 **NAMA : FITRI CAHYANIATI**

**CLASS : SIB1G**

**NO ID : 11**

Jobsheet Iv  
Brute Force Dan Divide Conquer

1. Tujuan Praktikum

Setelah Melakukan Materi Praktikum Ini, Mahasiswa Mampu:

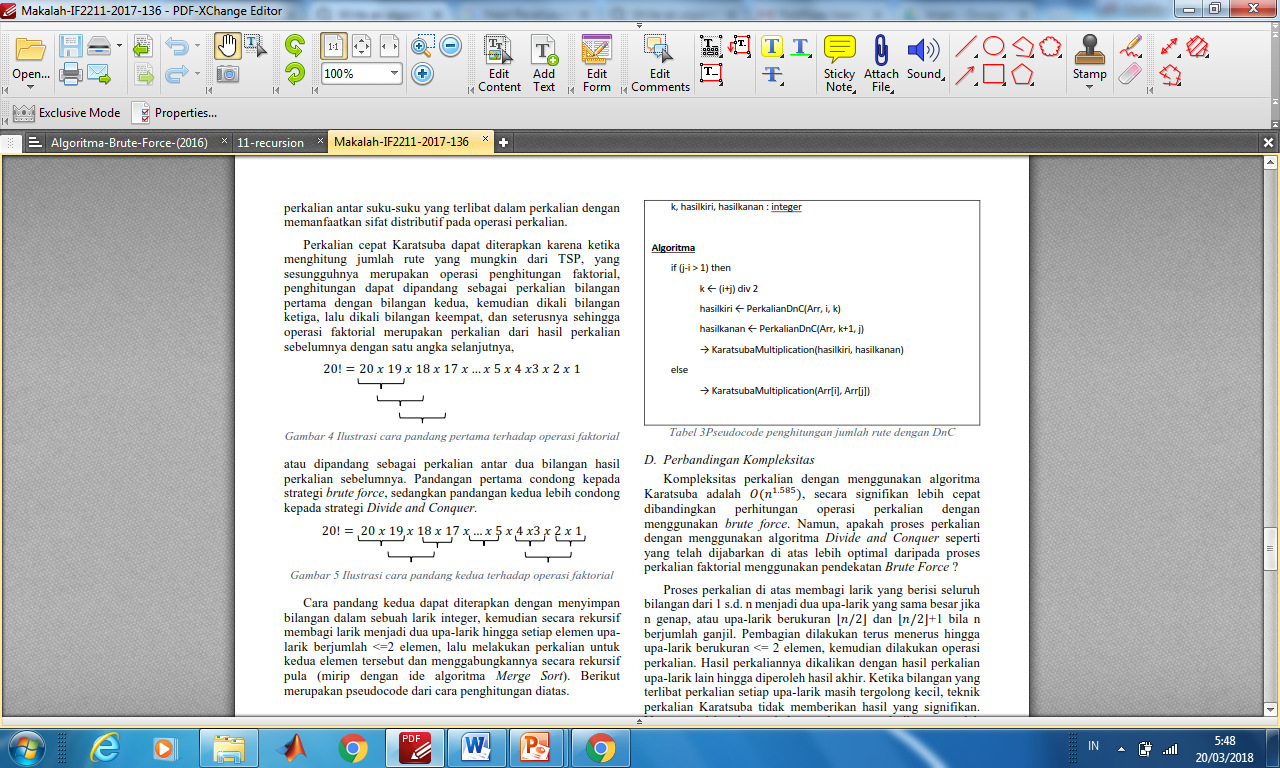
1. Mahasiswa Mampu Membuat Algoritma Bruteforce Dan Divide-Conquer
2. Mahasiswa Mampu Menerapkan Penggunaan Algorima Bruteforce Dan Divide-Conquer
3. Menghitung Nilai Faktorial Dengan Algoritma Brute Force Dan Divide And Conquer

Perhatikan Diagram Class Berikut Ini :

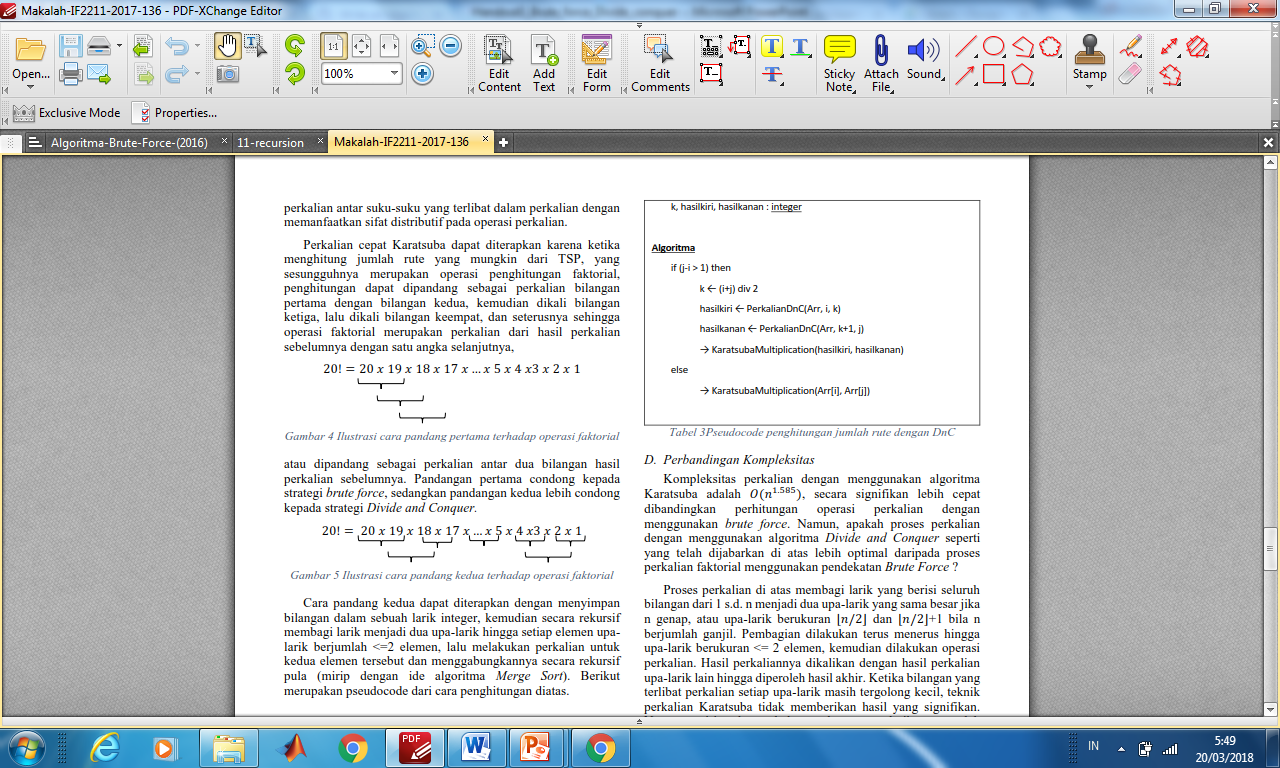
|  |
| --- |
| Faktorial |
| Nilai: Int |
| Faktorialbf(): Int  Faktorialdc(): Int |

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 :

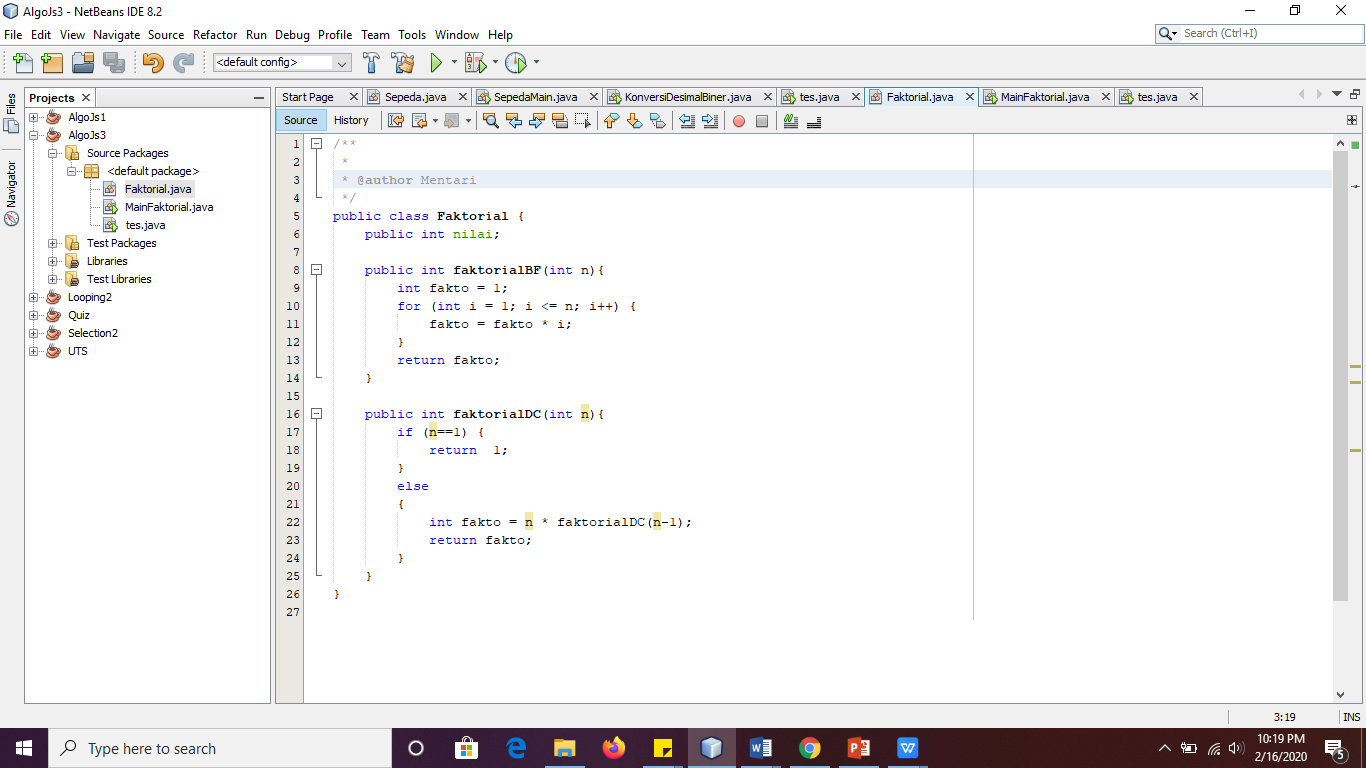
Tahapan Pencarian Nilai Faktorial Dengan Algoritma Brute Force :



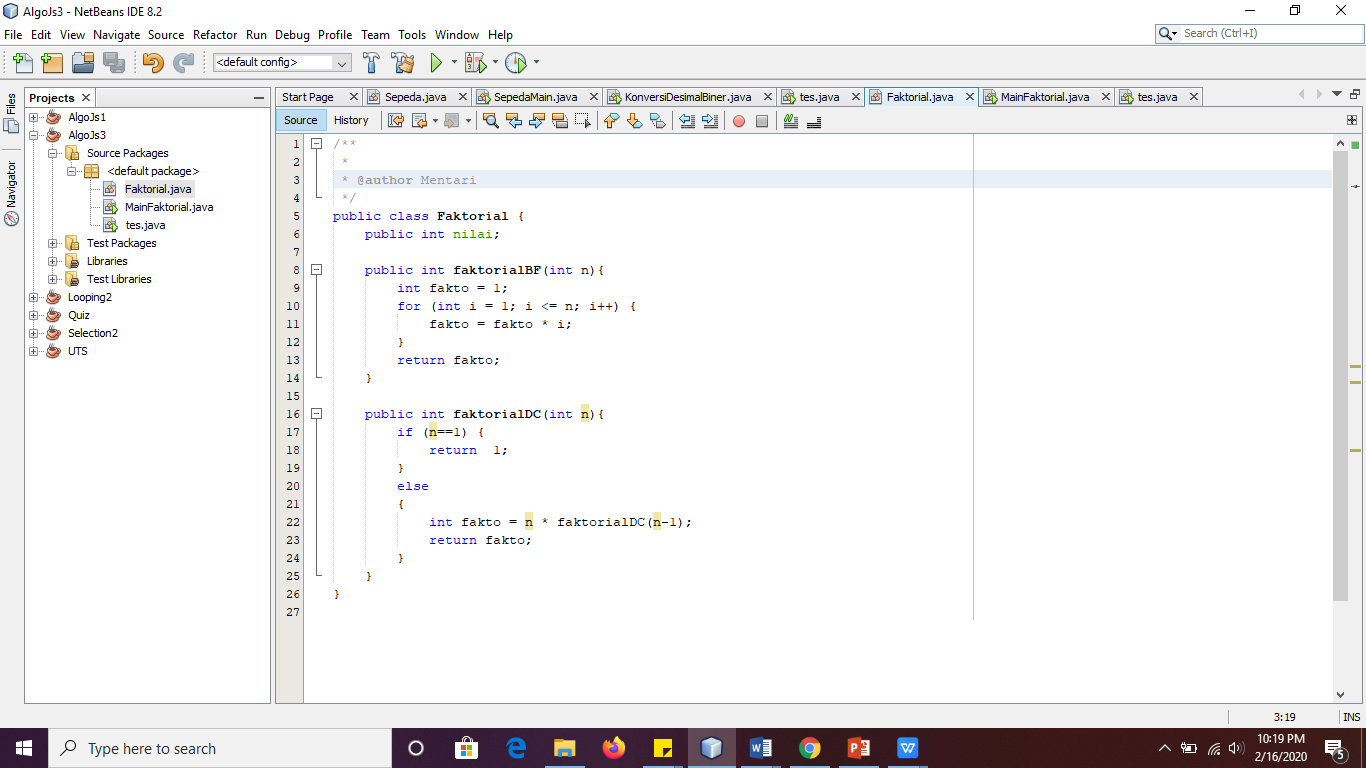
Tahapan Pencarian Nilai Faktorial Dengan Algoritma Divide And Conquer :



