**LAPORAN PRAKTIKUM**

**Politeknik Negeri Malang**

**Praktikan**

**ALGORITMA DAN STRUKTUR DATA**

****

2022

2141720183

RIDWAN CAESAR RIZQI KARISMA BIWARNI

TI 1C

Daftar Isi [harus diupdate setiap ada perubahan]

[Menghitung Nilai Faktorial dengan Algoritma Brute Force dan Devide Conquer 4](#_Toc98410079)

[Langkah 1 4](#_Toc98410080)

[Langkah 2 4](#_Toc98410081)

[Langkah 3 4](#_Toc98410082)

[Langkah 4 5](#_Toc98410083)

[Verifikasi hasil percobaan 6](#_Toc98410084)

[Pertanyaan 6](#_Toc98410085)

[Jawaban 7](#_Toc98410086)

[Menghitung Hasil Pangkat dengan Algoritma Brute Force dan Divide and Conquer 10](#_Toc98410087)

[Langkah 1 10](#_Toc98410088)

[Langkah 2 10](#_Toc98410089)

[Langkah 3 10](#_Toc98410090)

[Langkah 4 10](#_Toc98410091)

[Langkah 5 11](#_Toc98410092)

[Langkah 6 11](#_Toc98410093)

[Langkah 7 11](#_Toc98410094)

[Verifikasi Hasil Percobaan 11](#_Toc98410095)

[Pertanyaan 12](#_Toc98410096)

[Jawaban 12](#_Toc98410097)

[Menghitung Sum Array dengan Algoritma Brute Force dan Divide Conquer 14](#_Toc98410098)

[Langkah 1 14](#_Toc98410099)

[Langkah 2 14](#_Toc98410100)

[Langkah 3 14](#_Toc98410101)

[Langkah 4 15](#_Toc98410102)

[Langkah 5 15](#_Toc98410103)

[Langkah 6 15](#_Toc98410104)

[Verifikasi Hasil Percobaan 16](#_Toc98410105)

[Pertanyaan 16](#_Toc98410106)

[Jawaban 16](#_Toc98410107)

[Latihan Praktikum 19](#_Toc98410108)

[Jawaban Latihan Praktikum 20](#_Toc98410109)

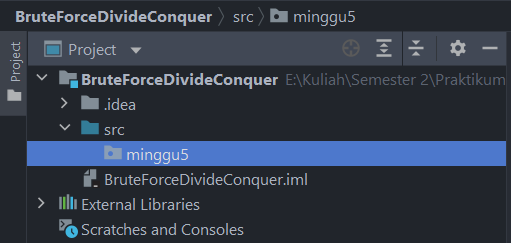
[Source code 20](#_Toc98410110)

[Output 21](#_Toc98410111)

# Menghitung Nilai Faktorial dengan Algoritma Brute Force dan Devide Conquer

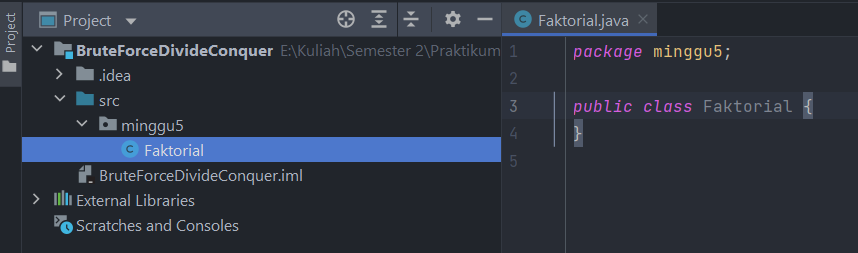
## Langkah 1

Buat Project baru, dengan nama “BruteForceDivideConquer”. Buat package dengan nama minggu5



## Langkah 2

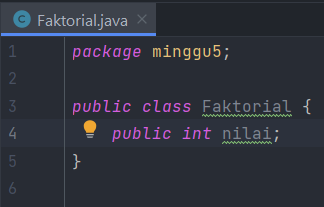
Buatlah class baru dengan nama Faktorial



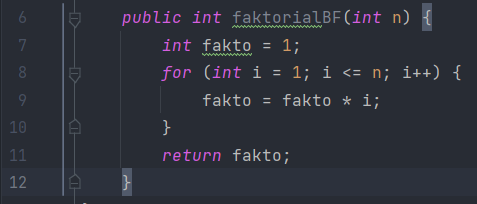
## Langkah 3

Lengkapi class Faktorial dengan atribut dan method yang telah digambarkan di dalam diagram class di atas, sebagai berikut:

