Praktikum 5

Bisma Adhiaksa

TI-1B

244107020216

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

1. Buat folder Jobsheet5 pada folder Praktikum ASD
2. Lalu buat class Faktorial dan isikan method yang diperlukan

```
class Faktorial {

    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;
        }
    }
}
```

3. Lalu buat file MainFaktorial

```java
import java.util.Scanner;

public class MainFaktorial {

    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print(s:"Masukkan nilai: ");
        int nilai = input.nextInt();
        Faktorial fk = new Faktorial();

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

4. Run program

```
Masukkan nilai: 5
Nilai Faktorial 5 menggunakan BF: 120
Nilai Faktorial 5 menggunakan DC: 120
PS D:\Kuliah\Semester2\PRAKTIKUM-ASD\jobsheet5>
```

5.2.3. Pertanyaan

1. Pada base line Algoritma Divide Conquer untuk melakukan pencarian nilai faktorial, jelaskan perbedaan bagian kode pada penggunaan if dan else!
2. Apakah memungkinkan perulangan pada method `faktorialBF()` diubah selain menggunakan for? Buktikan!
3. Jelaskan perbedaan antara `fakto *= i;` dan int `fakto = n * faktorialDC(n-1);` !
4. Buat Kesimpulan tentang perbedaan cara kerja method `faktorialBF()` dan `faktorialDC()`!

Jawaban

1. Dalam base line Algorima Divide Conquer pada class faktorial menggunakan pemilihan if(n==1) return 1. Artinya jika nilai yang dicek nantinya adalah 1 maka hasil yang akan ditampilkan oleh program main nantinya adalah 1. Base line ini berfungsi sebagai batas dari devide conquer dimana perulangan akan berakhir saat n sudah sama dengan 1.

2. Bisa menggunakan while, berikut saya beri perubahannya

```java
class Faktorial {

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

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

```
OBLEMS 1   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   ...

S D:\Kuliah\Semester2\PRAKTIKUM-ASD\jobsheet5> cd "d:\Kuli
bsheet5\" ; if ($?) { javac MainFaktorial.java } ; if ($?)
asukkan nilai: 5
ilai Faktorial 5 menggunakan BF: 120
ilai Faktorial 5 menggunakan DC: 120
S D:\Kuliah\Semester2\PRAKTIKUM-ASD\jobsheet5> cd "d:\Kuli
bsheet5\" ; if ($?) { javac MainFaktorial.java } ; if ($?)
asukkan nilai: 5
ilai Faktorial 5 menggunakan BF: 120
ilai Faktorial 5 menggunakan DC: 120
S D:\Kuliah\Semester2\PRAKTIKUM-ASD\jobsheet5>
```

3. Fakto *= i akan menghitung mulai dari angka kecil ke angka n, sedangkan fakto = n * faktorialDC(n-1) akan menghitung mulai dari angka ke n ke angka kecil
4. FaktorialBF menggunakan perulangan yang perhitungannya dimulai dari angka kecil, sedangkan FaktorialDC menggunakan rekursif yang perhitungannya dimulai dari angka ke n ke kecil, sebenarnya tergantung kodenya juga

Percobaan 2: Menghitung Hasil Pangkat dengan Algoritma Brute Force dan Divide and Conquer

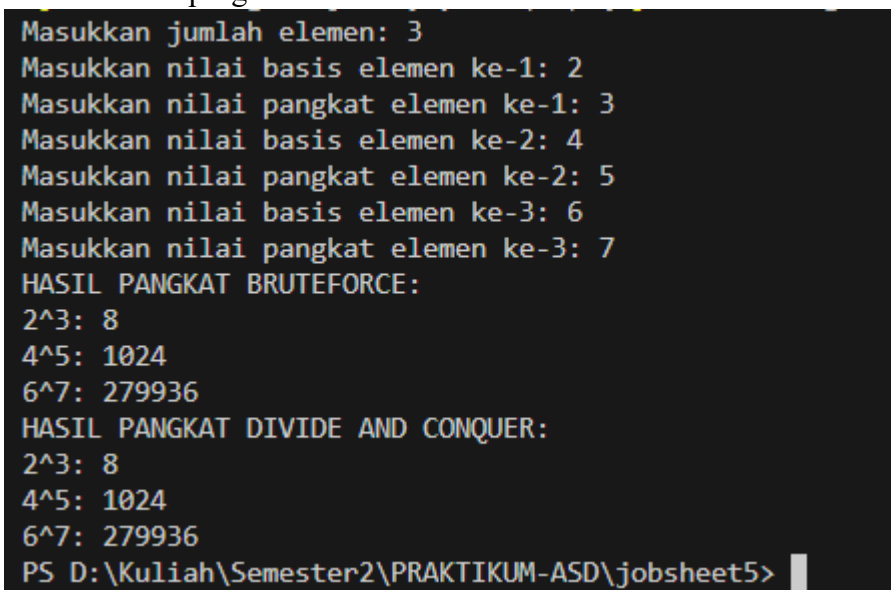
1. Buat class baru dengan nama Pangkat dan beri atribut dan buat konstruktornya

```java
class Pangkat {
    int nilai, pangkat;

    public Pangkat(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));
            }
        }
    }
```