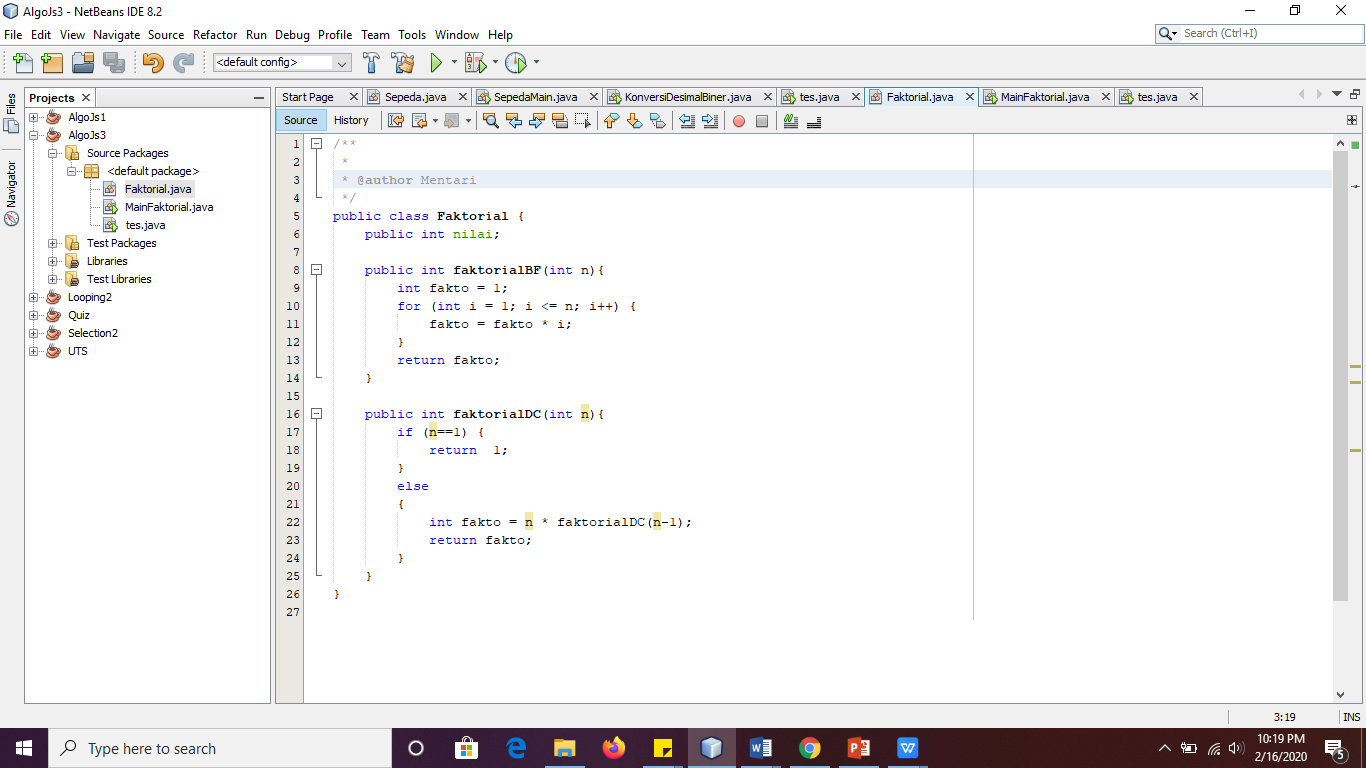
1. Langkah-Langkah Percobaan
2. Buat Project Baru, DenganNama “**Bruteforcedivideconquer**”. Buat Package Dengan Nama Minggu5.
3. Buatlah Class Baru Dengan Nama **Faktorial**
4. Lengkapi Class **Faktorial** Dengan Atribut Dan Method Yang Telah Digambarkan Di Dalam Diagram Class Di Atas, Sebagai Berikut:
5. Tambahkan Atribut Nilai



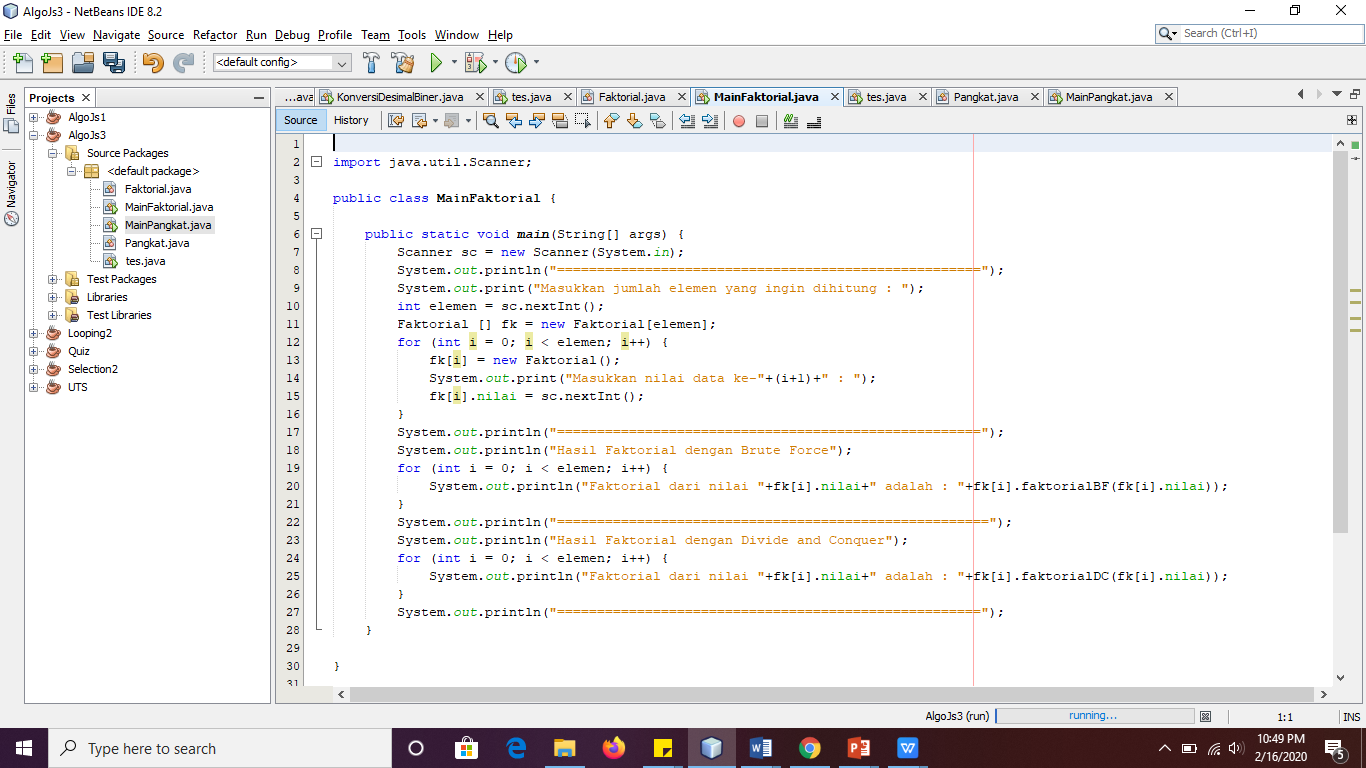
1. Tambahkan Method Faktorialbf() Nilai



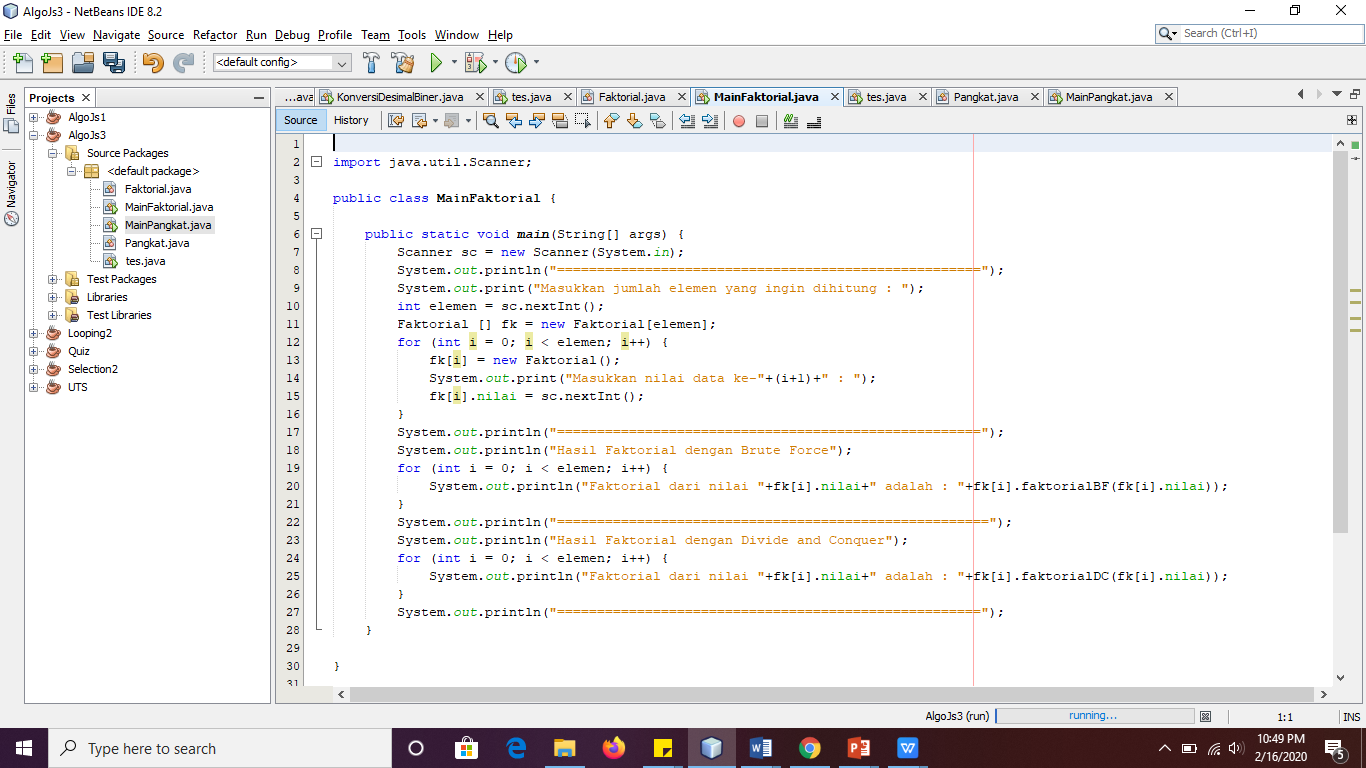
1. Tambahkan Method Faktorialdc() Nilai



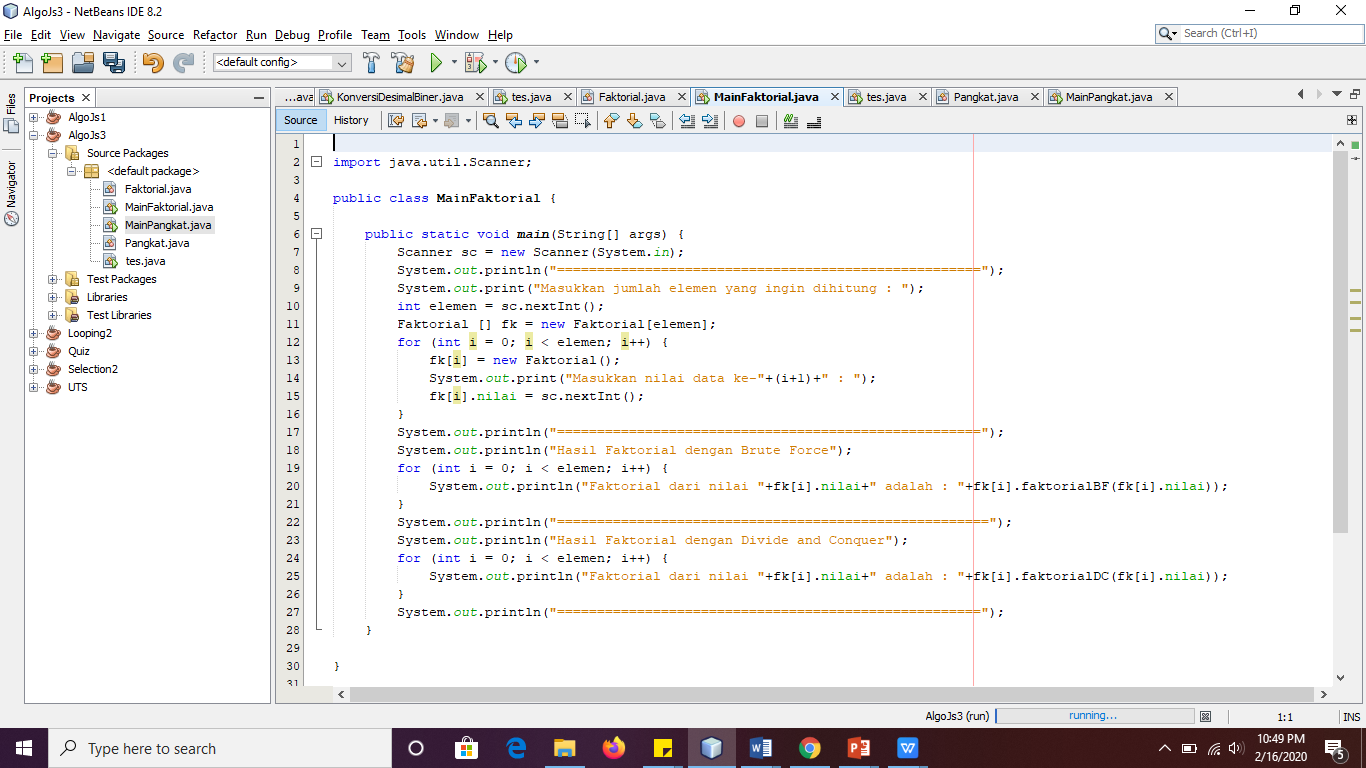
1. Coba Jalankan (Run) Class Faktorial Dengan Membuat Class Baru Mainfaktorial.
2. Di Dalam Fungsi Main Sediakan Komunikasi Dengan User Untuk Menginputkan Jumlah Angka Yang Akan Dicari Nilai Faktorialnya