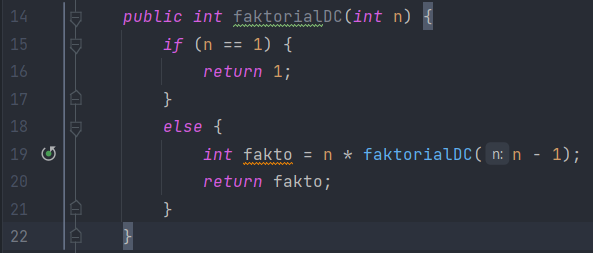
1. Tambahkan atribut nilai



1. Tambahkan method faktorialBF() nilai



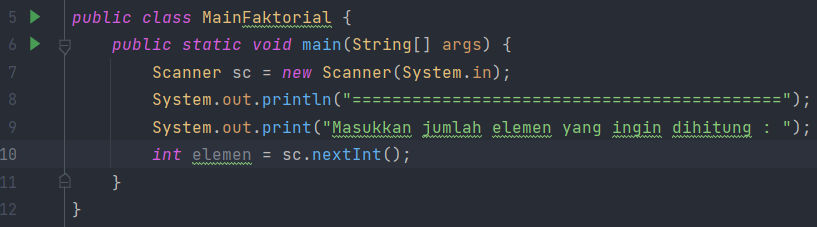
1. Tambahkan method faktorialDC() nilai



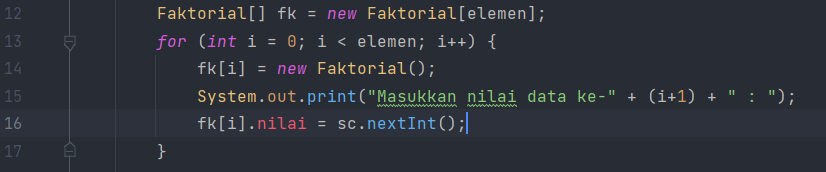
## Langkah 4

Coba jalankan (Run) class Faktorial dengan membuat class baru MainFaktorial

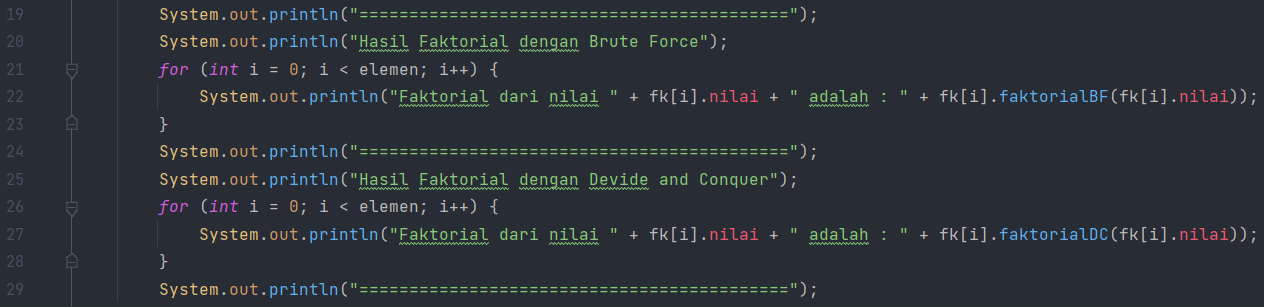
1. Di dalam fungsi main sediakan komunikasi dengan user untuk menginputkan jumlah angka yang akan dicari nilai faktorialnya



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



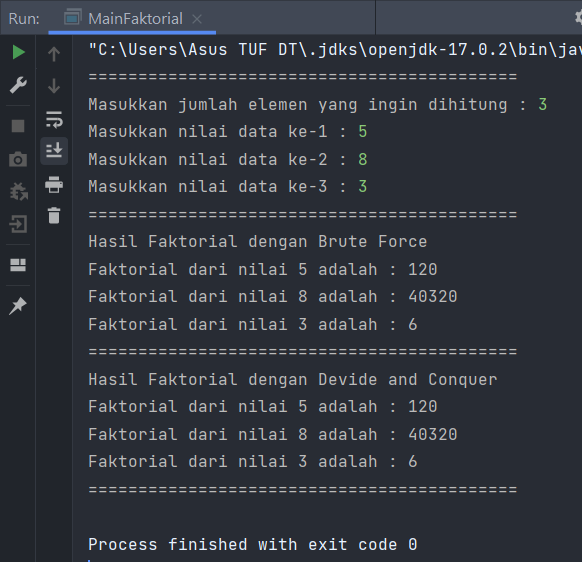
1. Tampilkan hasil pemanggilan method faktorialDC() dan faktorialBF()



1. Pastikan program sudah berjalan dengan baik!

## Verifikasi hasil percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.



### Pertanyaan

1. Jelaskan mengenai base line Algoritma Divide Conquer untuk melakukan pencarian nilai faktorial!
2. Pada implementasi Algoritma Divide and Conquer Faktorial apakah lengkap terdiri dari 3 tahapan divide, conquer, combine? Jelaskan masing-masing bagiannya pada kode program!
3. Apakah memungkinkan perulangan pada method faktorialBF() dirubah selain menggunakan for?Buktikan!
4. Tambahkan pegecekan waktu eksekusi kedua jenis method tersebut!
5. Buktikan dengan inputan elemen yang di atas 20 angka, apakah ada perbedaan waktu eksekusi?

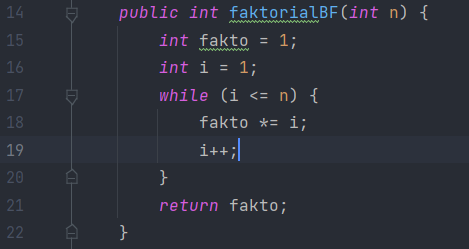
### Jawaban

1. Base line algoritma divide conquer untuk mencari nilai faktorial dari program tersebut adalah if (n==1 baris ini berfungsi untuk menghentikan perulangan rekursif dan sebagai tanda jika masalah sudah tidak bisa di pecah lagi.
2. Tahapan divide 🡪 if (n == 1) pada tahapan ini membagi masalah kedua kelompok yaitu base case dan jika n tidak sama dengan 1 maka akan dilakukan fungsi rekursif

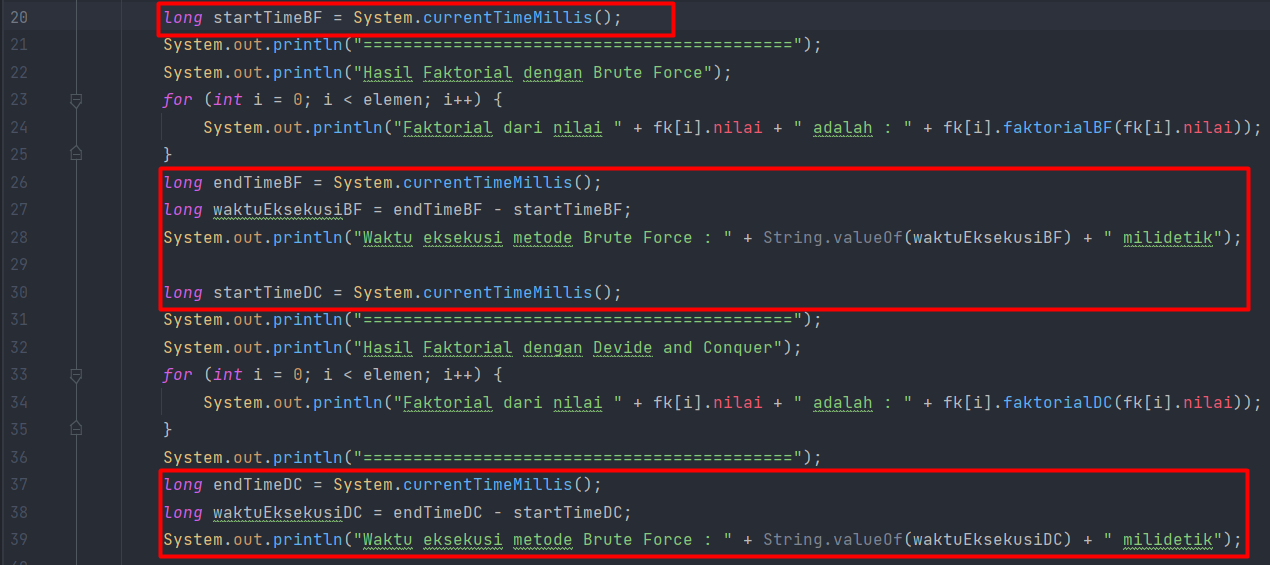
Tahapan conquer 🡪 int fakto = n \* faktorialDC(n-) pada tahapan ini masalah yang dipecah akan ditemukan solusinya menggunakan fungsi rekursif.

