Jobsheet 4

Praktikum Algoritma & Struktur Data



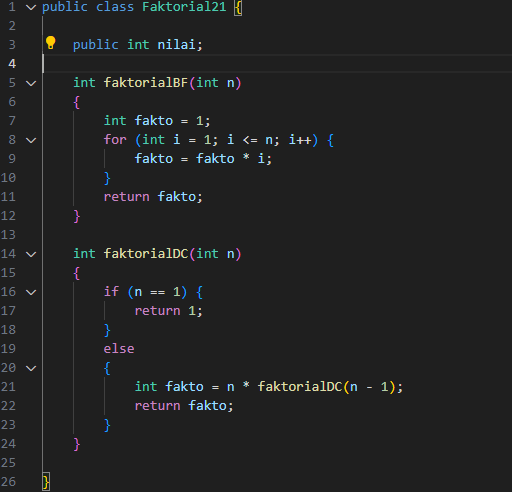
Rafi Ody Prasetyo  
(2341720180)

D-IV Teknik Informatika  
Politeknik Negeri Malang  
Semester 2  
2024

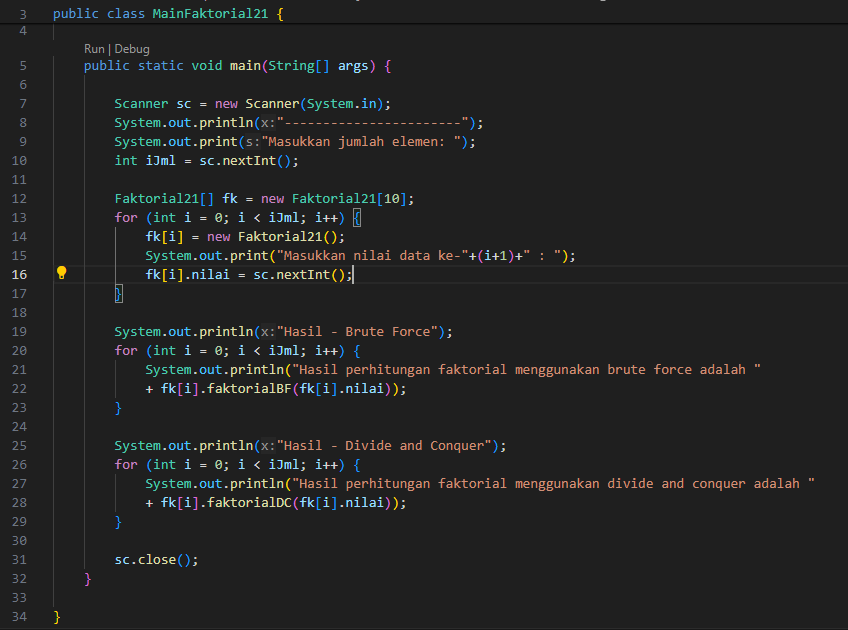
**Percobaan 1**

Code:

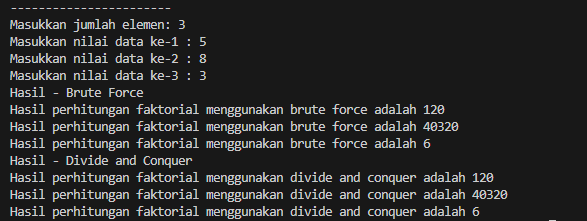
Faktorial21.java



MainFaktorial21.java



Output:



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:**

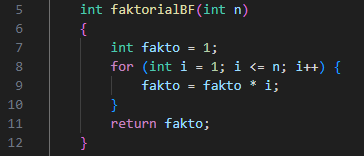
***if (n == 1) { return 1; }*** Ini adalah kondisi dasar atau base case dari algoritma rekursif. Ketika nilai n adalah 1, maka hasil faktorialnya adalah 1. Ini diperlukan agar rekursi berhenti dan tidak terjadi rekursi tak terbatas. ***Sedangkan else { int fakto = n \* faktorialDC(n - 1); return fakto; }*** 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.