* 1. Buat Array Of Objek Pada Fungsi Main, Kemudian Inputkan Beberapa Nilai Yang Akan Dihitung Faktorialnya



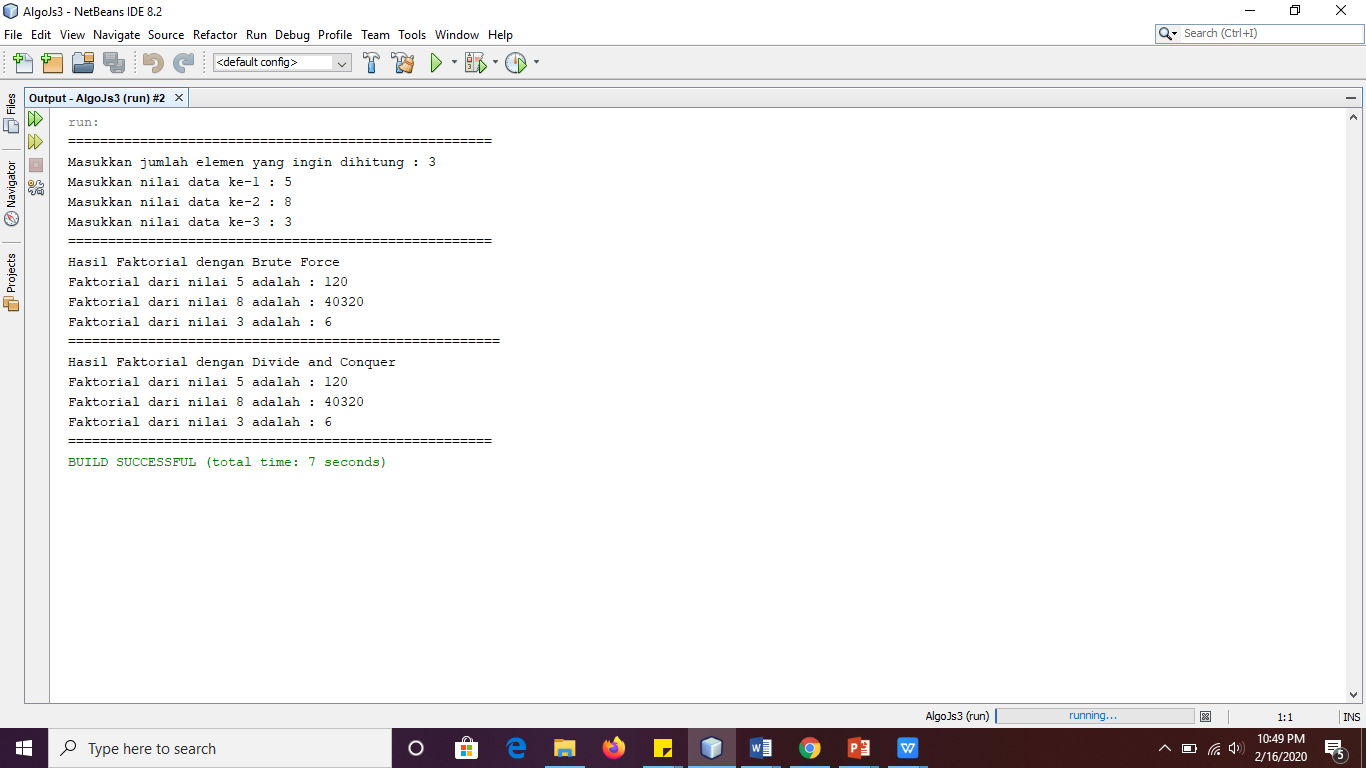
1. Tampilkan Hasil Pemanggilan Method Faktorialdc() Dan Faktorialbf()

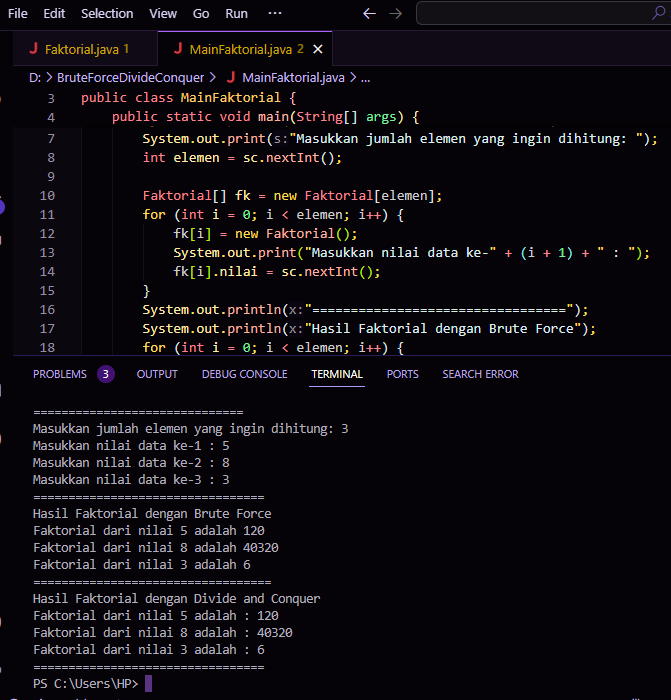


* 1. Pastikan Program Sudah Berjalan Dengan Baik!

1. Verifikasi Hasil Percobaan

Cocokkan Hasil Compile Kode Program Anda Dengan Gambar Berikut Ini.





1. Pertanyaan
2. Jelaskan Mengenai Base Line Algoritma Divide Conquer Untuk Melakukan Pencarian Nilai Faktorial!

**Baseline Algoritma Divide And Conquer Untuk Faktorial:**

* **Pemecahan Masalah:**

**Pecah Masalah Faktorial N Menjadi Dua Submasalah Dengan N/2.**

* **Penyelesaian Submasalah:**

**Hitung Faktorial Dari Masing-Masing Submasalah Secara Rekursif.**

* **Gabungkan Solusi:**

**Gabungkan Hasil Dari Masing-Masing Submasalah Dengan Operasi Perkalian.**

1. Pada Implementasi Algoritma Divide And Conquer Faktorial Apakah Lengkap Terdiri Dari 3 Tahapan Divide, Conquer, Combine? Jelaskan Masing-Masing Bagiannya Pada Kode Program!

**Dalam Implementasi Algoritma Divide And Conquer Untuk Faktorial:**

**1. Divide (Pemecahan Masalah)**

**- Masalah Faktorial \( N \) Dibagi Menjadi Dua Submasalah Yang Lebih Kecil Secara Rekursif Dalam Method `Faktorialdc`.**

**2. Conquer (Penyelesaian Submasalah)**

**- Setiap Submasalah Diselesaikan Dengan Menghitung Faktorial Dari Nilai Setengah \( N \) Secara Rekursif Dalam Method `Faktorialdc`.**

**3. Combine (Gabungkan Solusi)**

**- Hasil Dari Setiap Submasalah Secara Implisit Digabungkan Dengan Mengalikan Solusi Submasalah Untuk Mendapatkan Solusi Akhir Faktorial \( N \).**

1. Apakah Memungkinkan Perulangan Pada Method Faktorialbf() Dirubah Selain Menggunakan For?Buktikan!

**Ya, Memungkinkan Untuk Mengimplementasikan Perulangan Pada Method Faktorialbf() Selain Menggunakan For. Kita Dapat Menggunakan Perulangan While Sebagai Alternatif**.

Public Int Faktorialbf(Int N) {

    Int Fakto = 1;

    Int I = 1;

    While (I <= N) {

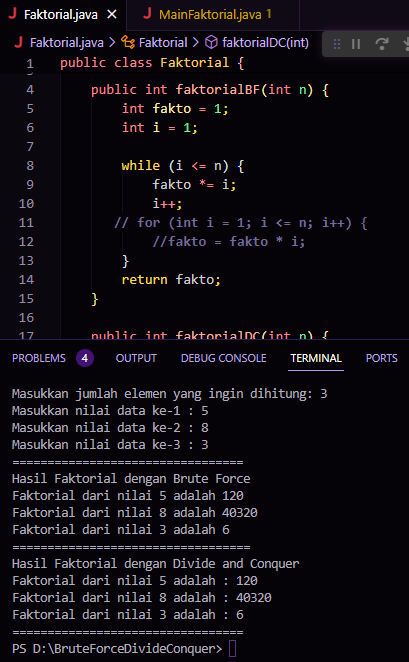
        Fakto \*= I;

        I++;

    }

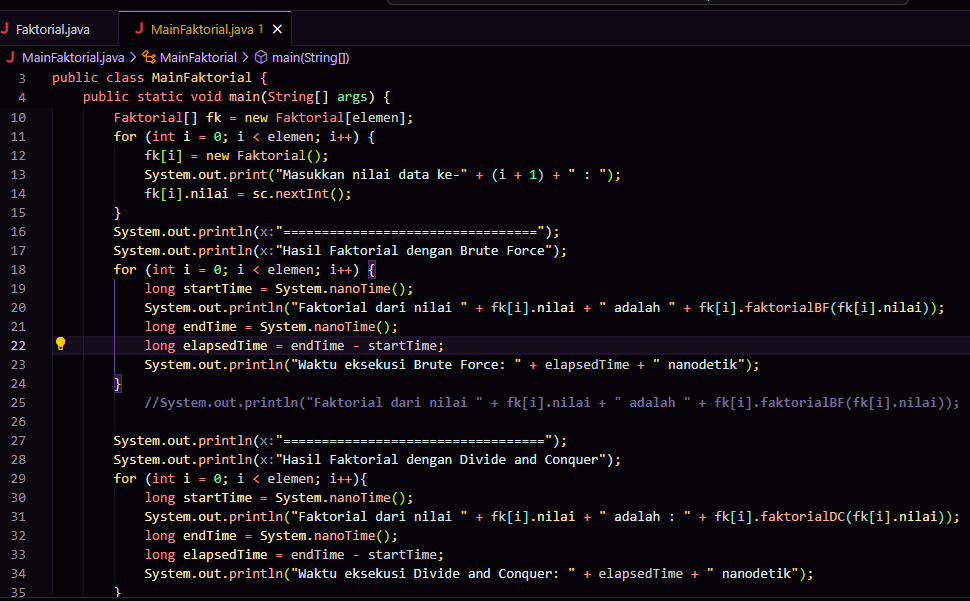
    Return Fakto;

}

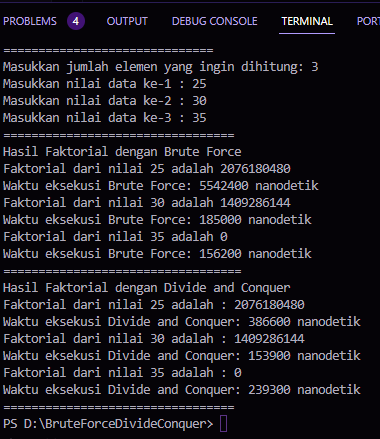
****

**Dalam Kode Itu, Perulangan While Digunakan Untuk Mengalikan Setiap Nilai Dari 1 Hingga N Dengan Fakto. Setiap Kali Loop Dieksekusi, Nilai Fakto Akan Dikalikan Dengan I, Dan I Akan Bertambah Satu Hingga Mencapai Nilai N. Ini Akan Menghasilkan Nilai Faktorial Dari N.**

