

LAPORAN HASIL PRAKTIKUM

Algoritma dan Struktur Data

Jobsheet 5



Alexsa Fitria Ayu Siswoyo.

244107020020

1E

Program Studi Teknik Informatika

Jurusan Teknologi Informasi

Politeknik Negeri Malang

2025

PERCOBAAN 1 Menghitung Nilai Faktorial dengan Algoritma Brute Force dan Divide and Conquer

Langkah-langkah

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:
 - a. Tambahkan method faktorialBF():
 - b. Tambahkan method faktorialDC():

```
package jobsheet5;

public class Faktorial02 {
    int faktorialBF;
    int faktorialDC;

    public int faktorialBF(int n){
        int fakto = 1;
        for(int i=1; i<=n; i++){
            fakto = fakto * i;
        }
        return fakto;
    }

    public int faktorialDC(int n){
        if(n==1){
            return 1;
        } else{
            int fakto = n *
faktorialDC(n-1);
            return fakto;
        }
    }
}
```

4. Coba jalankan (Run) class Faktorial dengan membuat class baru MainFaktorial.
 - a. Di dalam fungsi main sediakan komunikasi dengan user untuk memasukkan nilai yang akan dicari faktorialnya
 - b. Kemudian buat objek dari class Faktorial dan tampilkan hasil pemanggilan method

```

package jobsheet5;

import java.util.Scanner;

public class MainFaktorial02 {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Masukkan nilai: ");
        int nilai = input.nextInt();

        Faktorial02 fk = new Faktorial02();
        System.out.println("Nilai faktorial
"+nilai+"menggunakan BF: " +fk.faktorialBF(nilai));
        System.out.println("Nilai faktorial
"+nilai+"menggunakan DC: " +fk.faktorialDC(nilai));

        input.close();
    }
}

```

```

SD_6ab0b5a7\bin' 'jobsheet5.MainFaktorial02'
Masukkan nilai: 5
Nilai faktorial 5menggunakan BF: 120
Nilai faktorial 5menggunakan DC: 120
PS C:\Users\User\OneDrive\KAMPUS\KELAS\1E\SEM 2\Praktikum-ASD>

```

PERTANYAAN

1. Pada base line Algoritma Divide Conquer untuk melakukan pencarian nilai faktorial, jelaskan perbedaan bagian kode pada penggunaan if dan else!
 - If (base case) menghentikan rekursi saat $n == 0$ atau 1, langsung return 1
 - Else (recursive case) memanggil fungsi Kembali($n * \text{fatorialDC}(n-1)$)
2. Apakah memungkinkan perulangan pada method faktorialBF() diubah selain menggunakan for? Buktikan!

```

public int faktorialBF(int n) {
    int fakto = 1;
    int i = 1;
    while (i <= n) {
        fakto = fakto * i;
        i++;
    }
    return fakto;
}

```

3. Jelaskan perbedaan antara fakto *= i; dan int fakto = n * faktorialDC(n-1); !
 - Fakto *= i; → memperbarui nilai yang sudah ada di tiap langkah perulangan
 - n * faktorialDC(n-1); → proses perhitungan dilakukan lewat pemanggilan fungsi secara bertahap ke nilai yang lebih kecil
4. Buat Kesimpulan tentang perbedaan cara kerja method faktorialBF() dan faktorialDC()!
 - a. faktorialBF() menggunakan pendekatan iteratif, di mana faktorial dihitung melalui perulangan (*looping*).
 - b. faktorialDC() menggunakan pendekatan rekursif, di mana fungsi memanggil dirinya sendiri hingga mencapai kondisi dasar (*base case*).

PERCOBAAN 2 Menghitung Hasil Pangkat dengan Algoritma Brute Force dan Divide and Conquer.

Langkah-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. Membuat atribut angka yang akan dipangkatkan sekaligus dengan angka pangkatannya

```
int nilai,
```

2. Tambahkan konstruktor berparameter

```
Pangkat02(int n, int p) {
    nilai = n;;
    pangkat = p;
}
```

3. Pada class pangkat03 tambahkan method dengan menambahkan method pangkatBF();

```
int pangkatBF(int a, int n) {
    int hasil = 1;
    for(int i = 0; i < n; i++){
        hasil = hasil * a;
    }
    return hasil;
}
```

4. Tambahkan method pangkatDC();

```
int pangkatDC(int a, int n) {
    if (n==1) {
        return a;
    }else{
        if (n%2==1) {
            return (pangkatDC(a, n/2) * pangkatDC(a,
n/2) *a);
        }else{
            return (pangkatDC(a, n/2) * pangkatDC(a,
n/2));
        }
    }
}
```

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

```
Scanner input = new Scanner(System.in);
System.out.print("Masukkan jumlah elemen: ");
int elemen = input.nextInt();
```

