JOBSHEET IV

**BRUTE FORCE DAN DIVIDE CONQUER**

**Nama: Rizqi Bagus Andrean**

**Kelas: TI-1D**

**Absen: 25**

# Tujuan Praktikum

Setelah melakukan materi praktikum ini, mahasiswa mampu:

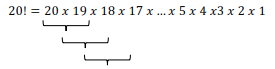
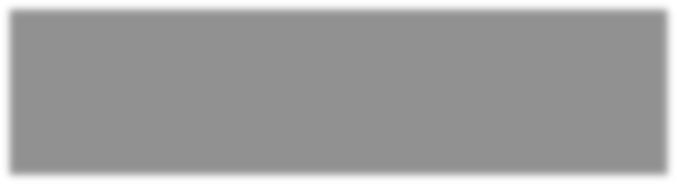
1. Mahasiswa mampu membuat algoritma bruteforce dan divide-conquer
2. Mahasiswa mampu menerapkan penggunaan algorima bruteforce dan divide-conquer

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

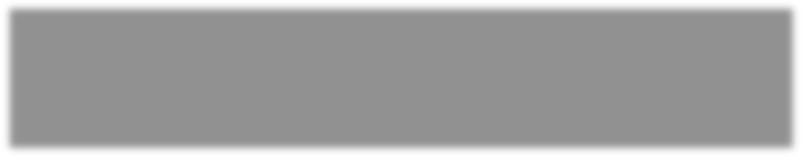
Perhatikan Diagram Class berikut ini :

|  |
| --- |
| 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 :

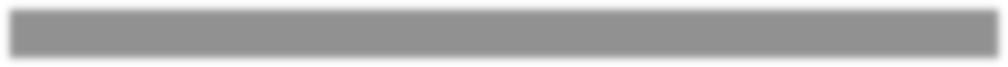


Tahapan pencarian nilai faktorial dengan algoritma Divide and Conquer :

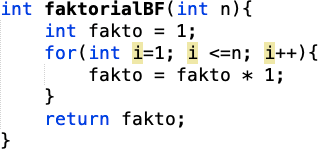


## Langkah-langkah Percobaan

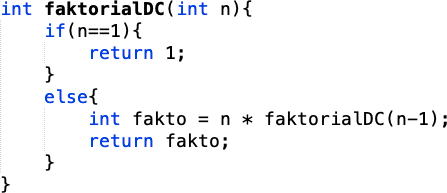
1. Buat Project baru, dengan nama “**BruteForceDivideConquer**”. Buat package dengan nama minggu5.
2. Buatlah class baru dengan nama **Faktorial**
3. Lengkapi class **Faktorial** dengan atribut dan method yang telah digambarkan di dalam diagram class di atas, sebagai berikut:
   1. Tambahkan atribut nilai



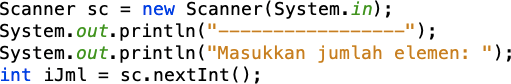
* 1. Tambahkan method faktorialBF() nilai



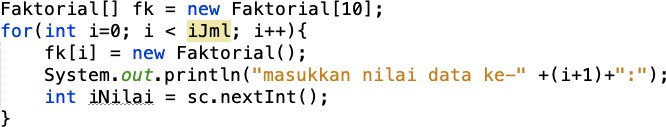
* 1. Tambahkan method faktorialDC() nilai



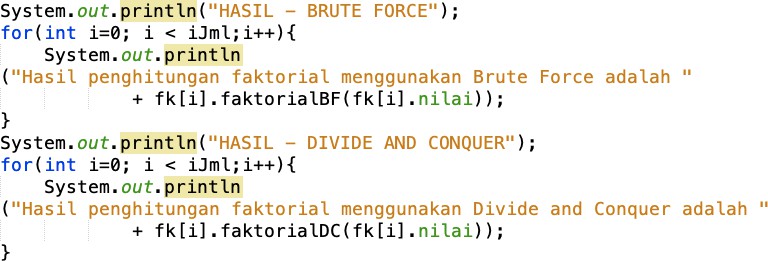
1. Coba jalankan (Run) class Faktorial dengan membuat class baru MainFaktorial.
   1. Di dalam fungsi main sediakan komunikasi dengan user untuk menginputkan jumlah angka yang akan dicari nilai faktorialnya