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

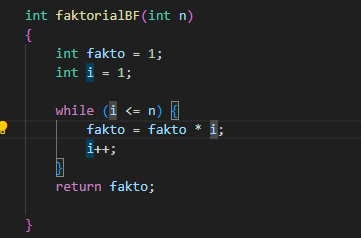
**Jawab:**

For dapat diubah dengan menggunakan perulangan WHILE.

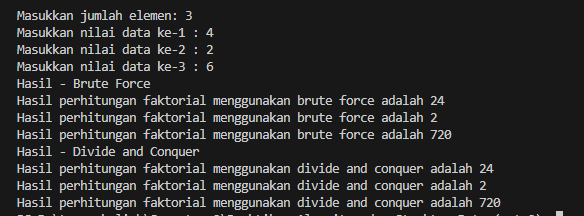
Before:



After:



Output:



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

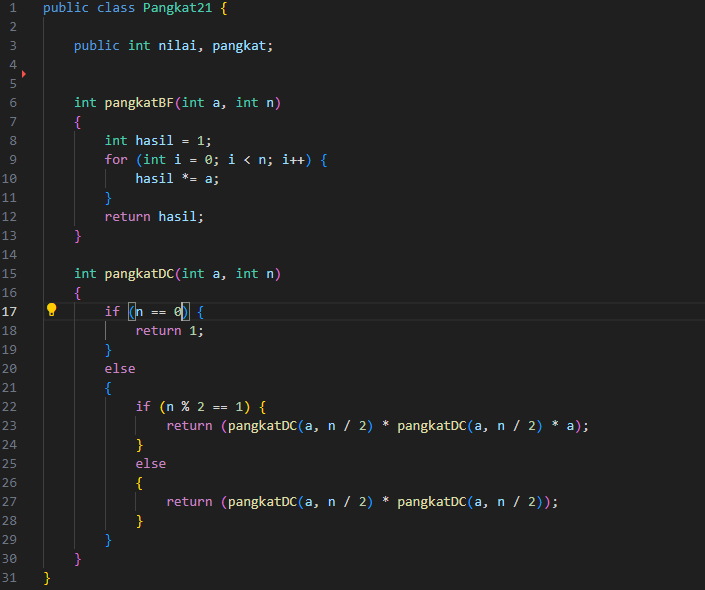
**Jawab:**

***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.

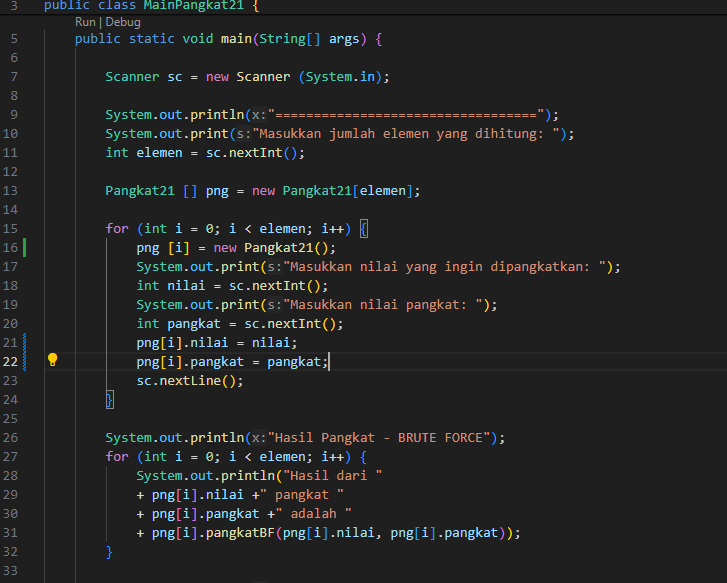
**Percobaan 2**

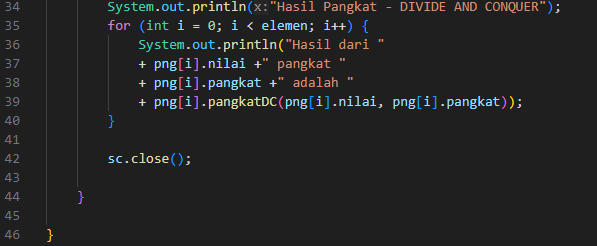
**Code:**

**Pangkat21.java**

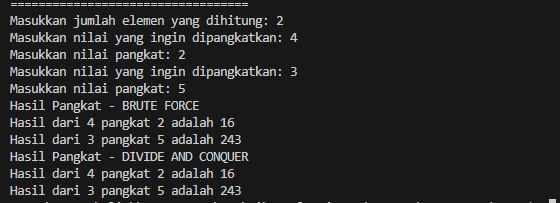
****

**MainPangkat21.java**

****

****

**Output:**

****

**Pertanyaan**

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

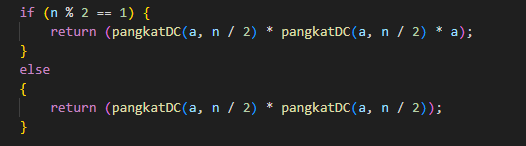
