Laporan Praktikum ASD

Jobsheet 5

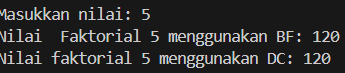


Nama : Bagas Ardiansa Putra

NIM : 244107020166

Kelas : TI – 1B

Percobaan 1

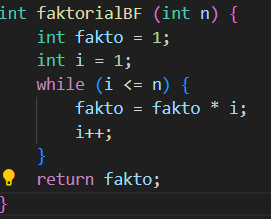


Pertanyaan:

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

* If = digunakan untuk basis kasus (base case)
* Else = digunakan untuk rekursif kasus (recursive case) yang biasanya jika di base case nilai tidak terpenuhi akan masuk bagian else

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

Bisa diubah yaitu menggunakan while 

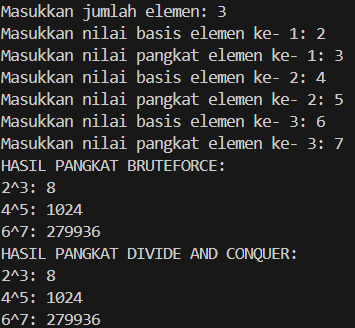
1. Jelaskan perbedaan antara fakto \*= i; dan int fakto = n \* faktorialDC(n-1); !

* fakto \*= i digunakan dalam metode iteratif (faktorialBF) yang berbasis perulangan (for/while)
* int fakto = n \* faktorialDC(n-1); digunakan dalam metode rekursif (faktorialDC), berfungsi memanggil dirinya sendiri untuk menghitung nilai factorial

1. Buat Kesimpulan tentang perbedaan cara kerja method faktorialBF() dan faktorialDC()!

faktorialBF() lebih efisien untuk menghitung factorial karena metode ini menggunakan perulangan yang bisa mempercepat perhitungan dan faktorialDC() memecah masalah menjadi sub yang kecil untuk menyelesaikan nya, menggunakan pemanggilan fungsi rekursif.

Percobaan 2

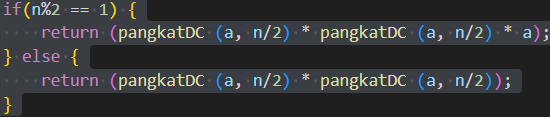


Pertanyaan:

1. Jelaskan mengenai perbedaan 2 method yang dibuat yaitu pangkatBF() dan pangkatDC()!

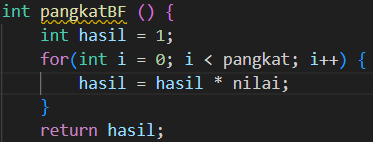
* pangkatBF() menggunakan perulangan for untuk mengalikan basis(a) sebanyak n kali.
* pangkatDC() menggunakan rekursi dan divide dan conquer untuk menghitung pangkat dengan efisien.

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

Sudah ada pada kode pangkatDC() 

1. Pada method pangkatBF()terdapat parameter untuk melewatkan nilai yang akan dipangkatkan dan pangkat berapa, padahal di sisi lain di class Pangkat telah ada atribut nilai dan pangkat, apakah menurut Anda method tersebut tetap relevan untuk memiliki parameter? Apakah bisa jika method tersebut dibuat dengan tanpa parameter? Jika bisa, seperti apa method pangkatBF() yang tanpa parameter?

Akan tetap relevan jika digunakan secara status yang tidak bergantung atribut objek parameter masih diperlukan. Method ini bisa dibuat tanpa parameter



1. Tarik tentang cara kerja method pangkatBF() dan pangkatDC()!

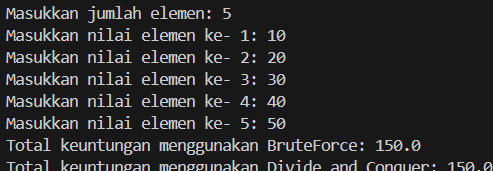
Metode pangkatBF() memakai perulangan untuk menghitung hasil pangkat bertahap. Langkah-langkah kerja:

* Inisialisasi variabel hasil = 1.
* Menggunakan perulangan (for) sebanyak n kali.
* Setiap iterasi, mengalikan hasil dengan a.
* Setelah perulangan selesai, mengembalikan hasil (return).

Metode pangkatDC() menggunakan rekursi dan strategi Divide and Conquer untuk menghitung pangkat. Langkah-langkah kerja:

* Basis kasus: Jika n == 1, kembalikan a karena a^1 = a.
* Divide: Memecah masalah menjadi sub-masalah lebih kecil dengan n / 2.
* Conquer: Memanggil metode pangkatDC() secara rekursif hingga mencapai basis kasus (base case).

Percobaan 3



Pertanyaan:

1. Kenapa dibutuhkan variable mid pada method TotalDC()?

Karena variable mid berguna untuk membagi masalah menjadi dua bagian lebih kecil dan bisa diproses secara rekursif.

1. Untuk apakah statement di bawah ini dilakukan dalam TotalDC()?

Statement ini bagian dari divide dan conquer yang berfungsi sebagai memecah masalah ke sub masalah lebih kecil menggunakan rekursi.

1. Kenapa diperlukan penjumlahan hasil lsum dan rsum seperti di bawah ini?

Dalam metode totalDC() berfungsi untuk menggabungjan hasil dari dua bagian array yang diproses menggunakan divide dan conquer, diperlukan karena agar hasil total dapat dihitung dari dua hasil bagian kecil.

1. Apakah base case dari totalDC()?

Base case adalah kondisi yang menghentikan rekursi agar tidak berjalan teru menerus, untuk metode totalDC() ini terjadi hanya ada satu elemen yang tersisa dalam array 

1. Tarik Kesimpulan tentang cara kerja totalDC()

* totalDC() membagi array menjadi dua bagian, menyelesaikan tiap bagian secara rekursif, lalu menggabungkan hasilnya. (mid = (l + r) / 2)  
  Base case (l == r) mencegah rekursi berjalan tanpa batas dan mengembalikan nilai saat hanya ada satu elemen.
* Combine (return lsum + rsum;)
* Setelah mendapatkan hasil total dari bagian kiri (lsum) dan bagian kanan (rsum).