1. Tambahkan Pegecekan Waktu Eksekusi Kedua Jenis Method Tersebut!



1. Buktikan Dengan Inputan Elemen Yang Di Atas 20 Angka, Apakah Ada Perbedaan Waktu Eksekusi?

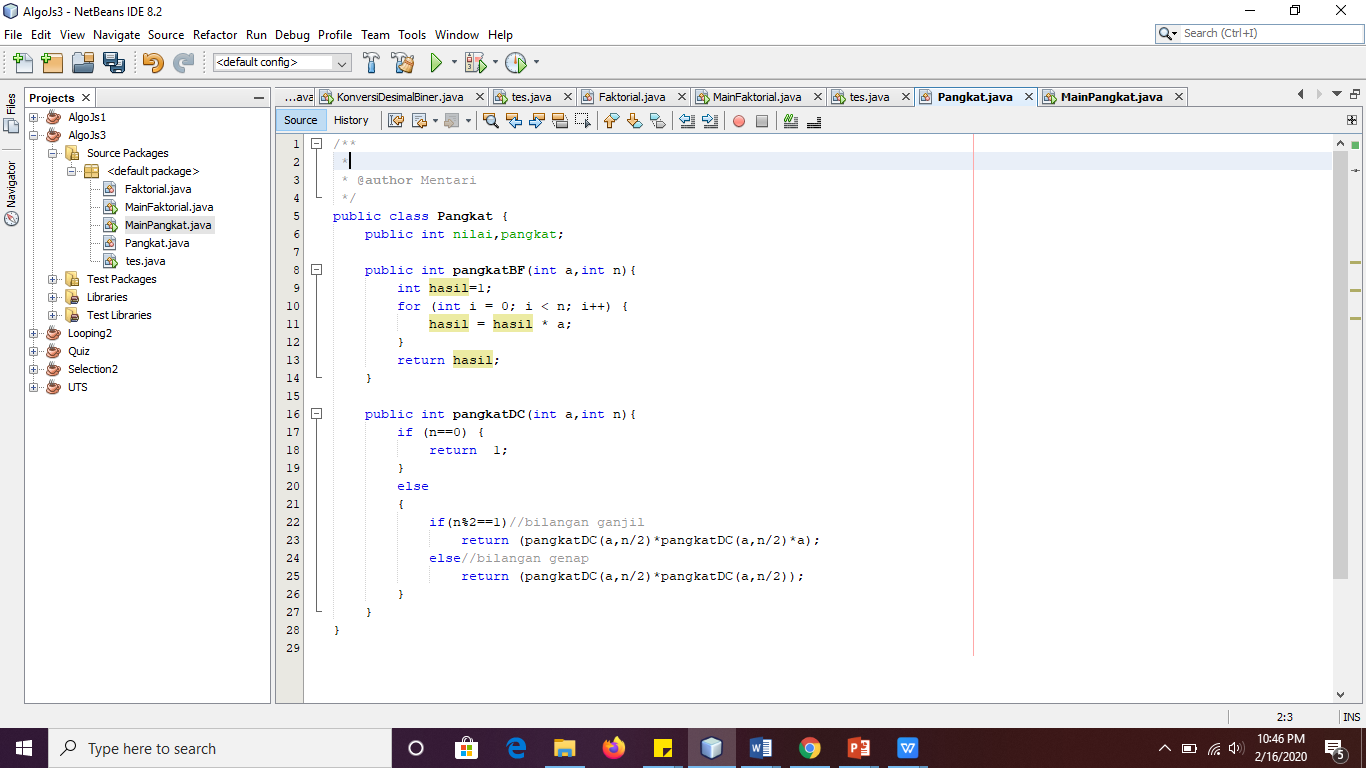


1. Menghitung Hasil Pangkat Dengan Algoritma Brute Force Dan Divide And Conquer

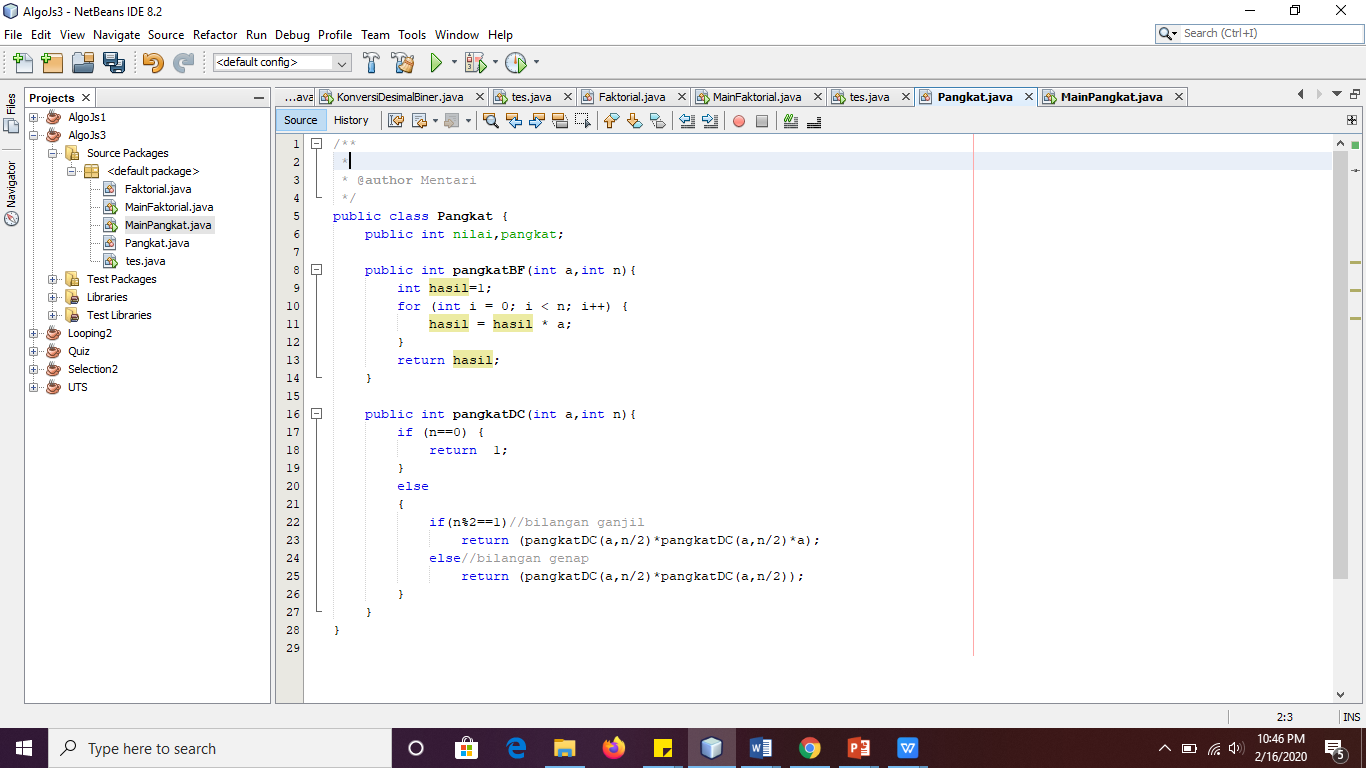
Pada Praktikum Ini Kita Akan Membuat Program Class Dalam Java. Untuk Menghitung Nilai Pangkat Suatu Angka Menggunakan 2 Jenis Algoritma, Brute Force Dan Divide And Conquer.

* + 1. Langkah-Langkah Percobaan

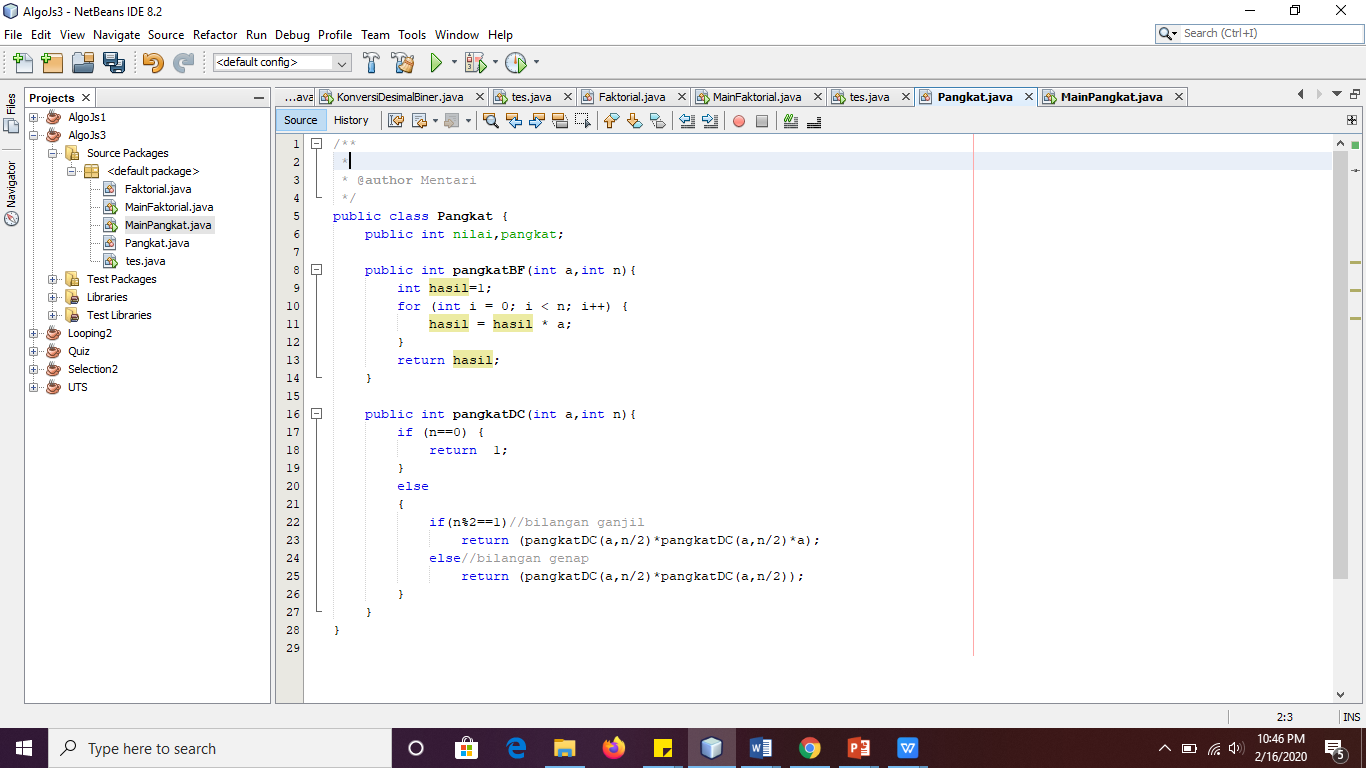
1. Di Dalam Paket **Minggu5**, Buatlah Class Baru Dengan Nama **Pangkat**. Dan Di Dalam Class **Pangkat** Tersebut, Buat Atribut Angka Yang Akan Dipangkatkan Sekaligus Dengan Angka Pemangkatnya



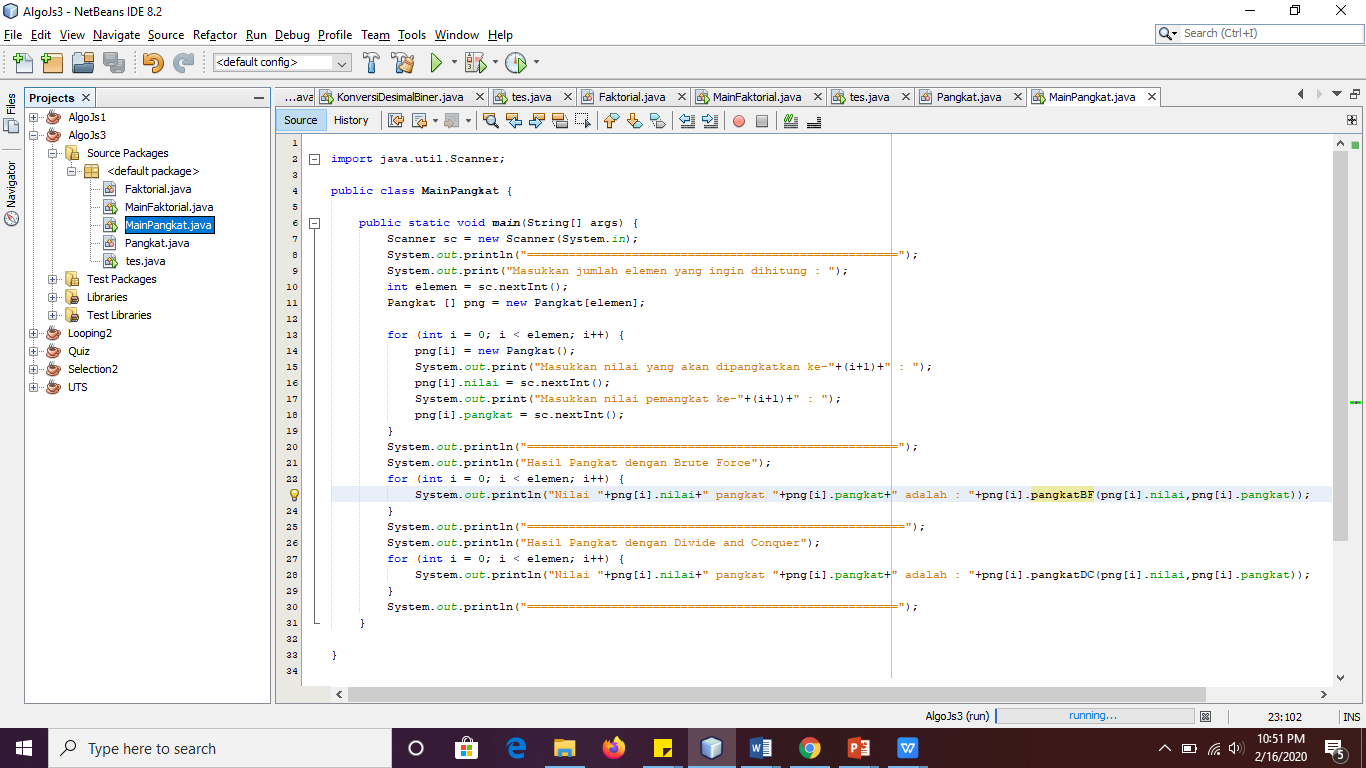
1. Pada Class Pangkat Tersebut, Tambahkan Method Pangkatbf()