* 1. Buat Array of Objek pada fungsi main, kemudian inputkan beberapa nilai yang akan dihitung faktorialnya



* 1. Tampilkan hasil pemanggilan method faktorialDC() dan faktorialBF()

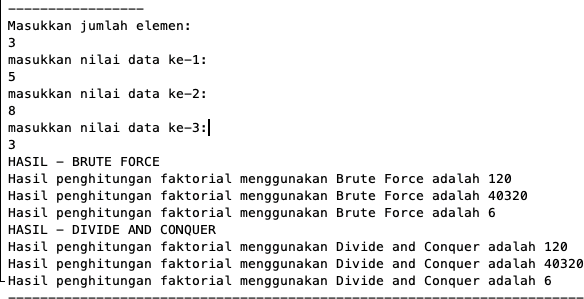


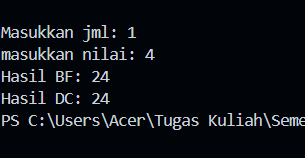
* 1. Pastikan program sudah berjalan dengan baik!



## Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.



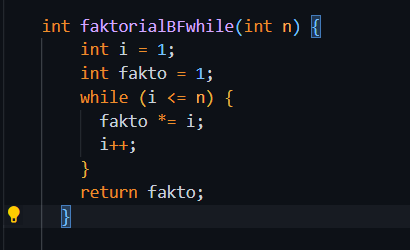


## Pertanyaan

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.

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



1. 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

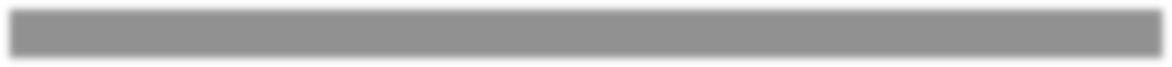
# Menghitung Hasil Pangkat dengan Algoritma Brute Force dan Divide and Conquer

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.

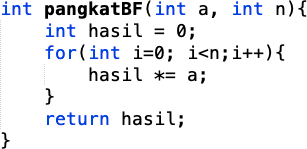
## Langkah-langkah Percobaan

1. Di dalam paket **minggu5**, buatlah class baru dengan nama **Pangkat**. Dan di dalam class **Pangkat**

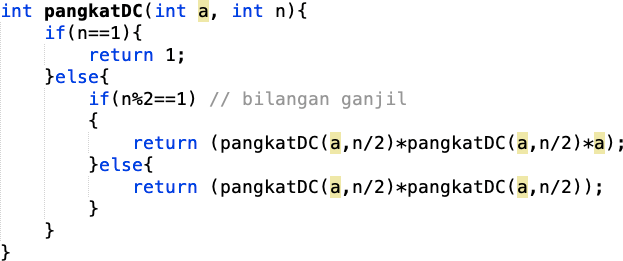
tersebut, buat atribut angka yang akan dipangkatkan sekaligus dengan angka pemangkatnya



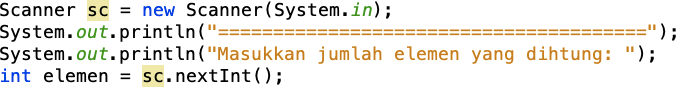
1. Pada class Pangkat tersebut, tambahkan method PangkatBF()