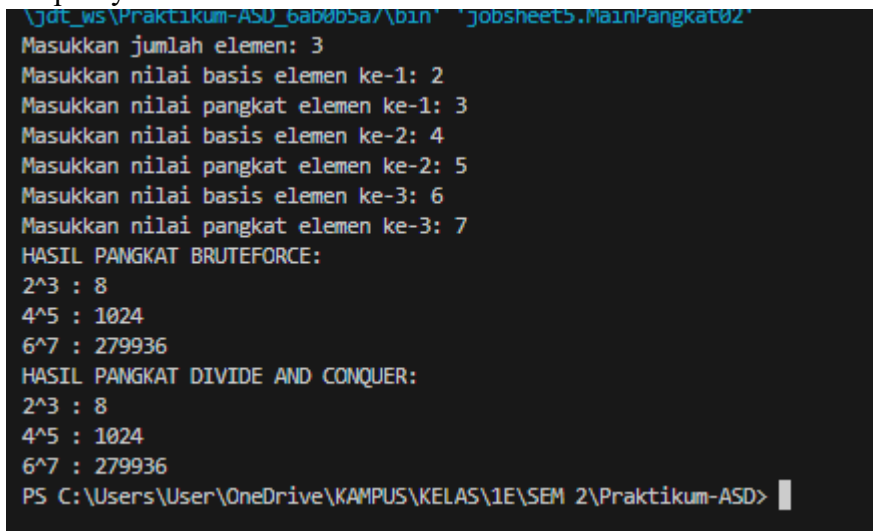
7. Nilai pada tahap 5 selanjutnya digunakan untuk instansiasi array of objek. Di dalam kode berikut ditambahkan proses pengisian beberapa nilai yang akan dipangkatkan sekaligus dengan pemangkatnya.

```
Pangkat02[] png = new Pangkat02[elemen];
for (int i = 0; i < elemen; i++) {
    System.out.print("Masukkan nilai basis elemen
ke-" + (i + 1) + ": ");
    int basis = input.nextInt();
    System.out.print("Masukkan nilai pangkat elemen
ke-" + (i + 1) + ": ");
    int pangkat = input.nextInt();
    png[i] = new Pangkat02(basis, pangkat);
}
```

8. Panggil hasil dengan mengeluarkan return value dari method pangkatBF() dan pangkatDC()

```
System.out.println("HASIL PANGKAT BRUTEFORCE:");
for (Pangkat02 p : png) {
    System.out.println(p.nilai + "^" + p.pangkat + "
: " + p.pangkatBF(p.nilai, p.pangkat));
}
System.out.println("HASIL PANGKAT DIVIDE AND
CONQUER:");
for (Pangkat02 p : png) {
    System.out.println(p.nilai + "^" + p.pangkat + "
: " + p.pangkatDC(p.nilai, p.pangkat));
}
```

Outputnya



```
\\dot_ws\Praktikum-ASD_6ab0b5a7\bin\ "jobsheet5.MainPangkat02"
Masukkan jumlah elemen: 3
Masukkan nilai basis elemen ke-1: 2
Masukkan nilai pangkat elemen ke-1: 3
Masukkan nilai basis elemen ke-2: 4
Masukkan nilai pangkat elemen ke-2: 5
Masukkan nilai basis elemen ke-3: 6
Masukkan nilai pangkat elemen ke-3: 7
HASIL PANGKAT BRUTEFORCE:
2^3 : 8
4^5 : 1024
6^7 : 279936
HASIL PANGKAT DIVIDE AND CONQUER:
2^3 : 8
4^5 : 1024
6^7 : 279936
PS C:\Users\User\OneDrive\KAMPUS\KELAS\1E\SEM 2\Praktikum-ASD>
```

PERTANYAAN

1. Jelaskan mengenai perbedaan 2 method yang dibuat yaitu pangkatBF() dan pangkatDC()!
 - 1) Pada method pangkatBF() operasi mencari hitung hasil pangkat dilakukan dengan cara buteforce yang dilakukan dengan iteratife/prulangan/looping dan lagoritma buteforcenya adalah mendeklarasikan dahulu hasil = 1 lalu melakukan perulanga dengan batas n(pangkatnya) dan dalam perulangan tersebut dilakukan looping dari hasil tadi dikali dengan a(bilangan yang akan dipangkatkan) dan perulangan akan terus berlanjut hingga < n sehingga a akan menghasilkan nilai hasil dari pemangkatnya

- 2) Pada method pangkatDC() operasi mencari hitung hasil pangkat dilakukan dengan cara divide conquer yang dilakukan dengan rekursif dan algoritma divide conquer yang dilakukan terbagi dalam 3 tahap yaitu : divide, conquer, combine.
2. Apakah tahap combine sudah termasuk dalam kode tersebut? Tunjukkan!
Combine terdapat pada method PangkatDC()

```
if (n%2==1) {  
    return (pangkatDC(a, n/2) * pangkatDC(a, n/2) * a);  
}else{  
    return (pangkatDC(a, n/2) * pangkatDC(a, n/2));  
}
```