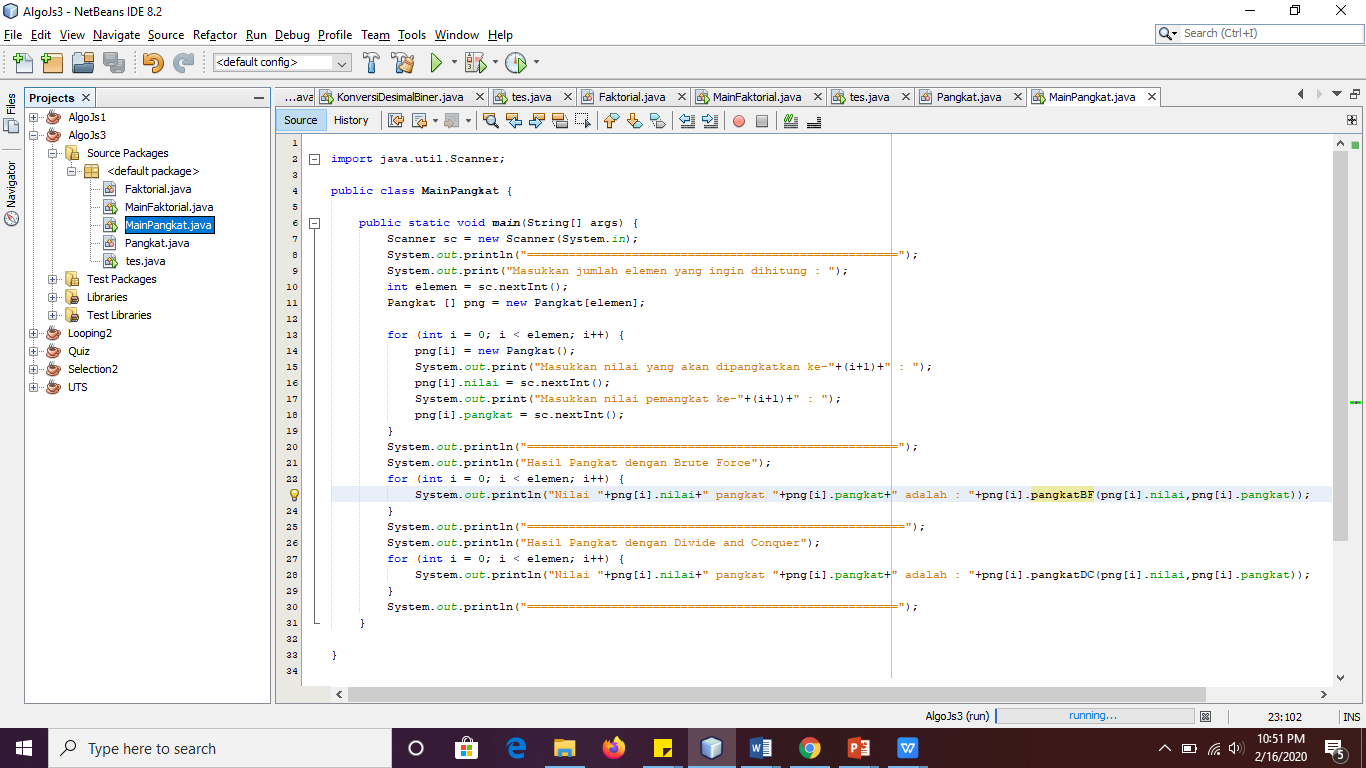
1. Pada Class Pangkat Juga Tambahkan Method Pangkatdc()



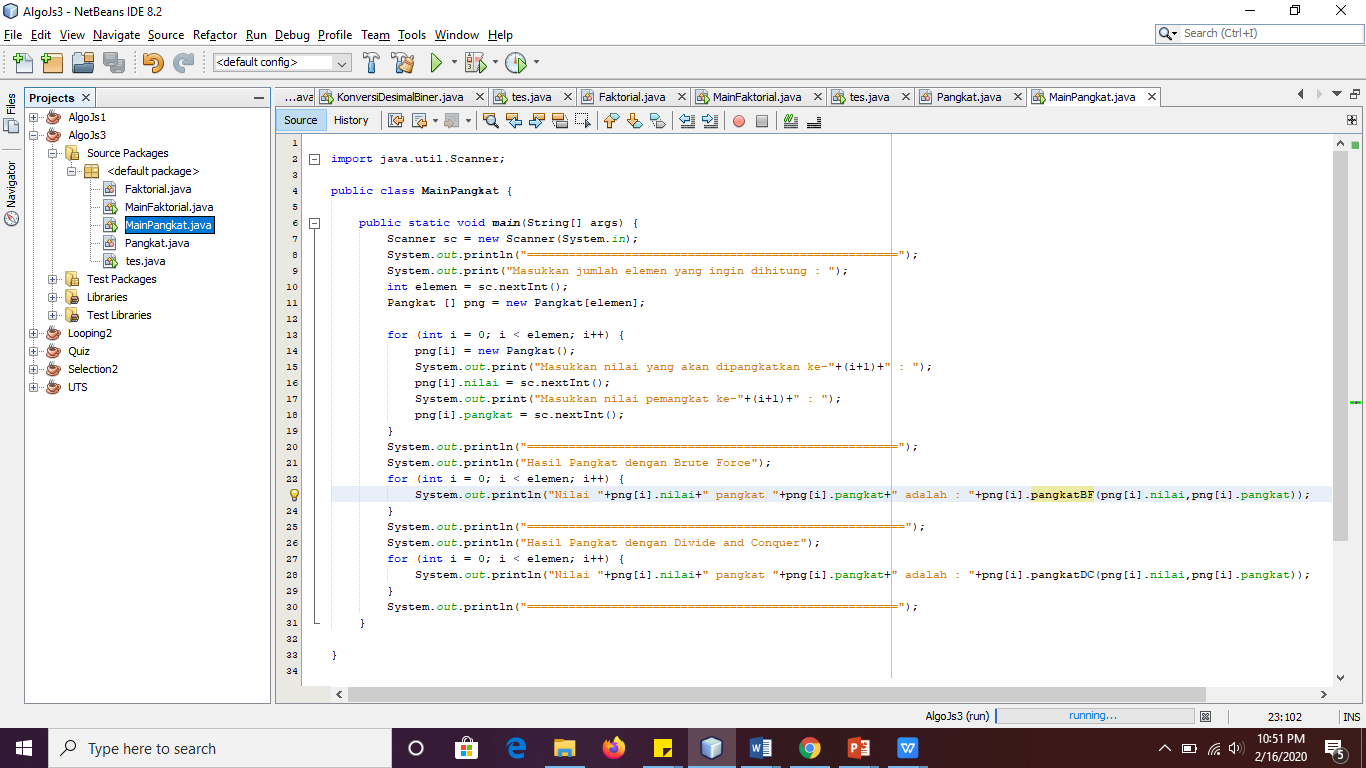
1. Perhatikan Apakah Sudah Tidak Ada Kesalahan Yang Muncul Dalam Pembuatan Class Pangkat
2. Selanjutnya Buat Class Baru Yang Di Dalamnya Terdapat Method Main. Class Tersebut Dapat Dinamakan Mainpangkat. Tambahkan Kode Pada Class Main Untuk Menginputkan Jumlah Nilai Yang Akan Dihitung Pangkatnya.



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.

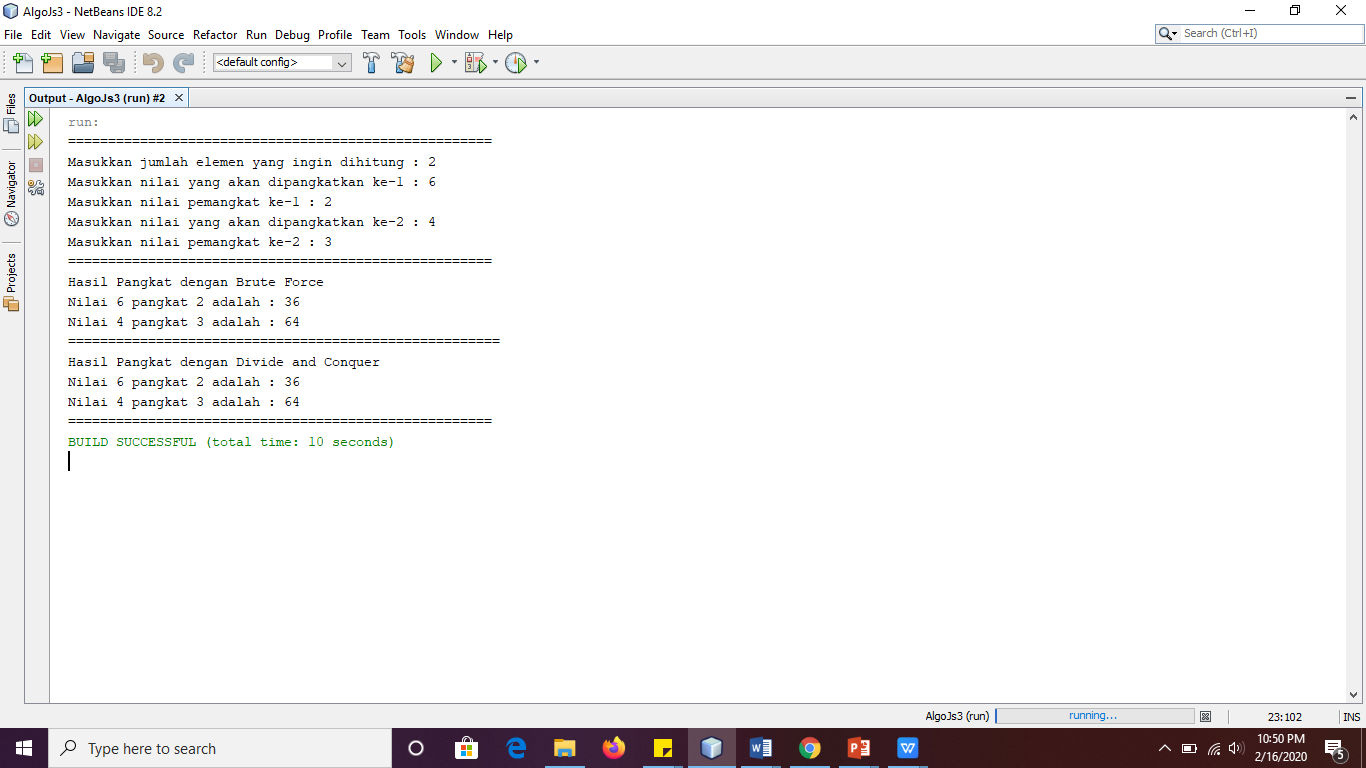


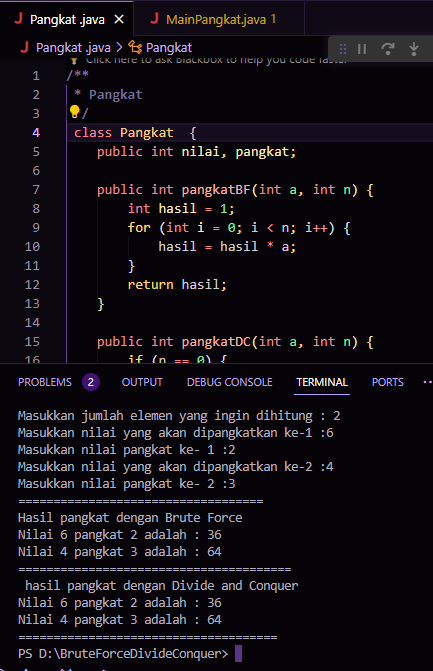
1. Kemudian, Panggil Hasil Nya Dengan Mengeluarkan Return Value Dari Method Pangkatbf() Dan Pangkatdc().



* + 1. Verifikasi Hasil Percobaan

Pastikan Output Yang Ditampilkan Sudah Benar Seperti Di Bawah Ini.





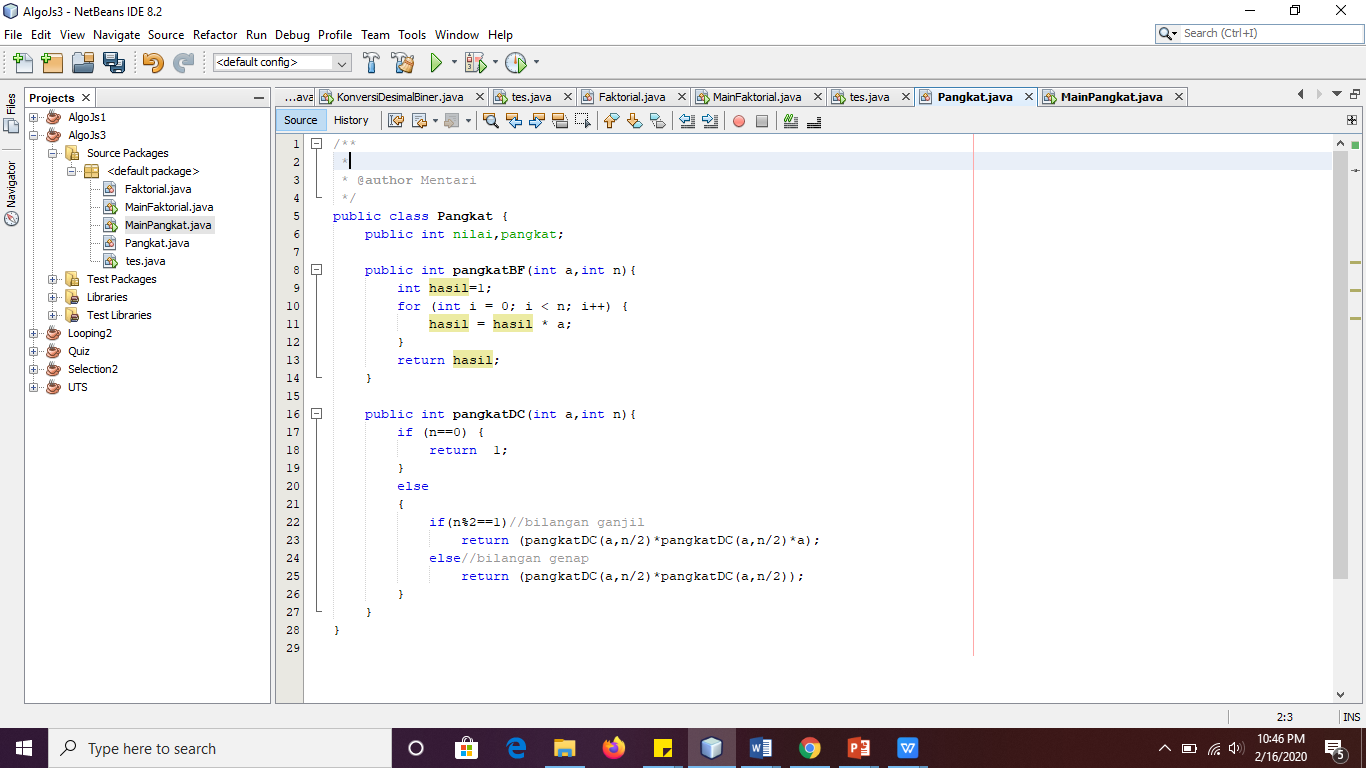
* + 1. Pertanyaan

1. Jelaskan Mengenai Perbedaan 2 Method Yang Dibuat Yaitu Pangkatbf() Dan Pangkatdc()!

* **Pangkatbf() Menggunakan Perulangan Untuk Mengalikan Angka Dengan Dirinya Sendiri Sebanyak N Kali.**
* **Pangkatdc() Menggunakan Pendekatan Rekursif Untuk Membagi Masalah Menjadi Submasalah Yang Lebih Kecil. Metode Ini Memangkatkan A Dengan Pangkat N/2 Dan Mengalikan Hasilnya.**