Tahpan combine 🡪 return 1 dan return fakto. Pada tahapan ini semua solusi yang sudah diselesaikan dari pecahan masalah akan disatukan kembali.

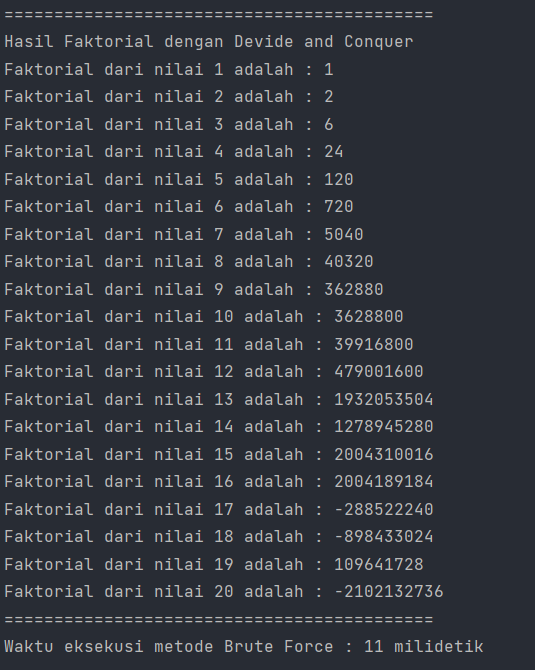
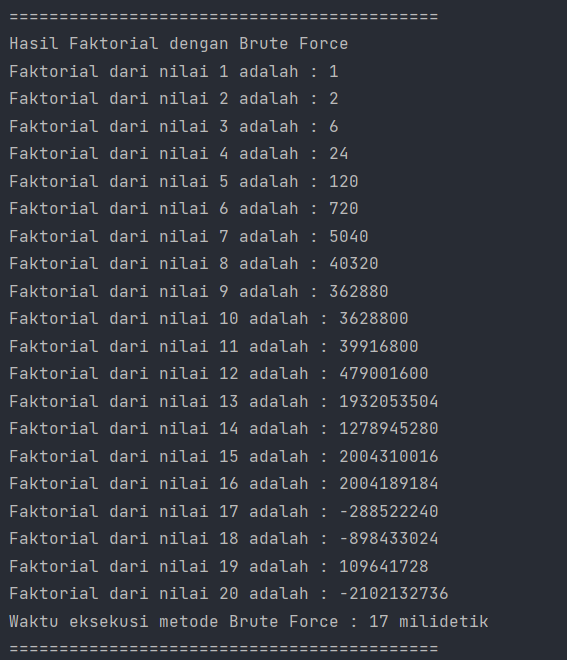
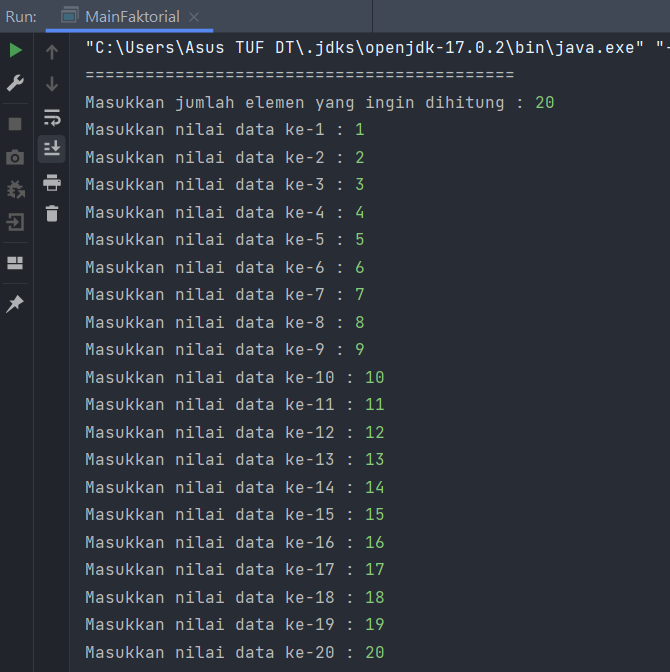
1. Bisa menggunakan perulangan tipe lainnya



1. Menambahkan perintah untuk menghitung waktu eksekusi kedua method



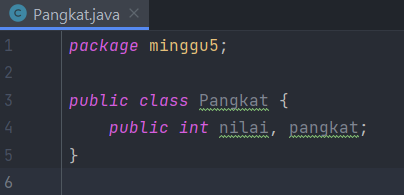
1. Ada perbedaan waktu eksekusi



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

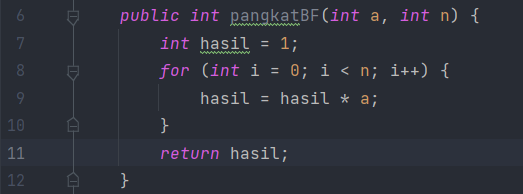
## Langkah 1

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



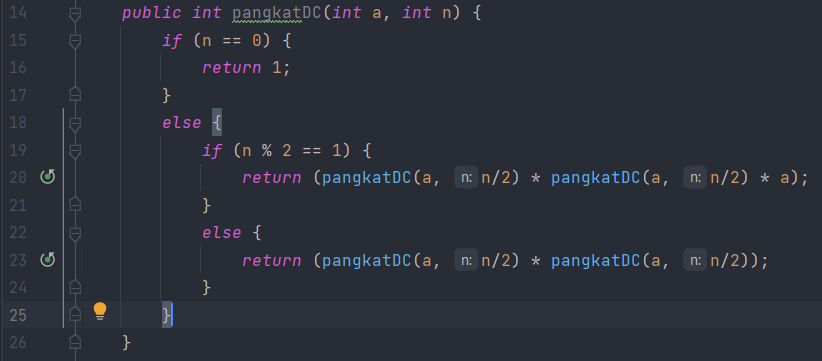
## Langkah 2

Pada class Pangkat tersebut, tambahkan method PangkatBF()



## Langkah 3

Pada class Pangkat juga tambahkan method PangkatDC()

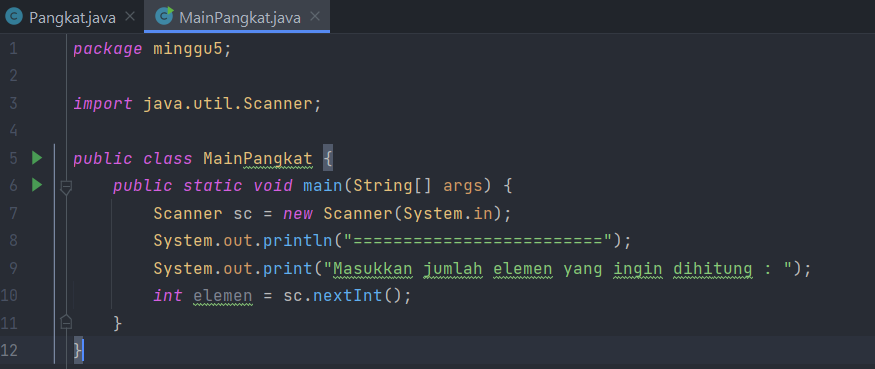


## Langkah 4

Perhatikan apakah sudah tidak ada kesalahan yang muncul dalam pembuatan class Pangkat

