Brute Force :



Tahapan pencarian nilai faktorial dengan algoritma Divide and Conquer :



4.2.1 Langkah-langkah Percobaan

1. Buat Project baru, dengan nama "**`BruteForceDivideConquer`**". Buat package dengan nama minggu5.
2. Buatlah class baru dengan nama **`Faktorial`**

3. Lengkapi class **`Faktorial`** dengan atribut dan method yang telah digambarkan di dalam diagram class di atas, sebagai berikut:
   a) Tambahkan atribut nilai

```
    public int nilai;
```

b) Tambahkan method `faktorialBF()` nilai

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

c) Tambahkan method `faktorialDC()` nilai

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

4. Coba jalankan (Run) class `Faktorial` dengan membuat class baru `MainFaktorial`.

a) Di dalam fungsi main sediakan komunikasi dengan user untuk menginputkan jumlah angka yang akan dicari nilai faktorialnya

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

b) Buat Array of Objek pada fungsi main, kemudian inputkan beberapa nilai yang akan dihitung faktorialnya

```
Faktorial[] fk = new Faktorial[10];
for(int i=0; i < iJml; i++){
    fk[i] = new Faktorial();
    System.out.println("masukkan nilai data ke-" +(i+1)+":");
    int iNilai = sc.nextInt();
}
```

c) Tampilkan hasil pemanggilan method `faktorialDC()` dan `faktorialBF()`

```
System.out.println("HASIL - BRUTE FORCE");
for(int i=0; i < iJml;i++){
    System.out.println
("Hasil penghitungan faktorial menggunakan Brute Force adalah "
            + fk[i].faktorialBF(fk[i].nilai));
}
System.out.println("HASIL - DIVIDE AND CONQUER");
for(int i=0; i < iJml;i++){
    System.out.println
("Hasil penghitungan faktorial menggunakan Divide and Conquer adalah "
            + fk[i].faktorialDC(fk[i].nilai));
}
```

d) Pastikan program sudah berjalan dengan baik!

```java
import java.util.Scanner;

class Faktorial{

    public int nilai, pangkat;

    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;
        }
    }

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

    int pangkatDC(int a, int n) {
        if (n==1) {
            return a;
        }else{

            if (n%2==0) {
            return pangkatDC(a, n/2) * pangkatDC(a, n/2) * a;

            }else{
                return pangkatDC(a, n/2) * pangkatDC(a, n/2);
            }

        }
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.println("---------");
        System.out.print("Masukkan jml: ");
        int iJml = sc.nextInt();

        Faktorial[] fk = new Faktorial[10];
        for(int i=0; i<iJml; i++) {
            fk[i] = new Faktorial();
            System.out.print("masukkan nilai: ");
            int iNilai = sc.nextInt();
            fk[i].nilai = iNilai;
        }

        for(int i=0; i<iJml; i++) {
            System.out.println("Hasil BF: "  + fk[i].faktorialBF(fk[i].nilai));
        }

        for(int i=0; i<iJml; i++) {
            System.out.println("Hasil DC: "  + fk[i].faktorialDC(fk[i].nilai));
        }
    }

}
```

4.2.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.

```
-----------------
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 penghitungan faktorial menggunakan Brute Force adalah 120
Hasil penghitungan faktorial menggunakan Brute Force adalah 40320
Hasil penghitungan faktorial menggunakan Brute Force adalah 6
HASIL - DIVIDE AND CONQUER
Hasil penghitungan faktorial menggunakan Divide and Conquer adalah 120
Hasil penghitungan faktorial menggunakan Divide and Conquer adalah 40320
Hasil penghitungan faktorial menggunakan Divide and Conquer adalah 6
-------------------------------------------------------------------------
```

```
Masukkan jml: 1
masukkan nilai: 4
Hasil BF: 24
Hasil DC: 24
PS C:\Users\Acer\Tugas Kuliah\Seme
```

4.2.3 Pertanyaan

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

   Pada if else, membaca nilai n jika n == 1 maka akan mereturn 1 atau mereturn hasil faktorial dan mengehentikan fungsi rekursive nya. Apabila else maka dia akan mengkalikan n dengan hasil dari panggilan fungsi dirinya sendiri dengan parameter n yang dikurangi satu.