**Jika N Genap, Hasilnya Adalah Hasil Perkalian Pangkat A Dengan Pangkat N/2 Yang Dikuadratkan. Jika N Ganjil, Hasilnya Adalah Hasil Perkalian Pangkat A Dengan Pangkat (N-1)/2 Yang Dikuadratkan Dan Dikali A. Ini Lebih Efisien Untuk Pangkat Yang Besar Karena Mengurangi Jumlah Operasi Perkalian Yang Diperlukan.**

1. Pada Method Pangkatdc() Terdapat Potongan Program Sebagai Berikut:



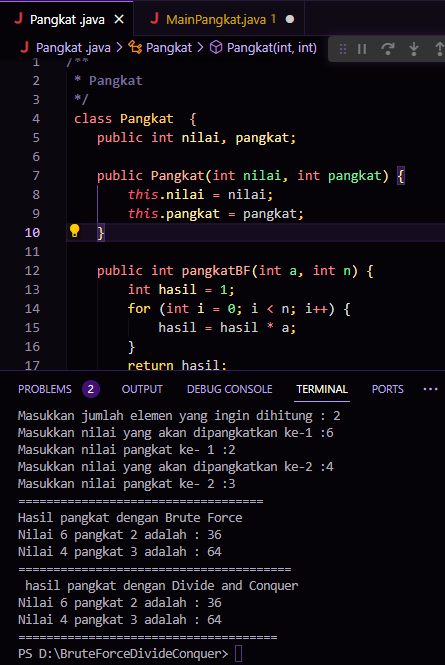
Jelaskan Arti Potongan Kode Tersebut

* **Jika Pangkat N Adalah Bilangan Ganjil, Maka Metode Pangkatdc() Akan Memangkatkan A Dengan Pangkat (N-1)/2 Dan Mengalikan Hasilnya Dengan Dirinya Sendiri (A) Untuk Memperoleh Hasil Akhir.**
* **Jika Pangkat N Adalah Bilangan Genap, Maka Metode Pangkatdc() Akan Memangkatkan A Dengan Pangkat N/2 Dan Mengalikan Hasilnya Dengan Dirinya Sendiri Untuk Memperoleh Hasil Akhir.**

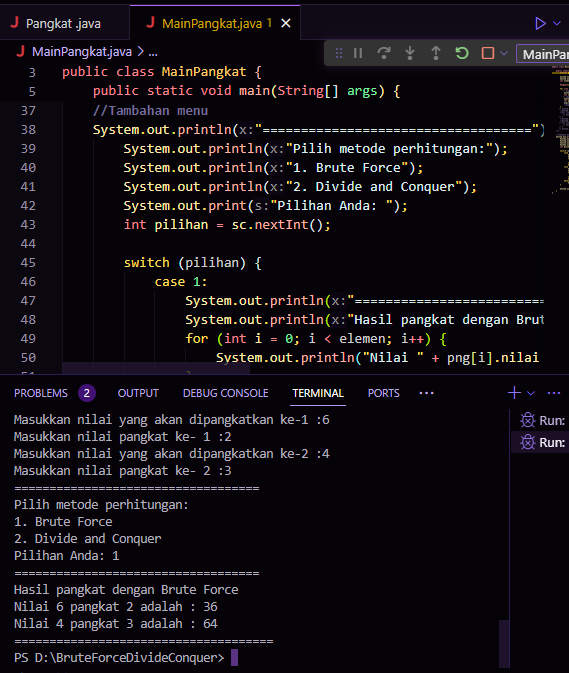
1. Apakah Tahap *Combine* Sudah Termasuk Dalam Kode Tersebut?Tunjukkan!

**Tidak, Tahap Combine Tidak Secara Eksplisit Terdapat Dalam Kode Pangkatdc(). Tahap Combine Secara Umum Merupakan Langkah Terakhir Dalam Algoritma Divide And Conquer Di Mana Solusi Dari Submasalah Digabungkan Untuk Menghasilkan Solusi Dari Masalah Asli.**

1. Modifikasi Kode Program Tersebut, Anggap Proses Pengisian Atribut Dilakukan Dengan Konstruktor.



1. Tambahkan Menu Agar Salah Satu Method Yang Terpilih Saja Yang Akan Dijalankan!

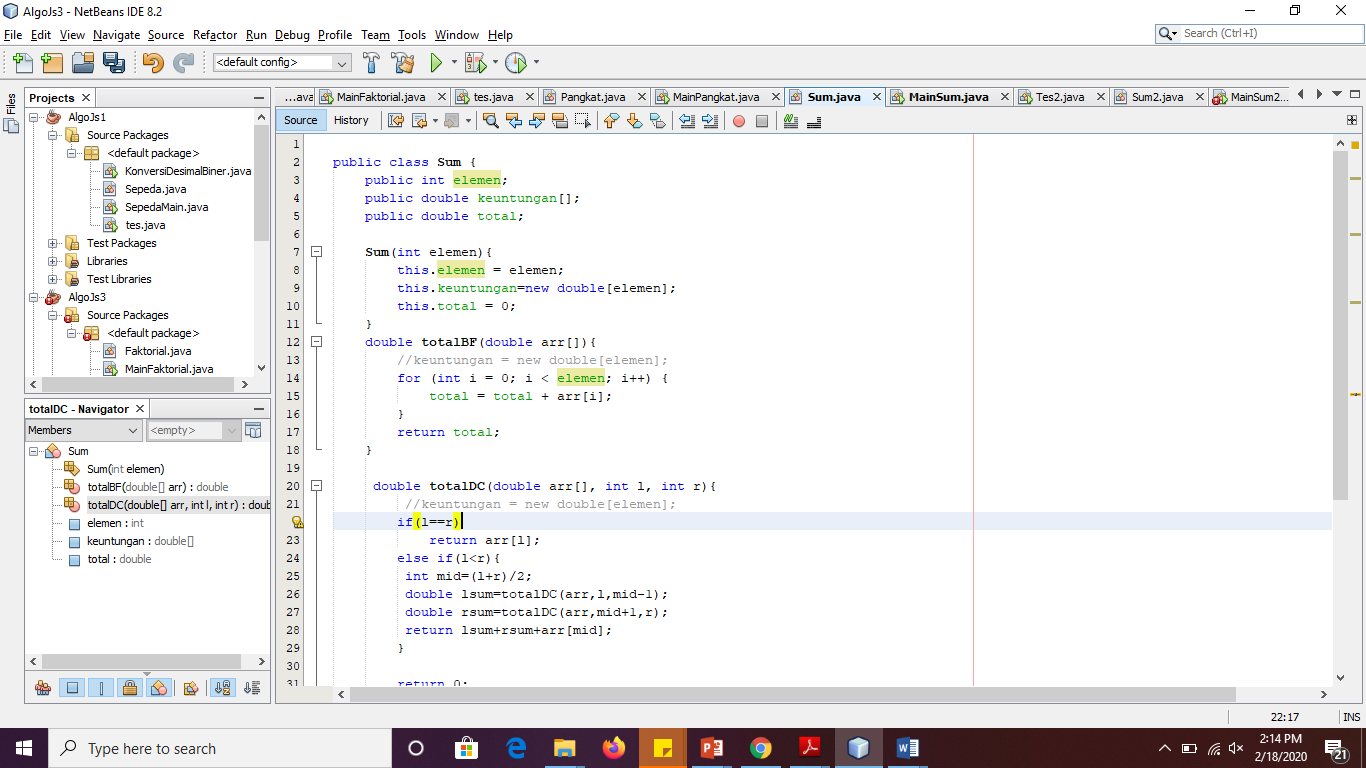


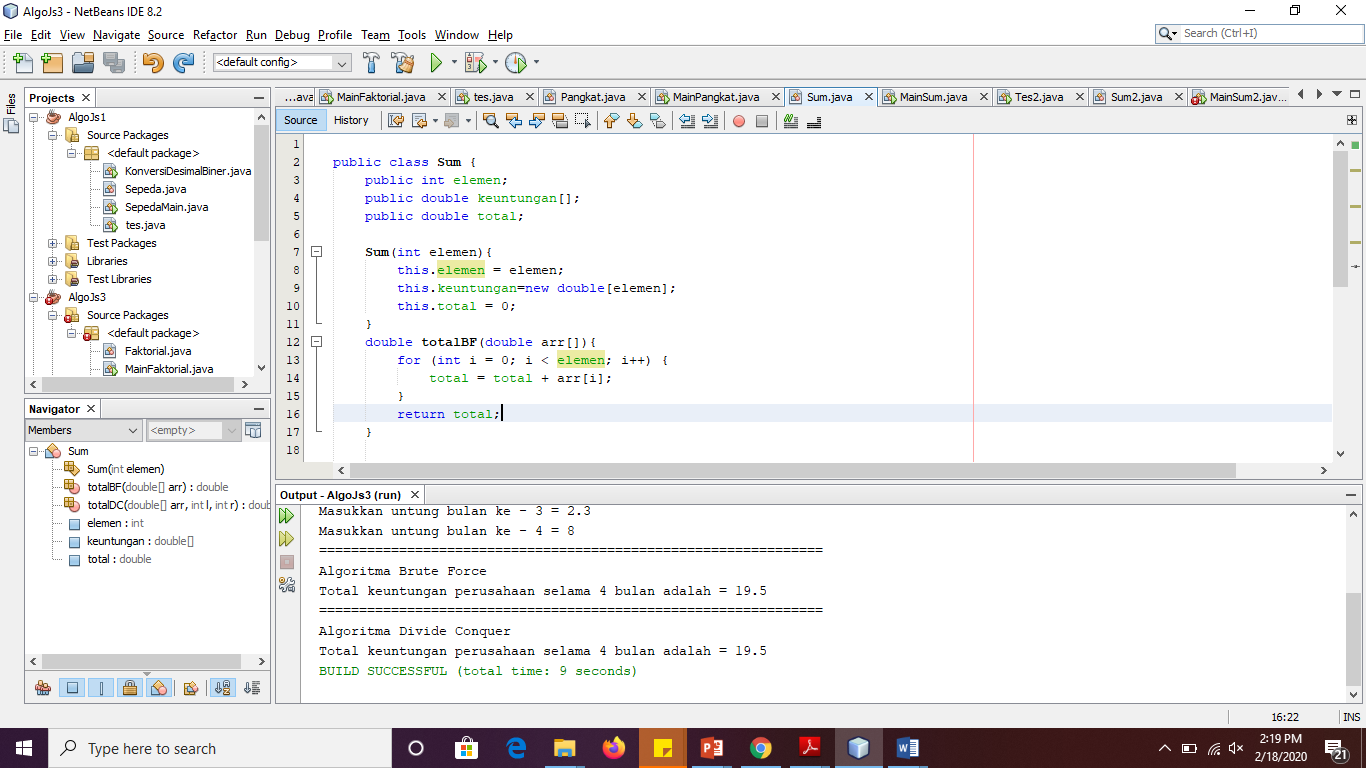
1. Menghitung Sum Array Dengan Algoritma Brute Force Dan Divide And Conquer

Di Dalam Percobaan Ini, Kita Akan Mempraktekkan Bagaimana Proses *Divide, Conquer*, Dan *Combine* Diterapkan Pada Studi Kasus Penjumlahan Keuntungan Suatu Perusahaan Dalam Beberapa Bulan.

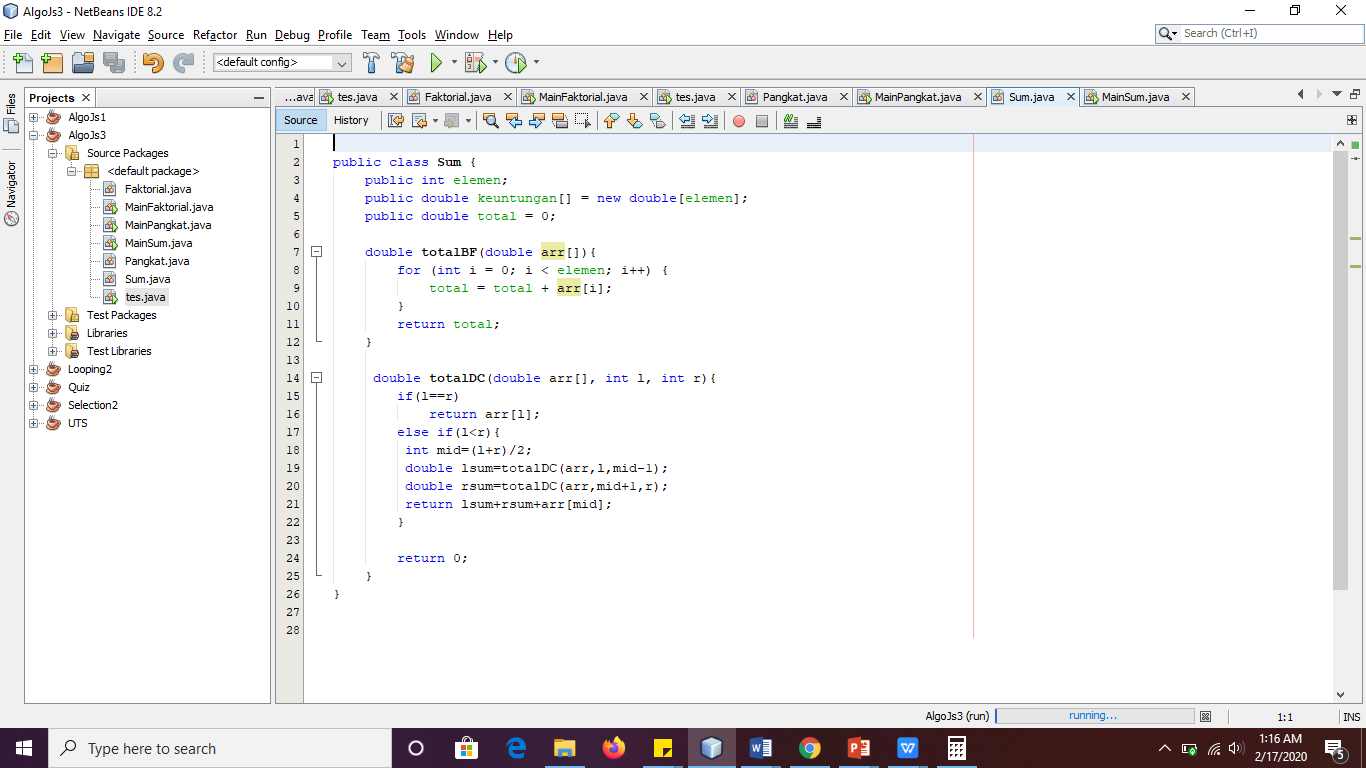
* + 1. Langkah-Langkah Percobaan

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.