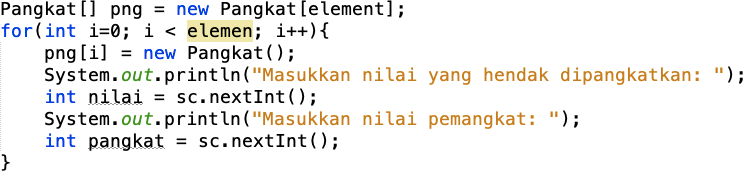


1. Pada class Pangkat juga tambahkan method PangkatDC()

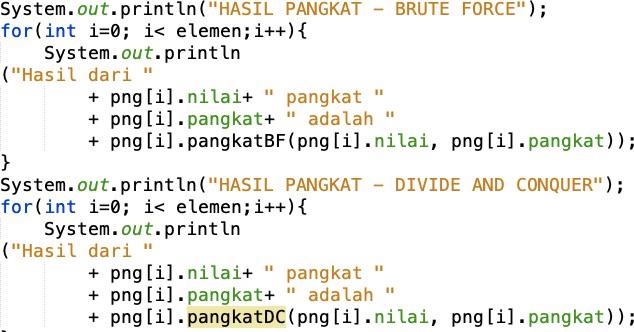


1. Perhatikan apakah sudah tidak ada kesalahan yang muncul dalam pembuatan class Pangkat
2. Selanjutnya buat class baru yang di dalamnya terdapat method main. Class tersebut dapat dinamakan MainPangkat. Tambahkan kode pada class main untuk menginputkan jumlah nilai yang akan dihitung pangkatnya.



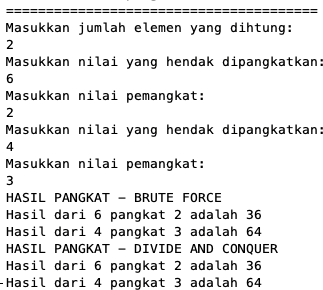
1. Nilai pada tahap 5 selanjutnya digunakan untuk instansiasi array of objek. Di dalam Kode berikut ditambahkan proses pengisian beberapa nilai yang akan dipangkatkan sekaligus dengan pemangkatnya.
2. Kemudian, panggil hasil nya dengan mengeluarkan return value dari method PangkatBF() dan

PangkatDC().

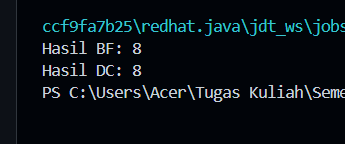


## Verifikasi Hasil Percobaan

Pastikan output yang ditampilkan sudah benar seperti di bawah ini.







## Pertanyaan

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

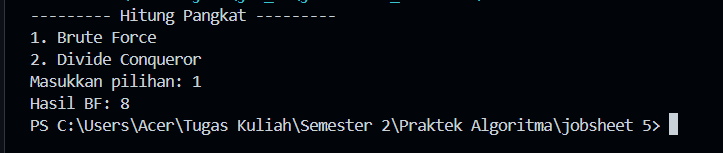
PangkatBF menggunakan looping untuk menghitung, tapi tidak cocok untuk nilai besar.

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

1. 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.

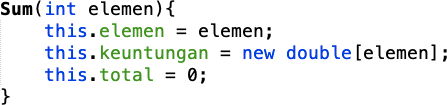
1. Modifikasi kode program tersebut, anggap proses pengisian atribut dilakukan dengan konstruktor.
2. Tambahkan menu agar salah satu method yang terpilih saja yang akan dijalankan menggunakan switch-case

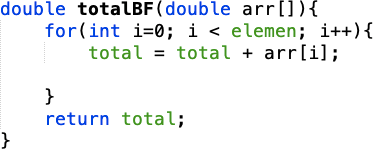


# Menghitung Sum Array dengan Algoritma Brute Force dan Divide and Conquer

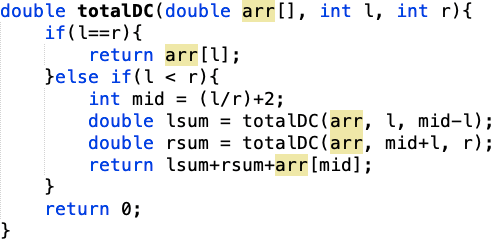
Di dalam percobaa1n ini, kita akan mempraktekkan bagaimana proses *divide, conquer*, dan *combine* diterapkan pada studi kasus penjumlahan keuntungan suatu perusahaan dalam beberapa bulan.