**Jawab:**

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)).

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

**Jawab:**

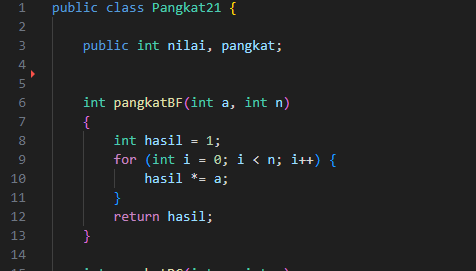
Tahap combine dalam algoritma divide-and-conquer (bagi dan taklukkan) menggabungkan solusi dari sub-masalah untuk menghasilkan solusi akhir. Pada kode pangkatDC, tahap combine terjadi di bagian else.



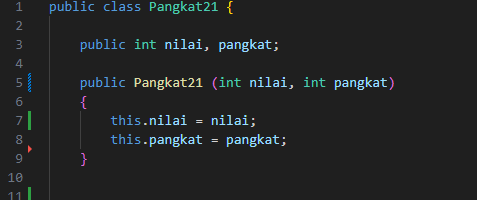
1. **Modifikasi kode program tersebut, anggap proses pengisian atribut dilakukan dengan konstruktor.**
2. **Tambahkan menu agar salah satu method yang terpilih saja yang akan dijalankan menggunakan switch-case!**

**Jawab:**

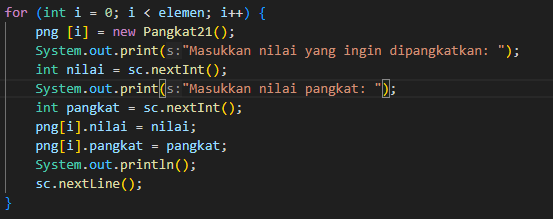
Before:



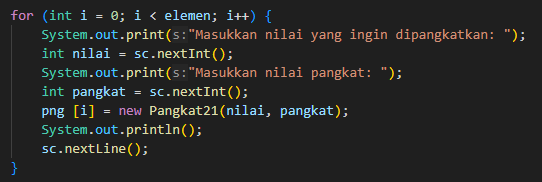
After:



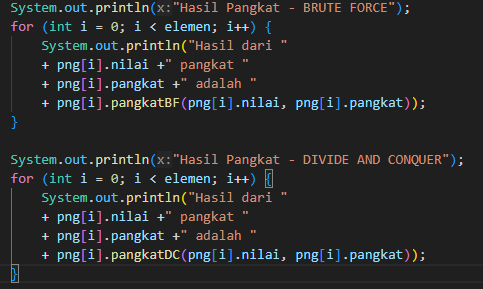
Before:



