LAPORAN HASIL PRAKTIKUM

ALGORITMA DAN STRUKTUR DASAR

JOBSHEET 5

NAMA : **DANDIKA MARTHA C.**

NIM : **244107020092**

KELAS : **1E**

Program Studi Teknik Informatika

Jurusan Teknologi Informasi

Praktikum

2025

Praktikum 1

1.1 Code

- Faktorial07.java

```
package jobsheet_5;

public class Faktorial07 {
    int faktorialBF(int n) {
        int fakto = 1;
        for (int i=1; i<=n; i++) {
            fakto = fakto * i;
        }
        return fakto;
    }
    int faktorialDC(int n) {
        if(n==1) {
            return 1;
        } else {
            int fakto = n *
faktorialDC(n-1);
            return fakto;
        }
    }
}
```

- MainFaktorial07.java

```
package jobsheet_5;
import java.util.Scanner;
public class MainFaktorial07 {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Masukkan nilai: ");
        int nilai = input.nextInt();

        Faktorial07 fk = new Faktorial07();
        System.out.println("Nilai faktorial " +nilai+" menggunakan BF: "
+fk.faktorialBF(nilai));
        System.out.println("Nilai faktorial " +nilai+" menggunakan DC: "
+fk.faktorialDC(nilai));
    }
}
```

1.2 Hasil

```
Masukkan nilai: 5
Nilai faktorial 5 menggunakan BF: 120
Nilai faktorial 5 menggunakan DC: 120
PS G:\DATA DIKO\POLINEMA TI\SEMESTER 2\ASD\praktikumASD>
```

Pertanyaan

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

Jawab : Pada penggunaan if merupakan base case di mana rekursif akan berhenti sesuai dengan keinginan contohnya **if(n==1),** maka ketika **n** bernilai 1 rekursif akan berhenti dan mereturn nilai = 1, sedangkan else adalah fungsi rekursif yang di panggil untuk menghitung nilai **n** dikalikan dengan fungsi **faktorialDC(n-1)** dimana akan terus berulang sampai base case dalam code **(n==1).**

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

Jawab : Memungkinkan, kita bisa menggunakan perulangan while, maupun do-while

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

Untuk hasilnya tetap sama seperti menggunakan **for diatas**

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

Jawab : fakto *= i adalah operator assignment dimana artinya **fakto = fakto * i**, contohnya fakto = 1, i = 1 dan n = 5, jika menggunakan perulangan maka fakto = 1 * 1; dan fakto = 1 * 2 dan seterusnya. Sedangkan untuk **fakto = n * faktorialDC(n-1)** menggunakan rekursif untuk mengembalikan nilai **n dikalikan nilai n-1** contohnya n = 5, maka fakto = 5 * faktorialDC(4) dan seterusnya samapi base case. Dan dihitung terbalik dimulai dari base case.

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

Jawab : faktorialBF() menggunakan perulangan iteratif untuk melakukan perhitungan dan penggunaan memori yang lebih kecil. Sedangkan untuk **faktorialDC()** menangani perhitungan dengan terus mengecilkan nilai **n** menjadi bagian lebih kecil hingga menyentuh base case dan menggunakan memori yang lebih besar.

Praktikum 2

2.1 Code

Pangkat07.java

```
package jobsheet_5;

public class Pangkat07 {
    public Pangkat07() {
    }
    int nilai, pangkat;
    public Pangkat07(int n, int p) {
        nilai = n;
        pangkat = p;
    }
    int pangkatBF(int a, int n) {
        int hasil = 1;
        for (int i=0; i<n; i++) {
            hasil = hasil * a;
        }
        return hasil;
    }
    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));
            }
        }
    }
}
```

MainPangkat07.java

```
package jobsheet_5;
import java.util.Scanner;
public class MainPangkat07 {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Masukkan jumlah elemen: ");
        int elemen = input.nextInt();

        Pangkat07[] png = new Pangkat07[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 Pangkat07(basis, pangkat);
        }
        System.out.println("HASIL PANGKAT BRUTEFORCE:");
        for (Pangkat07 p : png) {
            System.out.println(p.nilai+"^"+p.pangkat+": "+p.pangkatBF(p.nilai,
p.pangkat));
        }
        System.out.println("HASIL PANGKAT DIVIDE AND CONQUER:");
        for (Pangkat07 p : png) {
            System.out.println(p.nilai+"^"+p.pangkat+": "+p.pangkatDC(p.nilai,
p.pangkat));
        }
    }
}
```

2.2 Hasil

```
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 G:\DATA DIKO\POLINEMA TI\SEMESTER 2\ASD\praktikumASD>
```

Pertanyaan

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

Jawab : pangkatBF() menggalikan bilangan basis dengan dirinya sendiri sebanyak pangkat yang di inputkan menggunakan perulangan. Sedangkan untuk **pangkatDC()** membagi perhitungan perkalian ke dalam bentuk yang lebih sederhana, contoh a(basis) = 3 dan n(pangkat) = 2 (a , n), makan perhitungan sama dengan (3 , 1) * (3 ,1) = 3 * 3 = 9.