## Langkah-langkah Percobaan

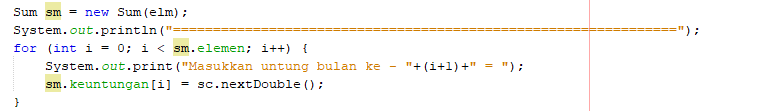
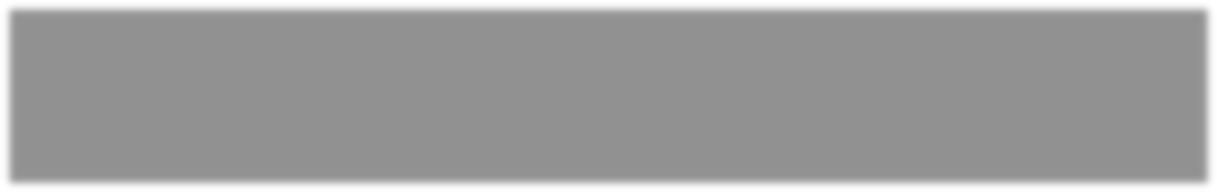
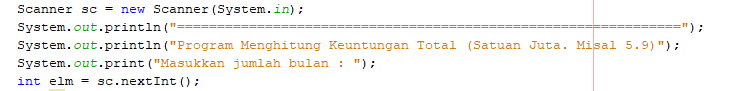
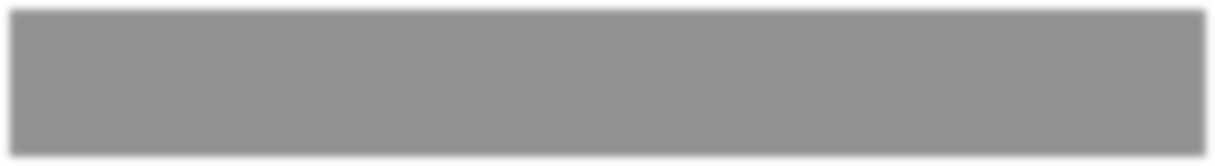
1. A green text on a white background  Description automatically generatedPada paket minggu5. Buat class baru yaitu class Sum. DI salam class tersebut terdapat beberapa atribut jumlah elemen array, array, dan juga total. Tambahkan pula konstruktor pada class Sum.
2. Tambahkan method TotalBF() yang akan menghitung total nilai array dengan cara *iterative*.



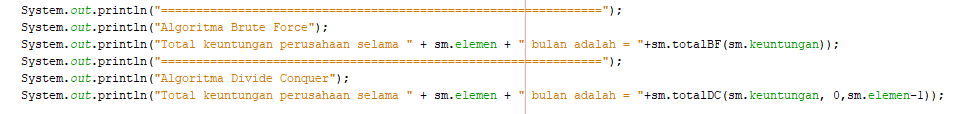
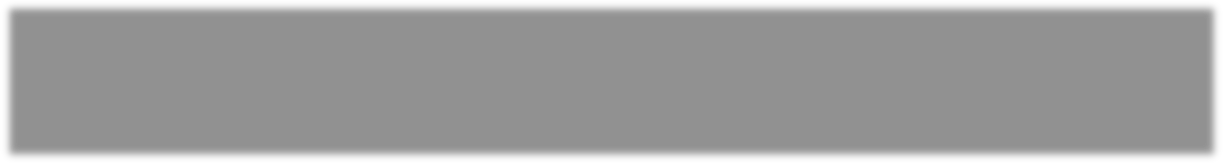
1. Tambahkan pula method TotalDC() untuk implementasi perhitungan nilai total array menggunakan algoritma Divide and Conquer



1. 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
2. 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.