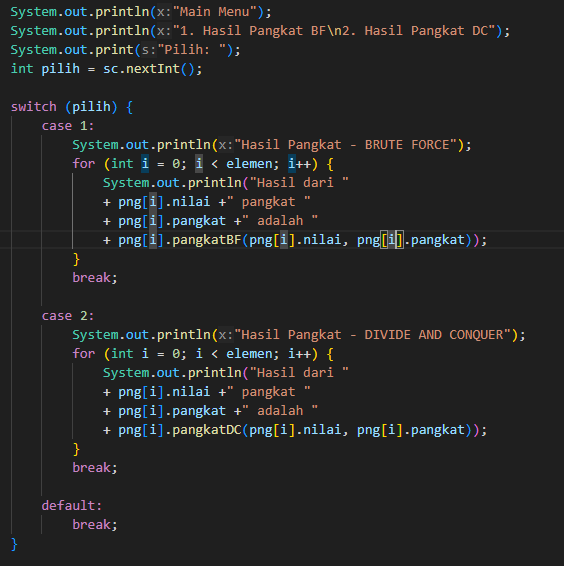
After:



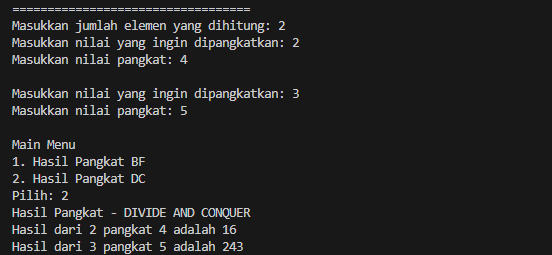
Before

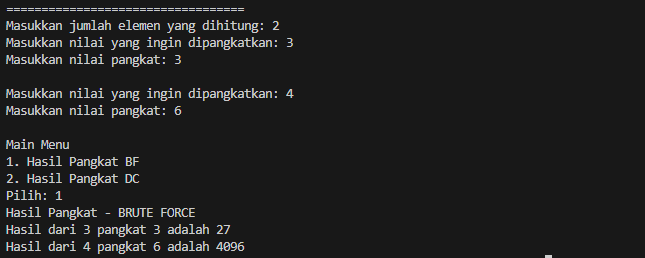


After



Output:

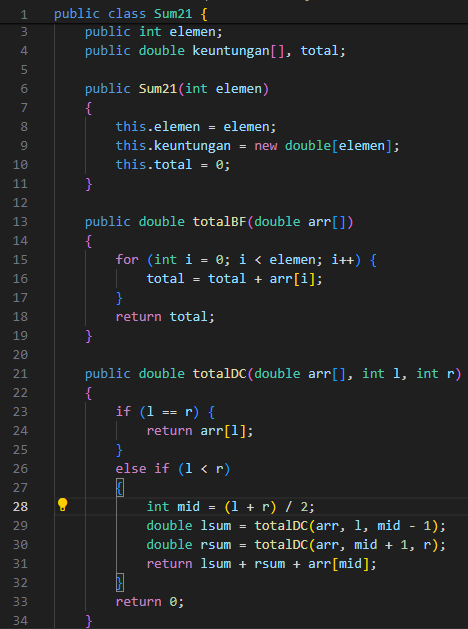




**Percobaan 3**

**Code:**

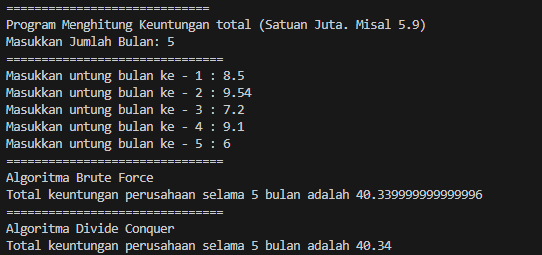
**Sum21.java**

****

**MainSum21.java**

****

**Output:**

****