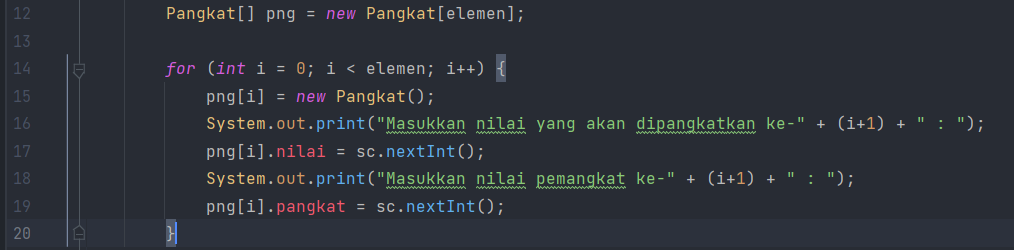
## Langkah 5

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.



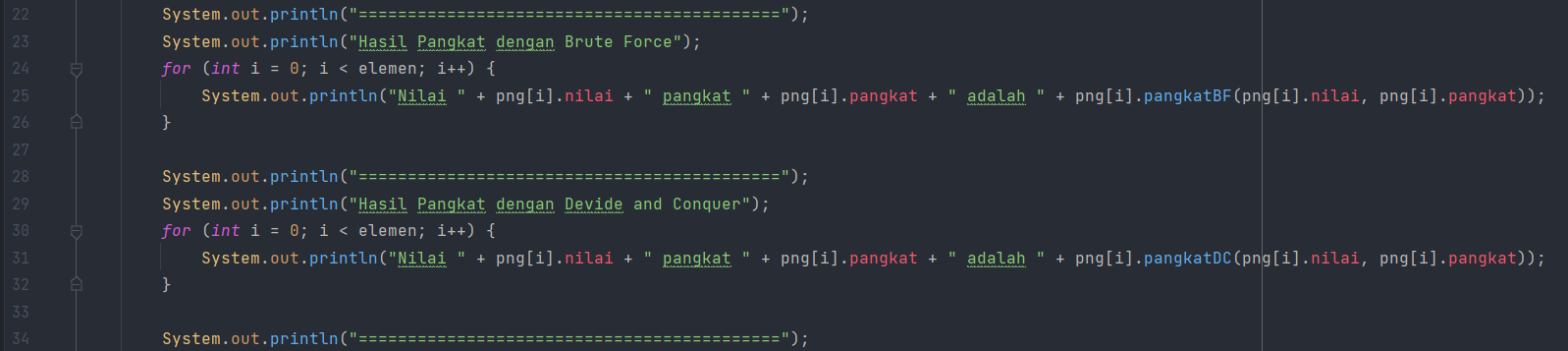
## Langkah 6

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.



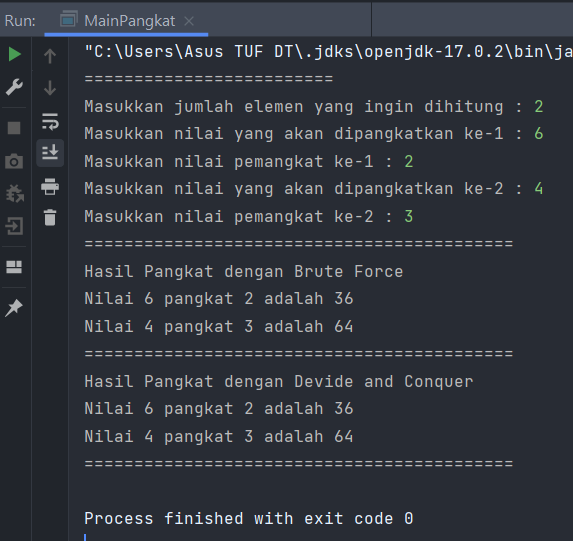
## Langkah 7

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



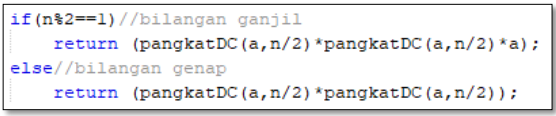
## Verifikasi Hasil Percobaan

Pastikan output yang ditampilkan sudah benar seperti di bawah ini.



### Pertanyaan

1. Jelaskan mengenai perbedaan 2 method yang dibuat yaitu PangkatBF() dan PangkatDC()!
2. Pada method PangkatDC() terdapat potongan program sebagai berikut:

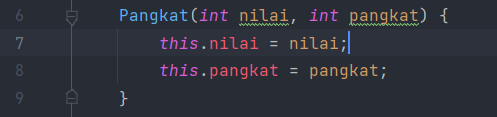


Jelaskan arti potongan kode tersebut

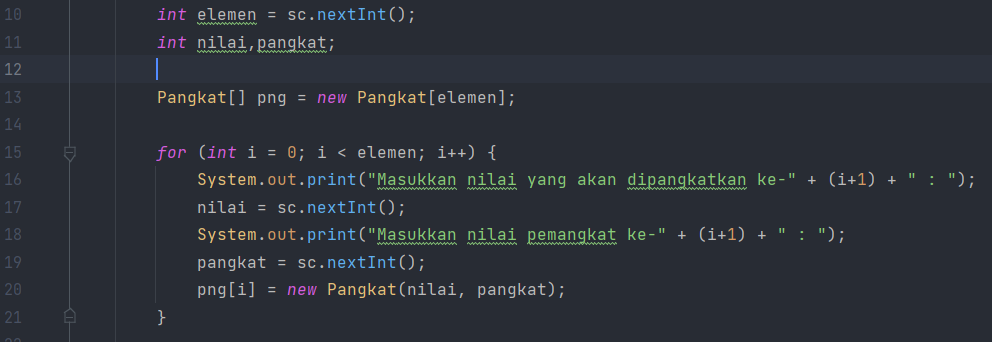
1. Apakah tahap combine sudah termasuk dalam kode tersebut?Tunjukkan!
2. Modifikasi kode program tersebut, anggap proses pengisian atribut dilakukan dengan konstruktor.
3. Tambahkan menu agar salah satu method yang terpilih saja yang akan dijalankan!