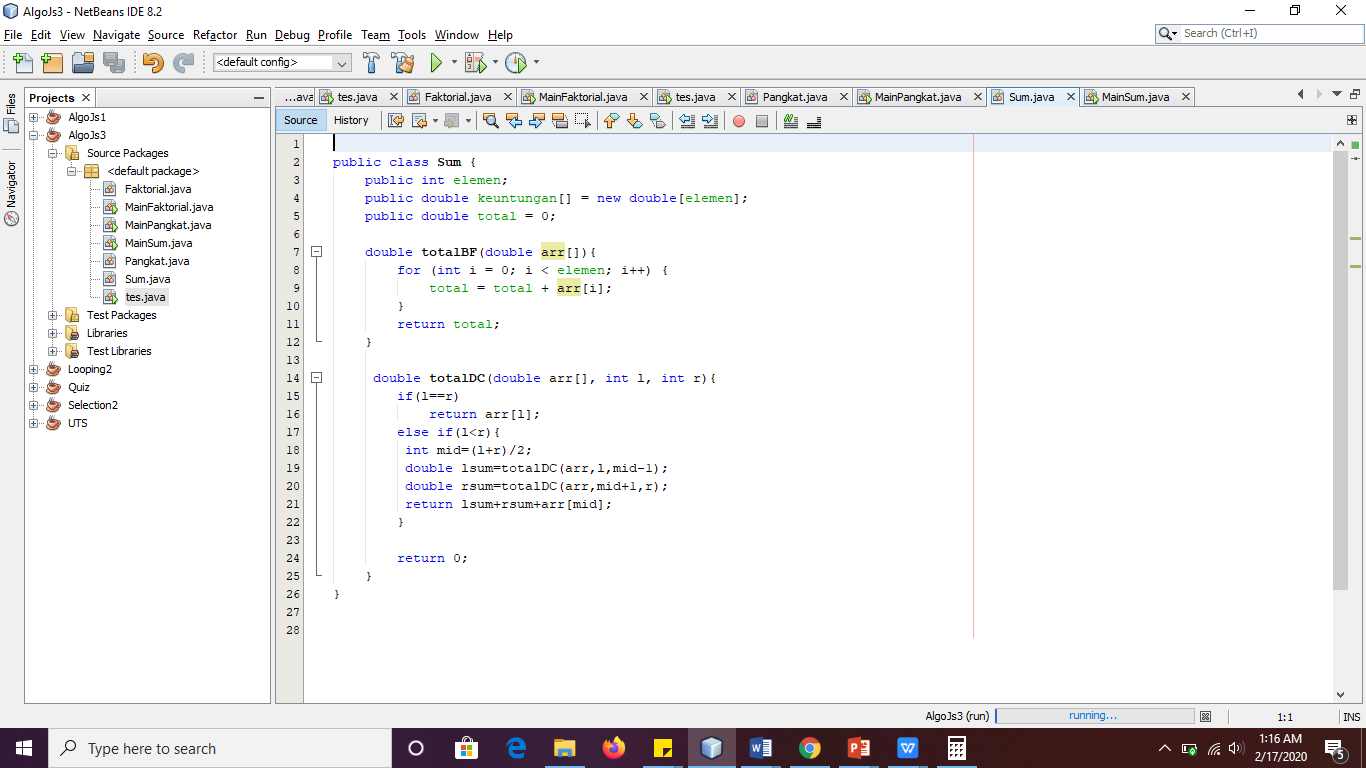




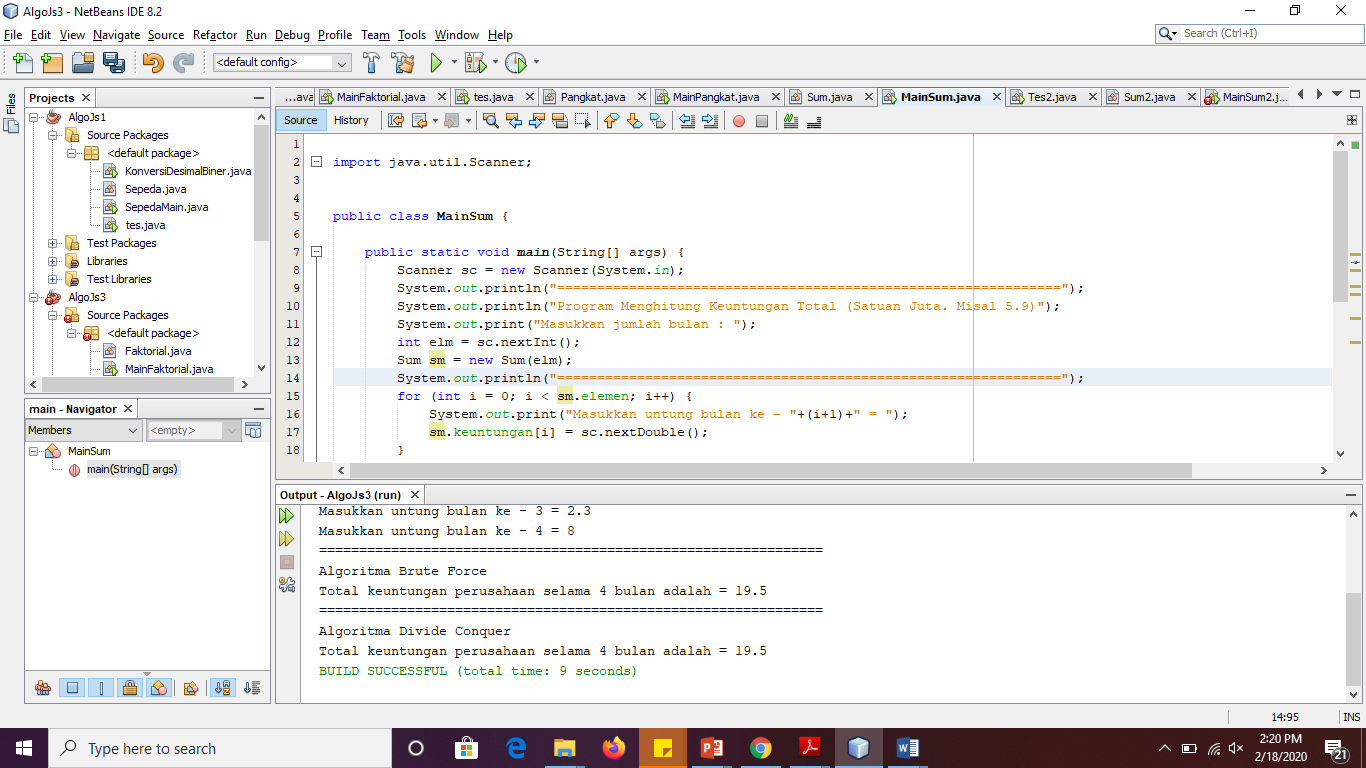
1. Tambahkan Method Totalbf() Yang Akan Menghitung Total Nilai Array Dengan Cara *Iterative*.



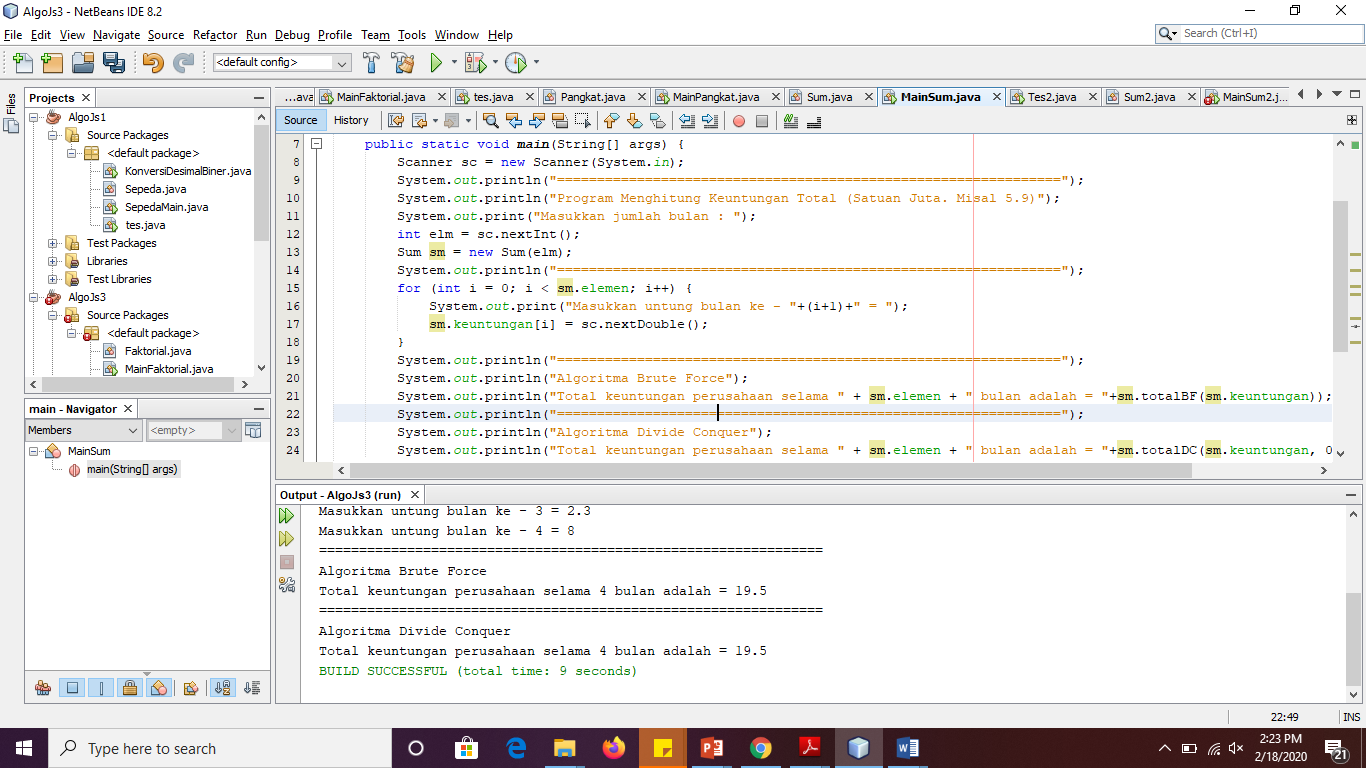
1. Tambahkan Pula Method Totaldc() Untuk Implementasi Perhitungan Nilai Total Array Menggunakan Algoritma Divide And Conquer



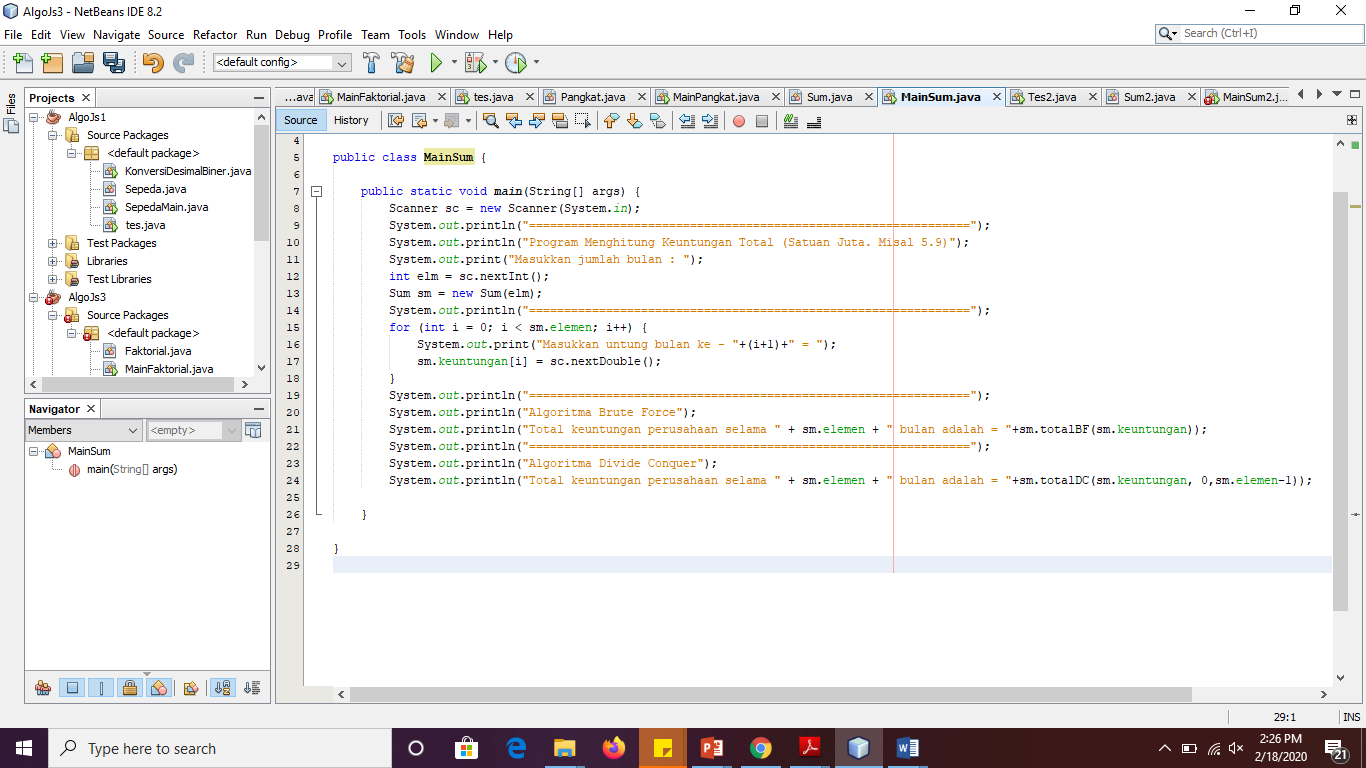
1. Buat Class Baru Yaitu Mainsum. Di Dalam Kelas Ini Terdapat Method Main. Pada Method Ini User Dapat Menuliskan Berapa Bulan Keuntungan Yang Akan Dihitung. Dalam Kelas Ini Sekaligus Dibuat Instansiasi Objek Untuk Memanggil Atribut Ataupun Fungsi Pada Class Sum



1. 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.