2. Apakah memungkinkan perulangan pada method `faktorialBF()` dirubah selain menggunakan for?Buktikan! bisa

```
int faktorialBFwhile(int n) {
    int i = 1;
    int fakto = 1;
    while (i <= n) {
      fakto *= i;
      i++;
    }
    return fakto;
}
```

3. Jelaskan perbedaan antara **fakto *= i;** dan **int fakto = n * faktorialDC(n-1);** !
   Kedua baris kode memiliki fungsi yang berbeda dalam menghitung faktorial.
   Fakto *=i digunakan dalam algoritma bruteforce untuk memperbarui fakto secara langsung.
   Sedangkan, int fakto = n * faktorialDC(n-1) diigunakan dalam alagoritma divide conqueror untuk menghitung nilai faktor baru dengan fungsi rekurss

4.3 Menghitung Hasil Pangkat dengan Algoritma Brute Force dan Divide and Conquer

Pada praktikum ini kita akan membuat program class dalam Java. Untuk menghitung nilai pangkat suatu angka menggunakan 2 jenis algoritma, Brute Force dan Divide and Conquer.

4.3.1 Langkah-langkah Percobaan

1. Di dalam paket **`minggu5`**, buatlah class baru dengan nama **`Pangkat`**. Dan di dalam class **`Pangkat`** tersebut, buat atribut angka yang akan dipangkatkan sekaligus dengan angka pemangkatnya

```
    public int nilai,pangkat;
```

2. Pada class Pangkat tersebut, tambahkan method `PangkatBF()`

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

3. Pada class Pangkat juga tambahkan method `PangkatDC()`

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

4. Perhatikan apakah sudah tidak ada kesalahan yang muncul dalam pembuatan class Pangkat
5. Selanjutnya buat class baru yang di dalamnya terdapat method main. Class tersebut dapat dinamakan `MainPangkat`. Tambahkan kode pada class main untuk menginputkan jumlah nilai yang akan dihitung pangkatnya.

```
Scanner sc = new Scanner(System.in);
System.out.println("=====================================");
System.out.println("Masukkan jumlah elemen yang dihtung: ");
int elemen = sc.nextInt();
```

6. Nilai pada tahap 5 selanjutnya digunakan untuk instansiasi array of objek. Di dalam Kode berikut ditambahkan proses pengisian beberapa nilai yang akan dipangkatkan sekaligus dengan pemangkatnya.

```
Pangkat[] png = new Pangkat[element];
for(int i=0; i < elemen; i++){
    png[i] = new Pangkat();
    System.out.println("Masukkan nilai yang hendak dipangkatkan: ");
    int nilai = sc.nextInt();
    System.out.println("Masukkan nilai pemangkat: ");
    int pangkat = sc.nextInt();
}
```

7. Kemudian, panggil hasil nya dengan mengeluarkan return value dari method `PangkatBF()` dan `PangkatDC()`.

```
System.out.println("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("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));
```

4.3.2 Verifikasi Hasil Percobaan

Pastikan output yang ditampilkan sudah benar seperti di bawah ini.

```
=====================================
Masukkan jumlah elemen yang dihtung:
2
Masukkan nilai yang hendak dipangkatkan:
6
Masukkan nilai pemangkat:
2
Masukkan nilai yang hendak dipangkatkan:
4
Masukkan nilai pemangkat:
3
HASIL PANGKAT - BRUTE FORCE
Hasil dari 6 pangkat 2 adalah 36
Hasil dari 4 pangkat 3 adalah 64
HASIL PANGKAT - DIVIDE AND CONQUER
Hasil dari 6 pangkat 2 adalah 36
Hasil dari 4 pangkat 3 adalah 64
```

```java
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);
    }
  }
```

```
ccf9fa7b25\redhat.java\jdt_ws\jobs
Hasil BF: 8
Hasil DC: 8
PS C:\Users\Acer\Tugas Kuliah\Seme
```

4.3.3 Pertanyaan

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

   PangkatBF menggunakan looping untuk menghitung, tapi tidak cocok untuk nilai besar.

   PangkatDC menggunakan fungsi rekursive dan cocok untuk nilai yang berpangkat besar.

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

   Tahap combine dalam PangkatDC() terjadi ketika hasil rekursi dari dua sub-masalah dikalikan untuk menghasilkan pangkat akhir.

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

```
int a = 2;
int n = 3;
Faktorial fk = new Faktorial();

System.out.println(x:"--------- Hitung Pangkat ---------");
System.out.println(x:"1. Brute Force");
System.out.println(x:"2. Divide Conqueror");

System.out.print(s:"Masukkan pilihan: ");
int pilihan = sc.nextInt();

switch (pilihan) {
    case 1:
    System.out.println("Hasil BF: "  + fk.pangkatBF(a, n));
        break;
    default:
    System.out.println("Hasil DC: "  + fk.pangkatDC(a, n));
        break;
}
```

```
--------- Hitung Pangkat ---------
1. Brute Force
2. Divide Conqueror
Masukkan pilihan: 1
Hasil BF: 8
PS C:\Users\Acer\Tugas Kuliah\Semester 2\Praktek Algoritma\jobsheet 5>
```

4.4 Menghitung Sum Array dengan Algoritma Brute Force dan Divide and Conquer

Di dalam percobaa1n ini, kita akan mempraktekkan bagaimana proses *divide, conquer*, dan

combine diterapkan pada studi kasus penjumlahan keuntungan suatu perusahaan dalam beberapa bulan.