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

Jawab : Sudah,

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

Pada kode tersebut kita memecah perhitungan ke bentuk yang lebih kecil dan menggabungkan 2 fungsi tersebut dengan perkalian.

3. 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?

Jawab : Masih relevan, method pangkatBF() bisa dibuat tanpa parameter.

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

Dan untuk di class MainFaktorial07 kita tidak perlu memamggil method pangkatBF() tanpa parameter

```
System.out.println(p.nilai+"^"+p.pangkat+": "+p.pangkatBF());
```

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

Jawab :

- cara kerja pangkatBF() , contoh nilai = 2 dan pangkat = 3.

Iterasi 1 : Hasil = 1 * 2 = 2

Iterasi 2 : Hasil = 2 * 2 = 4

Iterasi 3 : Hasil = 4 * 2 = 8

Iterasi diulang sebanyak nilai pangkat.

- cara kerja pangkatDC(), contoh nilai = 2 dan pangkat = 3.

Jika n/pangkat > 1 maka akan ke pemilihan apakah n%2==1, karena kita memakai n = 3, maka menggunakan **return (pangkatDC(a, n/2)*pangkatDC(a, n/2)*a**);. Maka nilai menjadi **(2 , 1) * (2 * 1) * 2. Ketika n bernilai 1 maka fungsi mereturn nilai = a.**

Maka perhitungannya **2 * 2 * 2**

Percobaan 3

3.1 Code

Sum07.java

```
package jobsheet_5;

public class Sum07 {
    double keuntungan[];

    Sum07(int el) {
        keuntungan = new double[el];
    }
    double totalBF(){
        double total = 0;
        for (int i = 0; i < keuntungan.length; i++) {
            total = total + keuntungan[i];
        }
        return total;
    }
    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;
    }
}
```

MainSum07.java

```
package jobsheet_5;
import java.util.Scanner;
public class MainSum07 {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Masukkan jumlah elemen: ");
        int elemen = input.nextInt();

        Sum07 sm = new Sum07(elemen);
        for(int i=0;i<elemen;i++) {
            System.out.print("Masukkan keuntungan ke-" +(i+1)+": ");
            sm.keuntungan[i] = input.nextDouble();
        }
        System.out.println("Total keuntungan menggunakan Bruteforce:
"+sm.totalBF());
        System.out.println("Total keuntungan menggunakan Divide and
Conquer: "+sm.totalDC(sm.keuntungan, 0, elemen-1));
    }
}
```

3.2 Hasil

```
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 G:\DATA DIKO\POLINEMA TI\SEMESTER 2\ASD\praktikumASD>
```

Pertanyaan

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

Jawab : Variabel mid = (l+r) / 2, digunakan untuk membagi array menjadi **2 bagian** hingga nilai l==r, kemudian di gabung untuk melakukan perhitungan. Hal ini untuk mempermudah penggambaran perhitungan.

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

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

Jawab : Untuk membagi array menjadi bentuk yang lebih kecil. Contoh variable **lsum = totalDC(arr, l, mid) arr** = array yang di inginkan, **l** = indeks awal dari array, dan untuk **mid** = adalah indeks yang di dapat setengah dari total indeks array, **mid digunakan sebagai batas array sebelah kiri**.Untuk **rsum, arr** = array yang diinginkan, **mid+1** = adalah awalan indeks dari array sebelah kanan, dan **r** adalah batas dari array sebelah kanan.

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

```
return lsum+rsum;
```

Jawab : Karena kita menggunakan metode DC hal itu di gunakan untuk menghitung nilai dari setiap array yang sudah di bagi dalam array sebelah kiri dan sebelah kanan.

4. Apakah base case dari totalDC()?

Jawab :

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

5. Tarik Kesimpulan tentang cara kerja totalDC()

Jawab : totalDC() menggunakan penyelesaian dengan divide and conqueror, dimana perhitungan total array dengan membagi array ke dalam 2 bagian (array kiri dan array kanan) sampai ke bentuk paling sederhana yakni menjadi penambahan 1 elemen array ditambah 1 elemen array lainnya, kemudian di gabung kembali menjadi penjumlahan seluruh nilai array.