3. Pada method pangkatBF() terdapat parameter untuk melewati 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?
- Relevan jika ingin fleksibel, agar method bisa digunakan dengan nilai lain.
 - Tidak wajib menggunakan parameter karena sudah ada atribut nilai dan pangkat di dalam class.
 - Bisa juga dibuat versi tanpa parameter

```
int pangkatBF() {  
    int hasil = 1;  
    for(int i = 0; i < pangkat; i++){  
        hasil *= hasil ;  
    }  
    return hasil;  
}
```

4. Tarik tentang cara kerja method pangkatBF() dan pangkatDC()!
- o pangkatBF();
 1. hasil awal = 1
 2. ulangi sebanyak pangkat kali
 3. disetiap iterasi, hasil dikali dengan nilai
 4. setelah perulangan selesai, hasil berisi hasil akhir dari perpangkatan

- pangkatDC();
 1. jika pangkat = 0 → hasil = 1 → hasil = nilai
 2. untuk pangkat > 1
 - a. jika genap

```

        if (n%2==1) {
            return (pangkatDC(a, n/2) * pangkatDC(a,
n/2) *a);
        }

```

- b. jika ganjil

```

        else{
            return (pangkatDC(a, n/2) * pangkatDC(a,
n/2));
        }

```

PERCOBAAN 3 Menghitung Sum Array dengan Algoritma Brute dan Divide and Conquer

Langkah-langkah

1. Pada paket minggu5. Buat class baru yaitu class Sum. Tambahkan pula konstruktor pada class Sum.

```

double keuntungan[];

Sum02 (int e1) {
    keuntungan = new double[e1];
}

```

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

```

double totalBF() {
    double total = 0;
    for (int i = 0; i < keuntungan.length; i++) {
        total = total + keuntungan[i];
    }
    return total;
}

```


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

```
double totalDC(double[] arr, int l, int r) {
    if (l == r) {
        return arr[l];
    }

    int mid = (l + r) / 2;
    double lsum = totalDC(arr, l, mid);
    double rsum = totalDC(arr, mid + 1, r);
    return lsum + rsum;
}
```

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

```
Scanner input = new Scanner(System.in);
System.out.print("Masukkan jumlah elemen: ");
int elemen = input.nextInt();
```

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.

```
Sum02 sm = new Sum02 (elemen);
for (int i = 0; i < elemen; i++) {
    System.out.print("Masukkan keuntungan ke-" + (i
+ 1) + ": ");
    sm.keuntungan[i] = input.nextDouble();
}
```

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

```
System.out.println("Total keuntungan menggunakan
Brute force: " + sm.totalBF());
System.out.println("Total keuntungan menggunakan
Divide and Conquer: " + sm.totalDC(sm.keuntungan, 0, elemen -
1));
```

Outputnya

```
\jdt_ws\Praktikum-ASD_6ab0b5a7\bin' 'jobsheet5.MainSum02'
Masukkan jumlah elemen: 5
Masukkan keuntungan ke-1: 10
Masukkan keuntungan ke-2: 20
Masukkan keuntungan ke-3: 30
Masukkan keuntungan ke-4: 40
Masukkan keuntungan ke-5: 50
Total keuntungan menggunakan Bruteforce: 150.0
Total keuntungan menggunakan Divide and Conquer: 150.0
PS C:\Users\User\OneDrive\KAMPUS\KELAS\1E\SEM 2\Praktikum-ASD>
```

PERTANYAAN

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

Jawab:

Karena mid digunakan untuk membagi array menjadi dua bagian, yaitu bagian kiri (l sampai mid) dan bagian kanan (mid+1 sampai r)

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

```
double lsum = totalDC(arr, l, mid);
double rsum = totalDC(arr, mid+1, r);
```

Jawab:

Kedua statement tersebut digunakan untuk menghitung total elemen bagian kiri (lsum) dan kanan (rsum) array secara rekursif.

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

```
return lsum+rsum;
```

Jawab:

Karena total keseluruhan adalah penjumlahan dari bagian kiri dan kanan array. menggabungkan solusi dari bagian-bagian kecil

4. Apakah base case dari totalDC()?

Jawab:

jika **hanya satu elemen yang tersisa**, maka langsung dikembalikan nilainya tanpa dibagi lagi

```
if (l == r) {
    return arr[l];
}
```

5. Tarik Kesimpulan tentang cara kerja totalDC()

Jawab:

membagi array menjadi dua bagian (Divide), menghitung total masing-masing bagian secara rekursif (Conquer), lalu menjumlahkan hasilnya (Combine).