### Jawaban

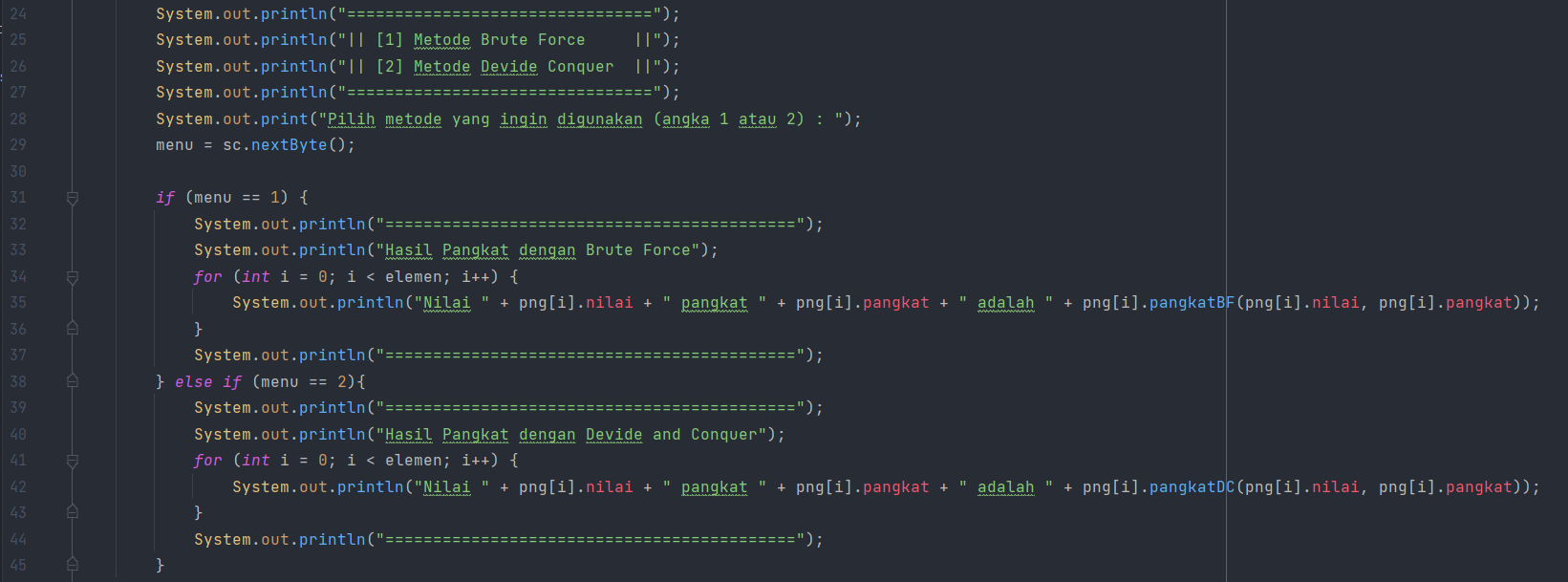
1. Pada method pangkatBF() menggunakan algoritma brute force sementara pada method pangkatDC() menggunakan algoritma devide conquer
2. Jika nilai n modulus 2 sama dengan 1 maka statement return (pangkatDC(a, n/2) \* pangkatDC(a, n/2) \* a); akan dijalankan sementara jika nilai n modulus 2 tidak sama dengan satu maka statement return (pangkatDC(a, n/2) \* pangkatDC()(a,n/2)); akan dijalankan
3. Tahap combine terjadi pada saat return 1, return (pangkatDC(a, n/2) \* pangkatDC(a, n/2) \* a) dan return (pangkatDC(a, n/2) \* pangkatDC()(a,n/2));
4. Source code class Pangkat



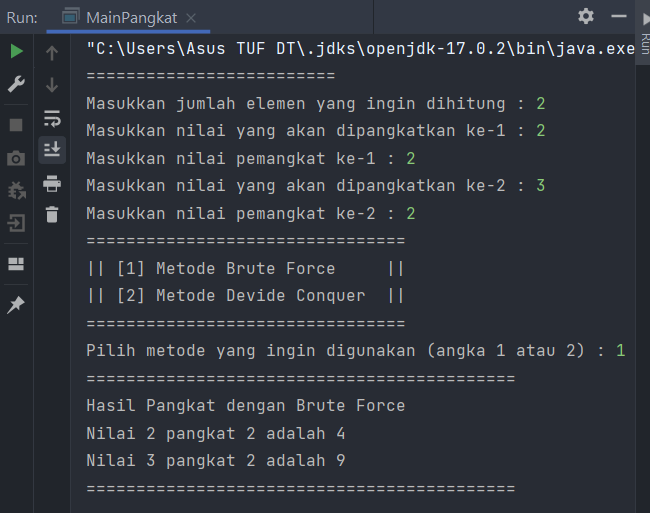
Source code class MainPangkat



1. Source code



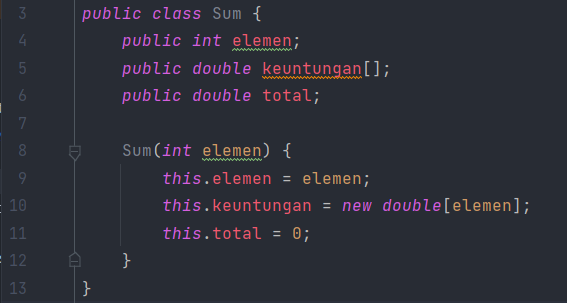
Output



# Menghitung Sum Array dengan Algoritma Brute Force dan Divide Conquer

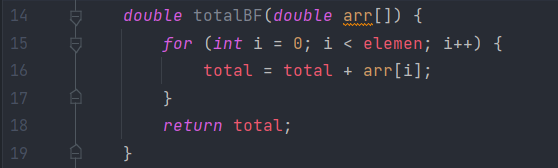
## Langkah 1

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.