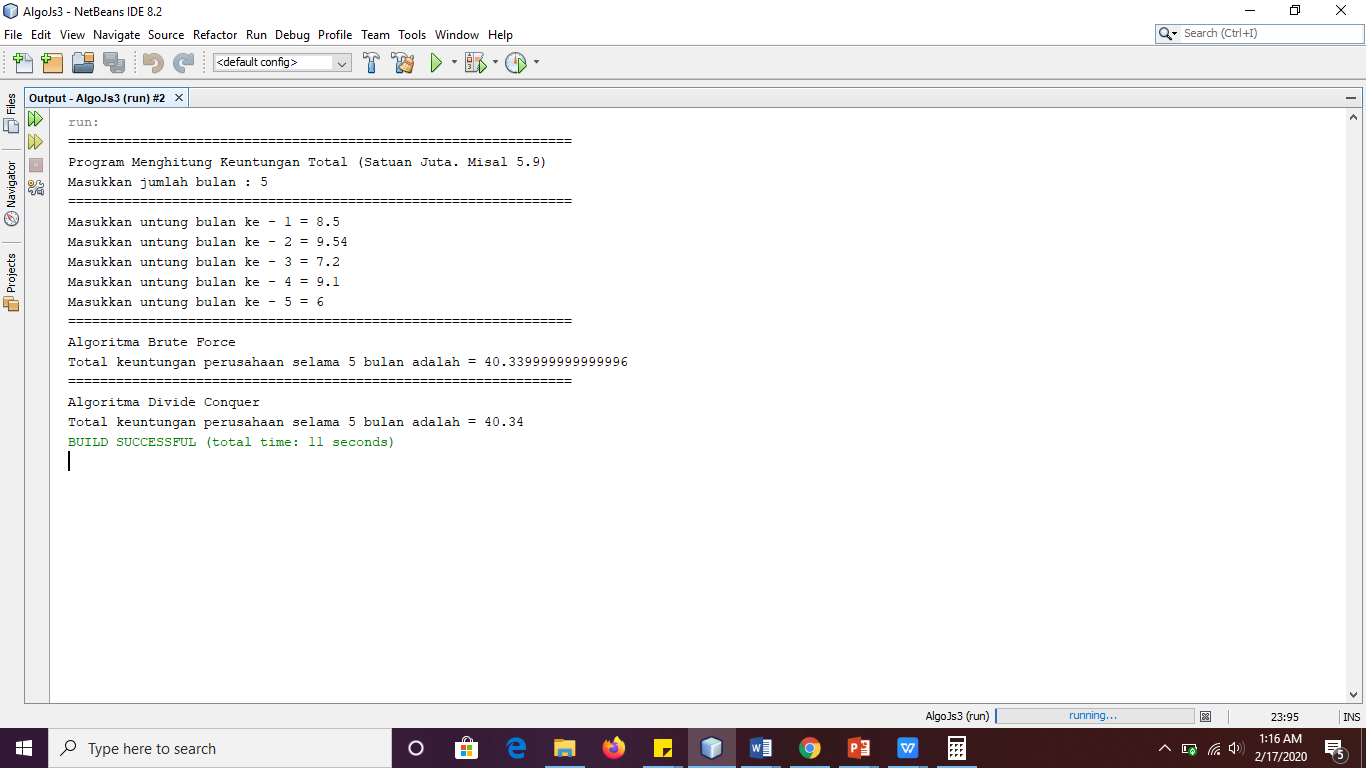


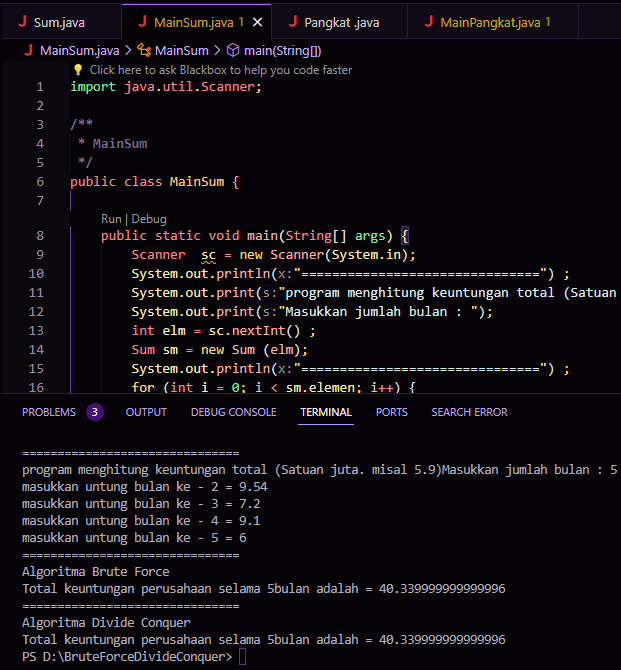
1. Tampilkan Hasil Perhitungan Melalui Objek Yang Telah Dibuat Untuk Kedua Cara Yang Ada (Brute Force Dan Divide And Conquer)



* + 1. Verifikasi Hasil Percobaan

Cocokkan Hasil Compile Kode Program Anda Dengan Gambar Berikut Ini.





* + 1. Pertanyaan

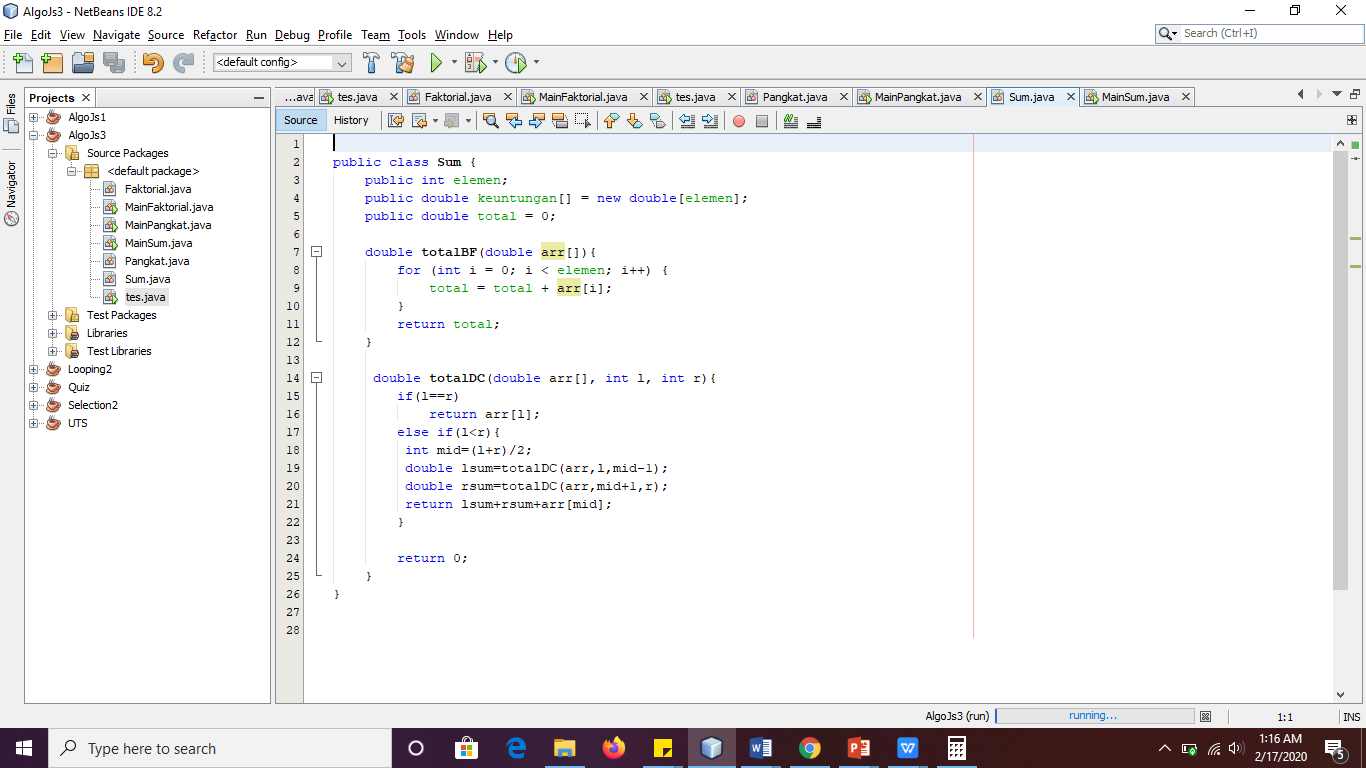
1. Berikan Ilustrasi Perbedaan Perhitungan Keuntungan Dengan Method Totalbf() Ataupun Totaldc()

* **Metode `Totalbf()` Langsung Menjumlahkan Semua Elemen Array Secara Berurutan. Contohnya, Jika Array Keuntungan Adalah [2, 3, 4, 5, 6], Maka Totalbf() Akan Menghitung 2 + 3 + 4 + 5 + 6 = 20.**
* **Metode `Totaldc()` Membagi Array Menjadi Dua Bagian, Menghitung Total Keuntungan Untuk Setiap Bagian, Lalu Menggabungkan Hasilnya. Misalnya, Dengan Array [2, 3, 4, 5, 6], Metode Totaldc() Akan Membaginya Menjadi [2, 3] Dan [4, 5, 6]. Kemudian, Jumlah Dari Kedua Bagian Tersebut Akan Digabungkan Untuk Mendapatkan Total Keuntungan.**

1. 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.

* **Untuk Membatasi Output Di Belakang Koma Agar Menjadi Standar, Anda Dapat Menggunakan Decimalformat Untuk Mengatur Jumlah Desimal Yang Ditampilkan.**

1. Mengapa Terdapat Formulasi *Return* *Value* Berikut?Jelaskan!

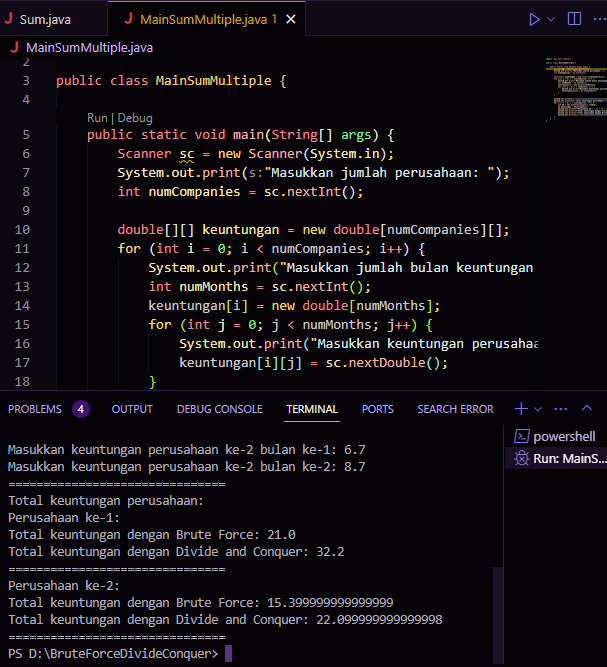


* **Pada Metode Totaldc(), Formulasi `Return Lsum + Rsum + Arr[Mid];` Digunakan Untuk Menggabungkan Hasil Dari Submasalah Yang Lebih Kecil Menjadi Hasil Akhir.**
* **`Lsum` Adalah Hasil Total Keuntungan Dari Subarray Kiri.**
* **`Rsum` Adalah Hasil Total Keuntungan Dari Subarray Kanan.**
* **`Arr[Mid]` Adalah Nilai Keuntungan Pada Bulan Tengah (Dalam Kasus Ini, Nilai Pada Indeks `Mid`).**
* **Jadi, Menggabungkan Total Keuntungan Dari Kedua Subarray Dengan Nilai Keuntungan Pada Bulan Tengah Untuk Mendapatkan Total Keuntungan Dari Seluruh Array.**

1. Kenapa Dibutuhkan Variable Mid Pada Method Totaldc()?

* **Variable Mid Digunakan Dalam Metode Totaldc() Untuk Menentukan Titik Tengah Saat Membagi Array Ke Dalam Dua Bagian. Dengan Mengetahui Indeks Tengah, Kita Dapat Membagi Array Menjadi Dua Bagian Yang Memiliki Jumlah Elemen Yang Seimbang Atau Mendekati Seimbang. Ini Penting Agar Pembagian Array Menjadi Dua Subarray Bisa Dilakukan Dengan Baik.**

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!



* 1. Latihan Praktikum

Buatlah Kode Program Untuk Menghitung Nilai Akar Dari Suatu Bilangan Dengan Algoritma Brute Force Dan Divide Conquer! *Jika Bilangan Tersebut Bukan Merupakan Kuadrat Sempurna, Bulatkan Angka Ke Bawah.*