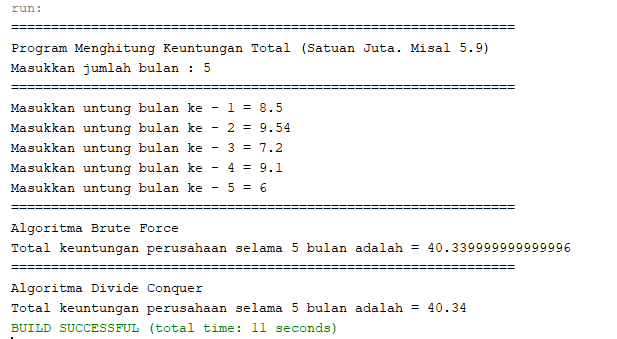
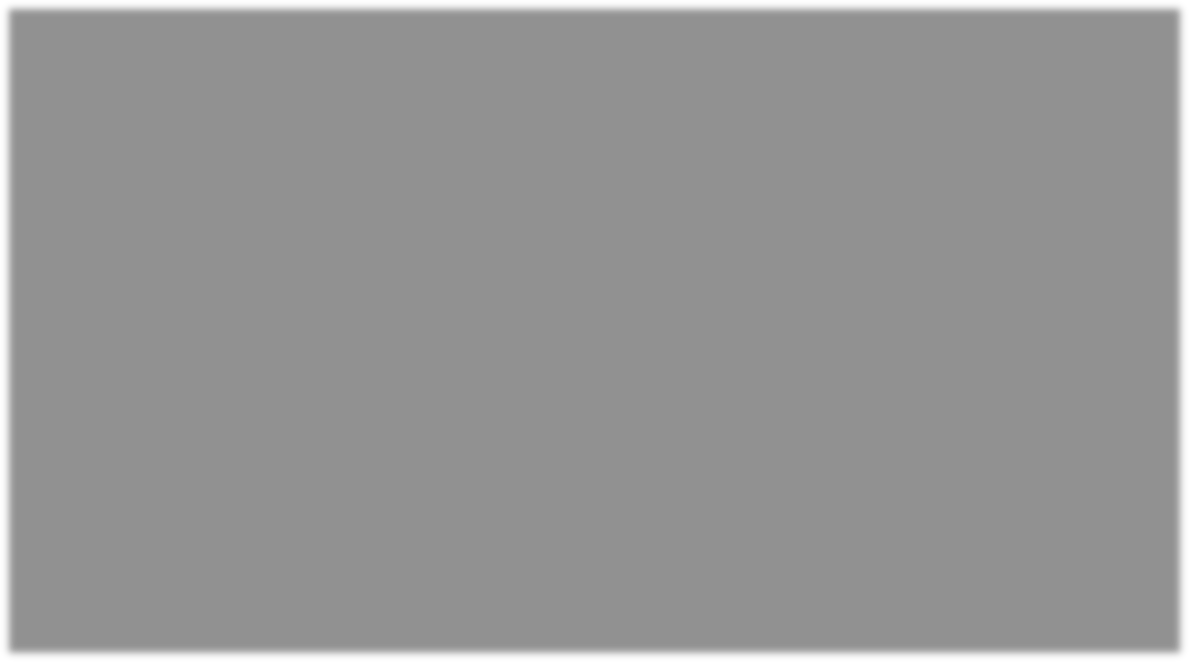


1. Tampilkan hasil perhitungan melalui objek yang telah dibuat untuk kedua cara yang ada (Brute Force dan Divide and Conquer)

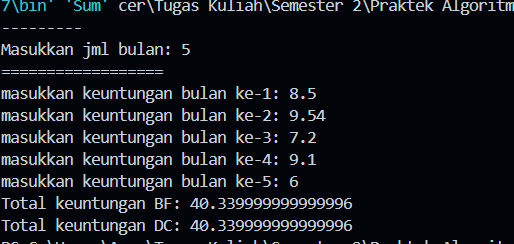


## Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.