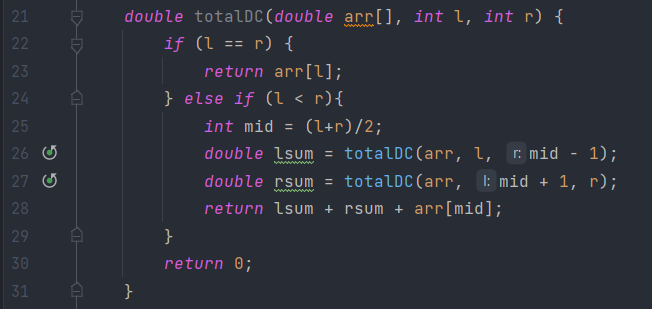
## Langkah 2

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



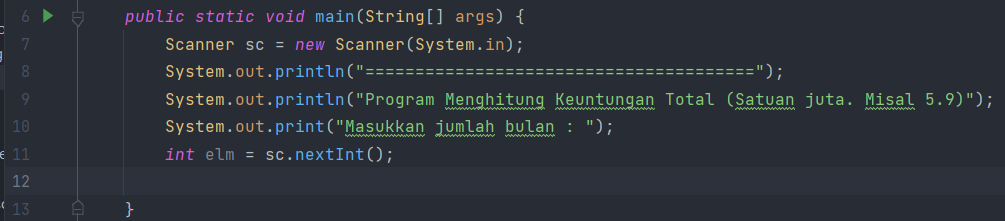
## Langkah 3

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



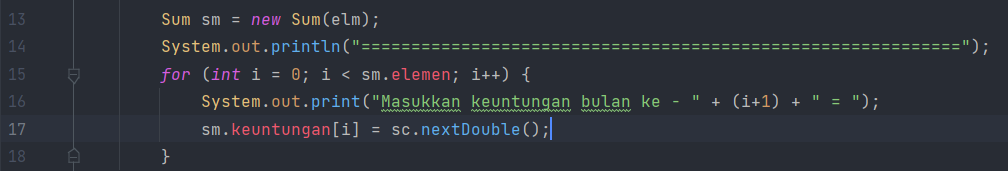
## Langkah 4

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



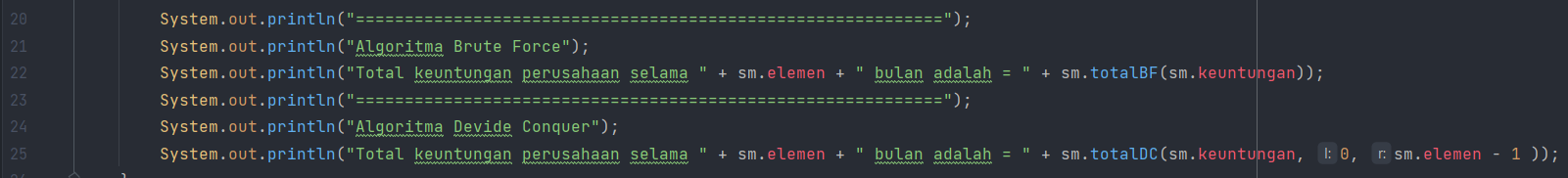
## Langkah 5

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

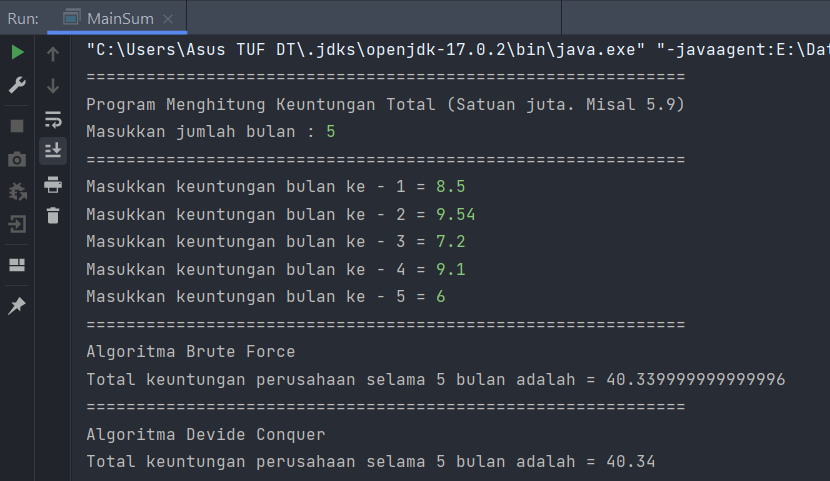


## Langkah 6

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



## Verifikasi Hasil Percobaan



### Pertanyaan

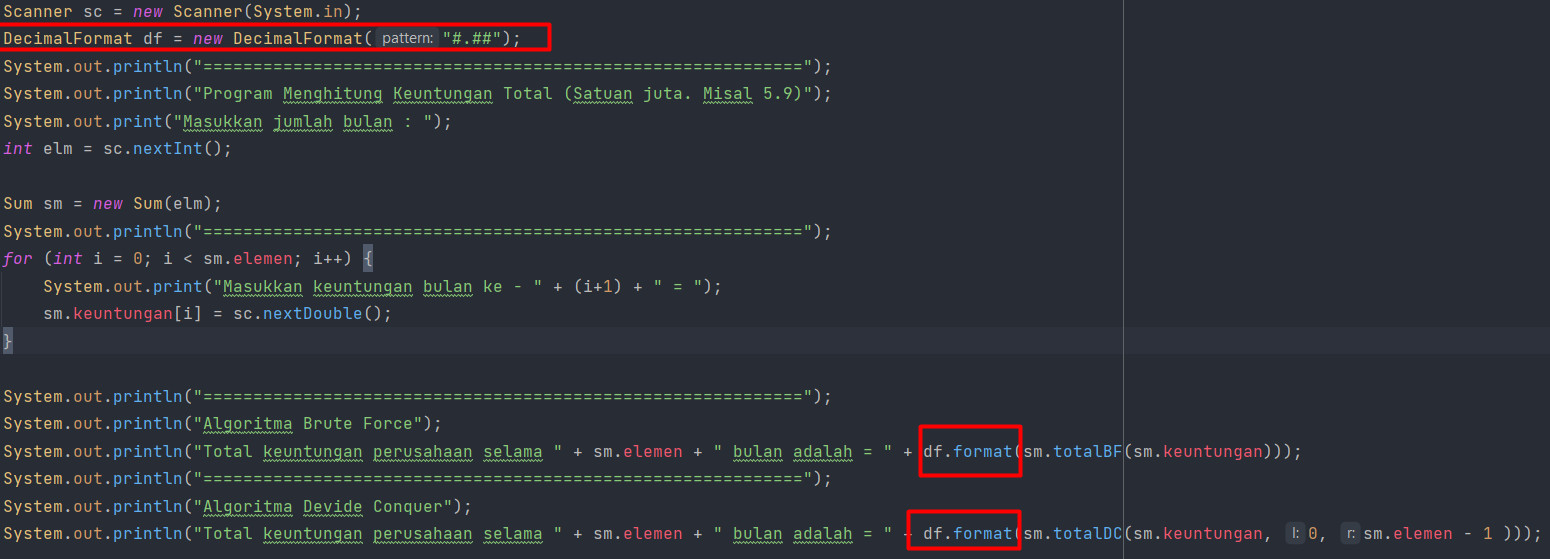
1. Berikan ilustrasi perbedaan perhitungan keuntungan dengan method TotalBF() ataupun TotalDC()
2. Perhatikan output dari kedua jenis algoritma tersebut bisa jadi memiliki hasil berbeda di belakang koma. Bagaimana membatasi output di belakang koma agar menjadi standar untuk kedua jenis algoritma tersebut.
3. Mengapa terdapat formulasi return value berikut?Jelaskan!