2. Buat class baru dengan nama MainPangkat dan beri input dan instansiasi objek

```java
import java.util.Scanner;

public class MainPangkat {

    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print(s:"Masukkan jumlah elemen: ");
        int elemen = input.nextInt();

        Pangkat[] png = new Pangkat[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 Pangkat(basis, pangkat);
        }

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

3. Hasil dari run program



5.3.3. Pertanyaan

1. Jelaskan mengenai perbedaan 2 method yang dibuat yaitu `pangkatBF()` dan `pangkatDC()`!
2. Apakah tahap *combine* sudah termasuk dalam kode tersebut? Tunjukkan!
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?
4. Tarik tentang cara kerja method `pangkatBF()` dan `pangkatDC()`!

Jawaban

1. pangkatBF menggunakan perulangan dalam perhitungannya, sedangkan pangkatDC menggunakan rekursif dalam perhitungannya
2. Ya, sudah termasuk. Dalam program terdapat kode untuk return yang mana adalah proses menggabungkan pecahan dari masalah.

3. bisa

```java
class Pangkat {

    int nilai, pangkat;

    public Pangkat(int n, int p) {
        nilai = n;
        pangkat = p;
    }

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

```

```
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 D:\Kuliah\Semester2\PRAKTIKUM-ASD\jobsheet5>
```

4. pangkatBF bekerja dengan menghitung dengan hasil kali bilangan itu sendiri, lalu akan di return setelah pengurangan sejumlah nilai dari pangkatnya. Sedangkan pangkatDC bekerja dengan rekursif

Percobaan 3: Menghitung Sum Array dengan Algoritma Brute Force dan Divide and Conquer

1. Buat class baru dengan nama Sum, lalu buat atribut

```
class Sum {

    double keuntungan[];
```

2. Lalu buat konstruktor dan method yang diperlukan

```
Sum(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;
}
```

3. Lalu buat class baru bernama MainSum untuk menjalankan program utama yang berisi program untuk menginput, instansiasi dan inisialisasi

```
import java.util.Scanner;

public class MainSum {
    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print(s:"Masukkan jumlah elemen: ");
        int elemen = input.nextInt();

        Sum sm = new Sum (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, l:0, elemen-1));
    }
}
```

5.4.3. Pertanyaan

1. Kenapa dibutuhkan variable `mid` pada method `TotalDC()`?
2. Untuk apakah statement di bawah ini dilakukan dalam `TotalDC()`?

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

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

```
return lsum+rsum;
```

4. Apakah base case dari `totalDC()`?
5. Tarik Kesimpulan tentang cara kerja `totalDC()`

Jawaban

1. Dibutuhkan variabel mid untuk titik awal dari penghitungan sebelah kanan, karena pada method ini perhitungan dibagi menjadi 2
2. lsum untuk menghitung jumlah nilai mulai dari indeks paling awal ke mid, lalu untuk rsum untuk menghitung jumlah nilai dari indeks Tengah + 1 ke indeks ukuran array-1
3. karena penilaiannya dibagi menjadi kenan dan kiri
4. jika l == r
5. perhitungan keuntungan menggunakan fungsi rekursif untuk melakukan proses divide yang diimplementasikan dengan pemiihan (if-else, if-else). lalu melakukan tahapan conquer untuk menyelesaikan setiap masalah tersebut yang dibagi menjadi 2 bagian yaitu kanan dan kiri, lalu pada tahap akhir atau combine maka semua hasil penyelesaian tadi dijadikan satu menjadi solusi