4.4.1 Langkah-langkah Percobaan

1. Pada paket minggu5. Buat class baru yaitu class `Sum`. DI salam class tersebut terdapat beberapa atribut jumlah elemen array, array, dan juga total. Tambahkan pula konstruktor pada class `Sum`.

```
int elemen;
double keuntungan[], total;


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

2. Tambahkan method `TotalBF()` yang akan menghitung total nilai array dengan cara *iterative*.

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

    }
    return total;
}
```

3. Tambahkan pula method `TotalDC()` untuk implementasi perhitungan nilai total array menggunakan algoritma Divide and Conquer

```
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-l);
        double rsum = totalDC(arr, mid+l, r);
        return lsum+rsum+arr[mid];
    }
    return 0;
}
```

4. Buat class baru yaitu `MainSum`. Di dalam kelas ini terdapat method main. Pada method ini user dapat menuliskan berapa bulan keuntungan yang akan dihitung. Dalam kelas ini sekaligus dibuat instansiasi objek untuk memanggil atribut ataupun fungsi pada class `Sum`

```
Scanner sc = new Scanner(System.in);
System.out.println("================================================================");
System.out.println("Program Menghitung Keuntungan Total (Satuan Juta. Misal 5.9)");
System.out.print("Masukkan jumlah bulan : ");
int elm = sc.nextInt();
```

5. Karena yang akan dihitung adalah total nilai keuntungan, maka ditambahkan pula pada method main mana array yang akan dihitung. Array tersebut merupakan atribut yang terdapat di class `Sum`, maka dari itu dibutuhkan pembuatan objek Sum terlebih dahulu.

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

6. Tampilkan hasil perhitungan melalui objek yang telah dibuat untuk kedua cara yang ada (Brute Force dan Divide and Conquer)

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

4.4.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.

```
run:
================================================================
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
BUILD SUCCESSFUL (total time: 11 seconds)
```

```java
import java.util.Scanner;

public class Sum {
    int elemen;
    double keuntungan[], total;

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

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

    double totalDC(double arr[], int l, int r) {
        if (l == r) {
          return arr[l];  // Base case: single element sub-array
        } else {
          int mid = (l + r) / 2;  // Correct integer division for mid index
          double lsum = totalDC(arr, l, mid);  // Recursive call for left sub-array
          double rsum = totalDC(arr, mid + 1, r);  // Recursive call for right sub-array (adjusted starting index)
          return lsum + rsum;  // Return the sum of left and right sub-arrays
        }
      }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("---------");
        System.out.print("Masukkan jml bulan: ");
        int elm = sc.nextInt();

        Sum sm = new Sum(elm);
        System.out.println("==================");

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

        System.out.println("Total keuntungan BF: "+sm.totalBF(sm.keuntungan));
        System.out.println("Total keuntungan DC: "+sm.totalDC(sm.keuntungan, 0, sm.elemen-1));
    }
}
```

```
7\bin' 'Sum' cer\Tugas Kuliah\Semester 2\Praktek Algoritm
---------
Masukkan jml bulan: 5
==================
masukkan keuntungan bulan ke-1: 8.5
masukkan keuntungan bulan ke-2: 9.54
masukkan keuntungan bulan ke-3: 7.2
masukkan keuntungan bulan ke-4: 9.1
masukkan keuntungan bulan ke-5: 6
Total keuntungan BF: 40.339999999999996
Total keuntungan DC: 40.339999999999996
```

4.4.3 Pertanyaan

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

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

Formulasi `return lsum + rsum` pada kode `totalDC` digunakan untuk menghitung total nilai elemen dalam sebuah array dengan metode divide and conquer. Formulasi ini menggabungkan hasil dari sub-masalah kiri dan kanan dengan menambahkan nilai `lsum` dan `rsum`.

2. Kenapa dibutuhkan variable `mid` pada method `TotalDC()`?

   `Mid` membantu memecah masalah penjumlahan array menjadi sub-masalah yang lebih kecil (kiri & kanan) pada pemanggilan rekursif, sehingga perhitungan lebih efisien. Tanpa `mid`, pembagian sub-array dan perhitungan rekursif jadi rumit, berdampak pada performa fungsi.