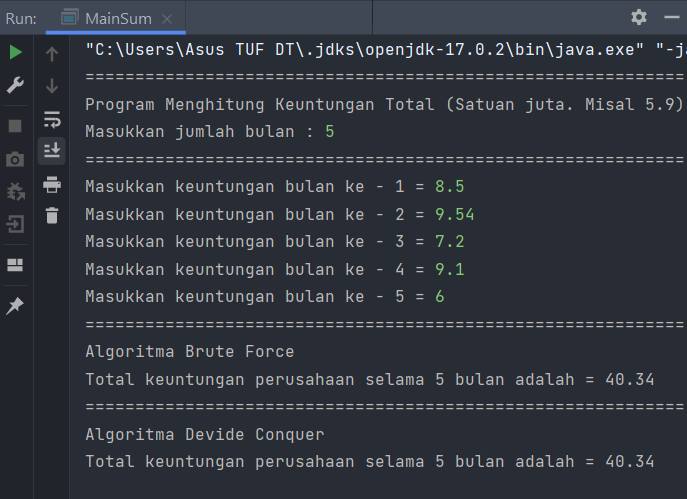
1. Kenapa dibutuhkan variable mid pada method TotalDC()?
2. Program perhitungan keuntungan suatu perusahaan ini hanya untuk satu perusahaan saja. Bagaimana cara menghitung sekaligus keuntungan beberapa bulan untuk beberapa perusahaan.(Setiap perusahaan bisa saja memiliki jumlah bulan berbeda-beda)? Buktikan dengan program!

### Jawaban

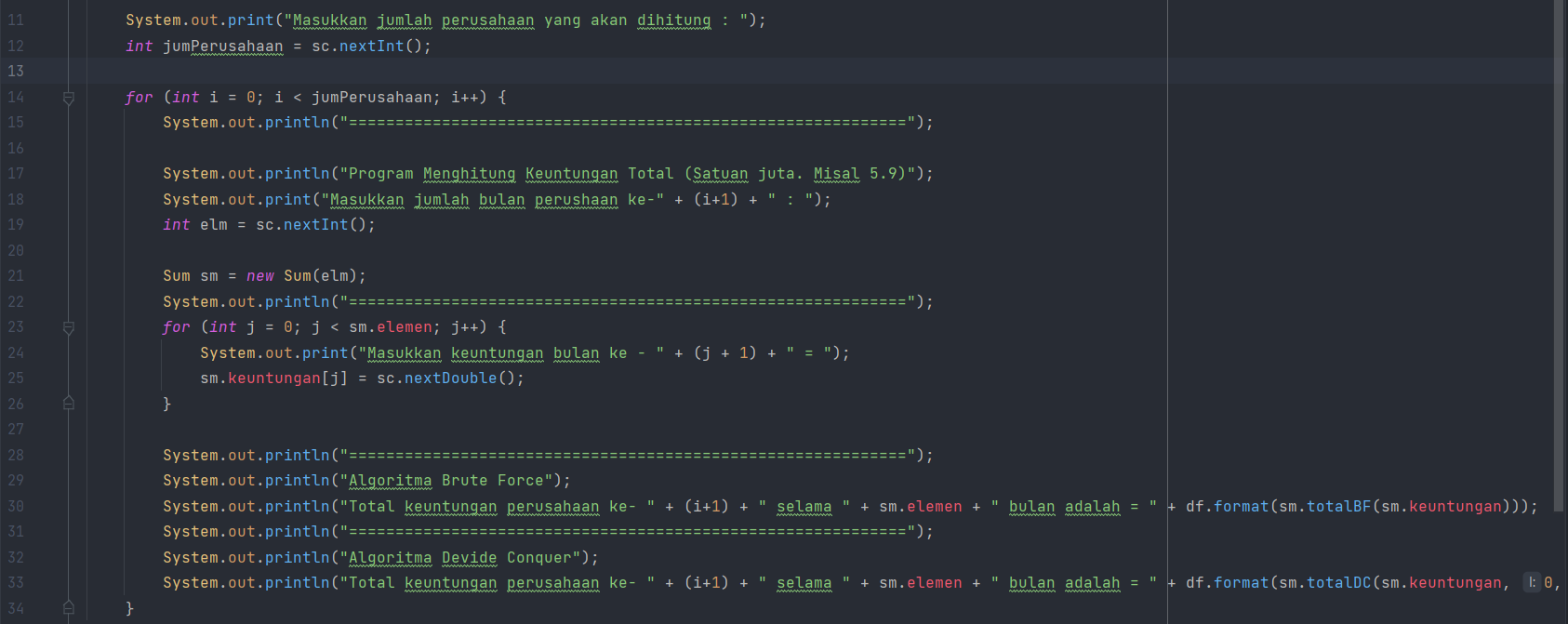
1. Perhitungan keuntungan menggunakan method TotalBF() menjumlahkan nilai keuntungan dari kiri ke kanan sedangkan jika menggunakan method TotalDC() memecah nilai array keuntungan menjadi bagian yang lebih kecil kemudian menjumlahkannya.
2. Source code



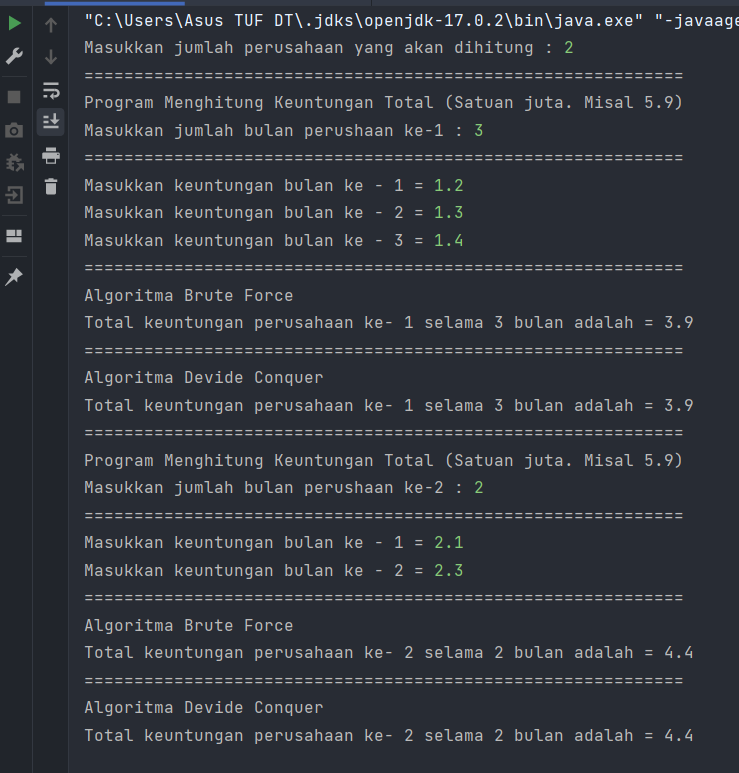
Output



1. Untuk menjumlahkan jumlah bagian kiri ditambah jumlah bagian kanan dan jumlah nilai tengah array.
2. Untuk mengetahui nilai median dari array
3. Source code