## Pertanyaan

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

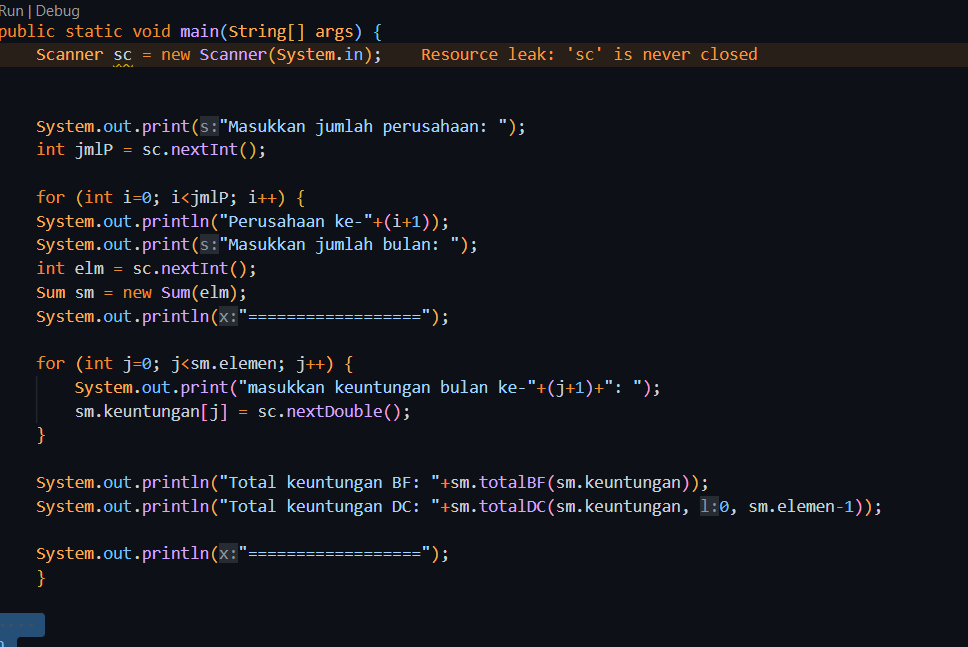


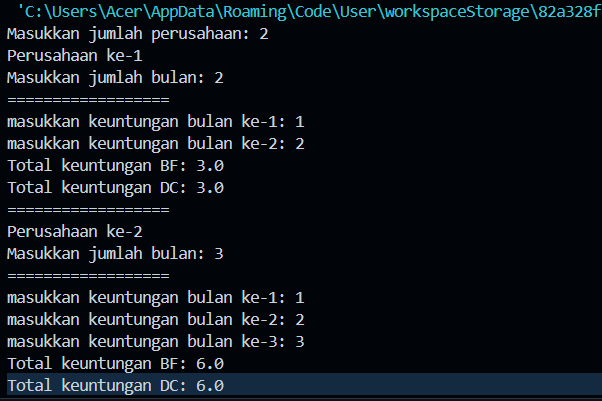
Formulasi return lsum + rsum pada kode totalDC digunakan untuk menghitung total nilai elemen dalam sebuah array dengan metode divide and conquer. Formulasi ini menggabungkan hasil dari sub-masalah kiri dan kanan dengan menambahkan nilai lsum dan rsum.

* + - 1. 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.

* + - 1. 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!





# 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:

* 1. top\_acceleration tertinggi menggunakan Divide and Conquer!
  2. top\_acceleration terendah menggunakan Divide and Conquer!
  3. Rata-rata top\_power dari seluruh mobil menggunakan Brute Force!