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!

```
Run | Debug
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);    Resource leak: 'sc' is never closed


    System.out.print(s:"Masukkan jumlah perusahaan: ");
    int jmlP = sc.nextInt();

    for (int i=0; i<jmlP; i++) {
    System.out.println("Perusahaan ke-"+(i+1));
    System.out.print(s:"Masukkan jumlah bulan: ");
    int elm = sc.nextInt();
    Sum sm = new Sum(elm);
    System.out.println(x:"==================");

    for (int j=0; j<sm.elemen; j++) {
        System.out.print("masukkan keuntungan bulan ke-"+(j+1)+": ");
        sm.keuntungan[j] = sc.nextDouble();
    }

    System.out.println("Total keuntungan BF: "+sm.totalBF(sm.keuntungan));
    System.out.println("Total keuntungan DC: "+sm.totalDC(sm.keuntungan, l:0, sm.elemen-1));

    System.out.println(x:"==================");
    }
```

```
 'C:\Users\Acer\AppData\Roaming\Code\User\workspaceStorage\82a328f
Masukkan jumlah perusahaan: 2
Perusahaan ke-1
Masukkan jumlah bulan: 2
==================
masukkan keuntungan bulan ke-1: 1
masukkan keuntungan bulan ke-2: 2
Total keuntungan BF: 3.0
Total keuntungan DC: 3.0
==================
Perusahaan ke-2
Masukkan jumlah bulan: 3
==================
masukkan keuntungan bulan ke-1: 1
masukkan keuntungan bulan ke-2: 2
masukkan keuntungan bulan ke-3: 3
Total keuntungan BF: 6.0
Total keuntungan DC: 6.0
```

4.5 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!

```java
package showroom;

public class Mobil {
    String merk, tipe;
    int tahun, topAcc, topPw;

    Mobil(String merk,String tipe,int tahun,int topAcc,int topPw){
        this.merk = merk;
        this.tipe = tipe;
        this.tahun = tahun;
        this.topAcc = topAcc;
        this.topPw = topPw;
    }

    void print(){
        System.out.println("Merk: "+merk);
        System.out.println("Tipe: "+tipe);
        System.out.println("Tahun: "+tahun);
        System.out.println("Top Speed Acceleration: "+topAcc);
        System.out.println("Top Speed Power: "+topPw);
    }

    public static Mobil findMobilTopAcc(Mobil[] mobil, int low, int high) {
        if (low == high) {
            return mobil[low];
        }

        int mid = (low + high) / 2;

        Mobil mobilTercepatKiri = findMobilTopAcc(mobil, low, mid);

        Mobil mobilTercepatKanan = findMobilTopAcc(mobil, mid + 1, high);
        return mobilTercepatKiri.topAcc > mobilTercepatKanan.topAcc ? mobilTercepatKiri : mobilTercepatKanan;
    }

    public static Mobil findMobilLowAcc(Mobil[] mobil, int low, int high) {
        if (low == high) {
            return mobil[low];
        }
        int mid = (low + high) / 2;

        Mobil mobilTerendahKiri = findMobilLowAcc(mobil, low, mid);

        Mobil mobilTerendahKanan = findMobilLowAcc(mobil, mid + 1, high);
        return mobilTerendahKiri.topAcc < mobilTerendahKanan.topAcc ? mobilTerendahKiri : mobilTerendahKanan;
    }

    public static double findAvg(Mobil[] mobil) {
        double totalTopAcc = 0.0;
        for (Mobil mobil1 : mobil) {
            totalTopAcc += mobil1.topAcc;
        }
        return totalTopAcc / mobil.length;
    }



    public static void main(String[] args) {
        Mobil mobils[] = new Mobil[8];

        mobils[0] = new Mobil("BMW","M2 Couple",2016,6816,728);
        mobils[1] = new Mobil("Ford","Fiesta ST",2014,3921,575);
        mobils[2] = new Mobil("Nissan","370Z",2009,4360,657);
        mobils[3] = new Mobil("Subaru","BRZ",2014,4058,609);
        mobils[4] = new Mobil("Subaru","Impreza WRX",2013,6255,703);
        mobils[5] = new Mobil("Toyota","AE86",1986,3700,553);
        mobils[6] = new Mobil("Toyota","86?GT",2014,4180,609);
        mobils[7] = new Mobil("Volkswagen","Golf GTI",2014,4180,631);

        findMobilTopAcc(mobils, 0, mobils.length - 1).print();
        System.out.println("==========================");
        findMobilLowAcc(mobils, 0, mobils.length - 1).print();
        System.out.println();
        System.out.println("Rata-Rata: " + findAvg(mobils));

    }
}

```

```
g\Code\User\workspaceStorage\82a328fecd2f7a24369eccccf9fa7b25\redhat.java\jdt_
Merk: BMW
Tipe: M2 Couple
Tahun: 2016
Top Speed Acceleration: 6816
Top Speed Power: 728
========================
Merk: Toyota
Tipe: AE86
Tahun: 1986
Top Speed Acceleration: 3700
Top Speed Power: 553

Rata-Rata: 4683.75
PS C:\Users\Acer\Tugas Kuliah\Semester 2\Praktek Algoritma\jobsheet 5>
```