Output



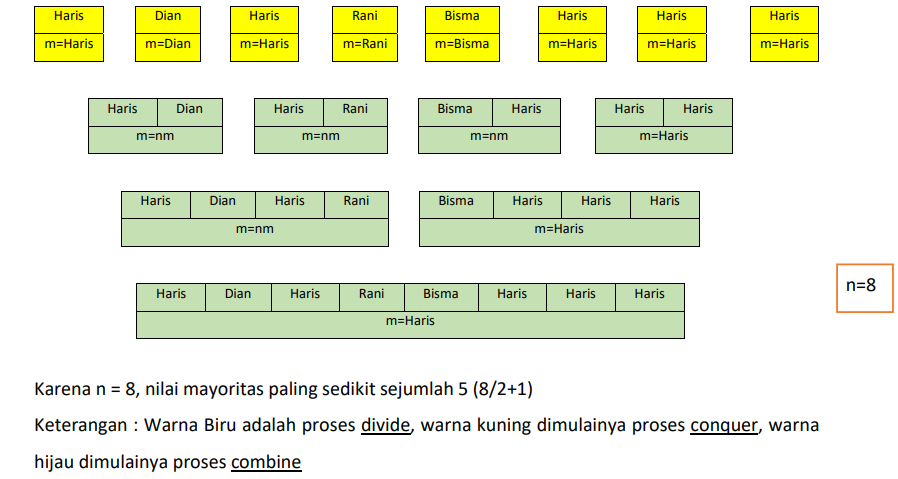
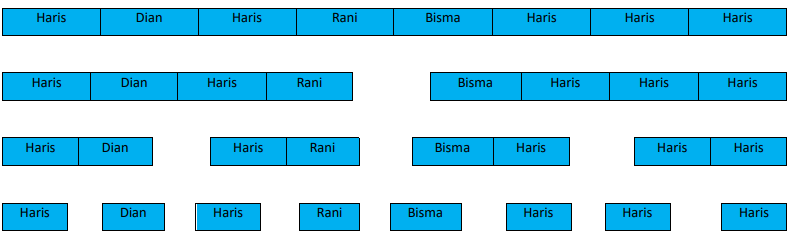
# Latihan Praktikum

Suatu Perguruan Tinggi di kota Malang sedang mengadakan pemilihan suara untuk memilih ketua BEM tahun 2022. Jika jumlah suara yang terkumpul diumpamakan selalu genap. Maka dengan inputan kandidat terpilih, carilah mayoritas jumlah suara untuk masing-masing kandidat. (Jumlah elemen array dan hasil pemilhan suara merupakan inputan user).

**Elemen Mayoritas : Elemen mayoritas di dalam A adalah elemen yang terdapat pada lebih dari n/2 posisi. Contohnya, jika n=6 atau n=7 maka nilai mayoritas paling sedikit adalah 4. Berasal dari (7/2)+1 atau (6/2)+1).**

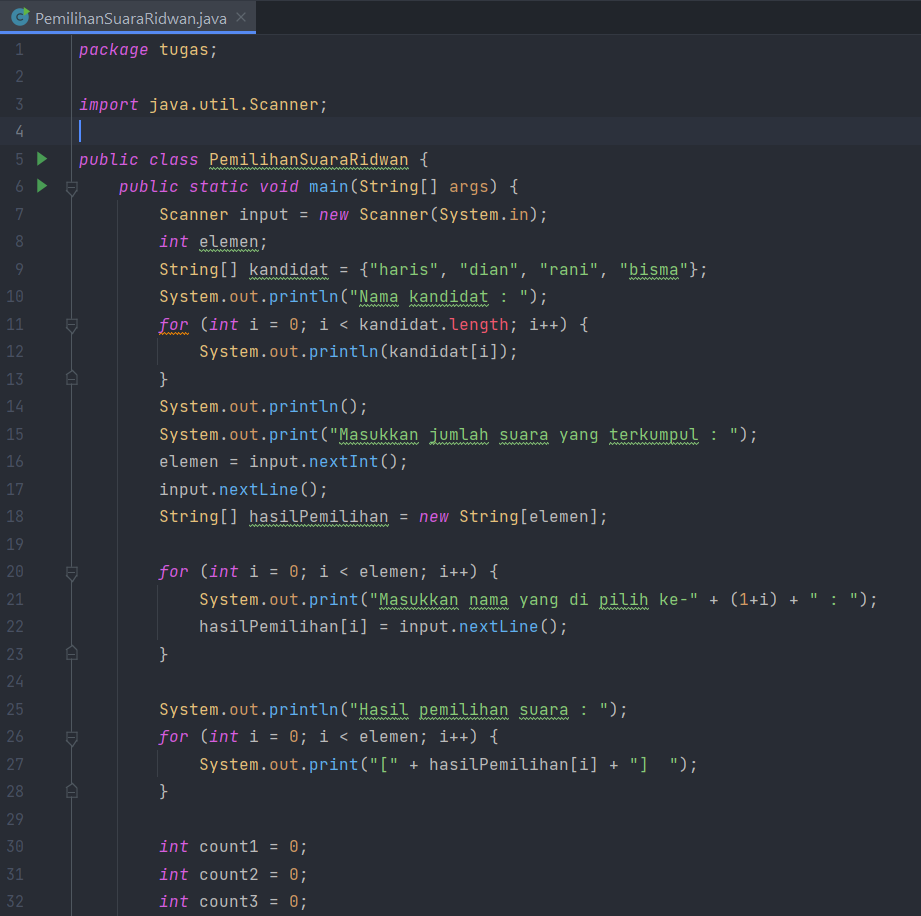
Nilai mayoritas berbeda konsep dengan menghitung total suara terbanyak kandidat terpilih !

Contoh : Hasil pemilihan suara sebagai berikut (m adalah mayoritas, nm adalah no mayoritas)



## Jawaban Latihan Praktikum

### Source code





### Output

