


Nama : Rachmad Aprisandhy
Kelas : TI-1B
no Abs: 18



JOBSHEET 5

A. Percobaan 1

1. Buat Folder Jobsheet 5

 JOBSHEET 5

2. Buat class baru bernama Faktorial

 Faktorial18.java 

3. Tambahkan Method pada 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;  
    }  
}
```

4. Membuat Class FaktorialMain untuk menjalankan program

 Faktorial18main.java 1 

5. Buat Scanner untuk menginputkan nilai yang akan dicari Faktorialnya

```
public static void main(String[] args) {  
    Scanner input = new Scanner(System.in);  
    System.out.println(x:"Masukkan nilai ");  
    int nilai = input.nextInt();  
}
```

6. Buat Objek dari class Faktorial

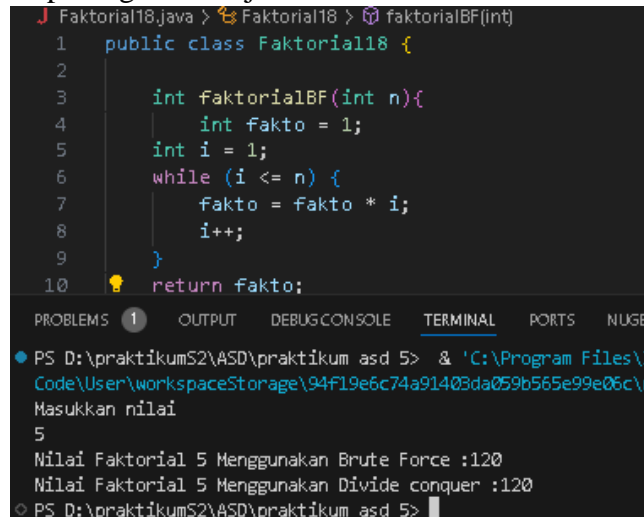
```
Faktorial18 fk = new Faktorial18();  
System.out.println("Nilai Faktorial " + nilai+ " Menggunakan Brute Force : " + fk.faktorialBF(nilai));  
System.out.println("Nilai Faktorial " + nilai+ " Menggunakan Divide conquer : " + fk.faktorialDC(nilai));
```

7. Tampilkan Hasilnya

```
Masukkan nilai  
5  
Nilai Faktorial 5 Menggunakan Brute Force :120  
Nilai Faktorial 5 Menggunakan Divide conquer :120
```

B. Pertanyaan

1. Perbedaan antara if dan else adalah, Jika nilai $n = 1$ yang merupakan base case maka program akan berhenti sementara jika nilai n bukan 1 maka nilai n akan dikurangi 1 sehingga memecah masalah menjadi sub masalah
2. Dapat diganti menjadi while



```
J Faktorial18.java > Faktorial18 > faktorialBF(int)
1 public class Faktorial18 {
2
3     int faktorialBF(int n){
4         int fakto = 1;
5         int i = 1;
6         while (i <= n) {
7             fakto = fakto * i;
8             i++;
9         }
10        return fakto;
11    }
12}

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS NUGET
PS D:\praktikumS2\ASD\praktikum asd 5> & 'C:\Program Files\Code\User\workspaceStorage\94f19e6c74a91403da059b565e99e06c\
Masukkan nilai
5
Nilai Faktorial 5 Menggunakan Brute Force :120
Nilai Faktorial 5 Menggunakan Divide conquer :120
PS D:\praktikumS2\ASD\praktikum asd 5>
```

3. Faktorial menggunakan metode iteratif, sementara faktorialDC(n - 1); menggunakan metode rekursif
4. Brute force Menggunakan perulangan yaitu memulai menghitung dari 1 hingga nilai yang dicari, sementara divide conquer menggunakan fungsi rekursif dimana memanggil dirinya sendiri hingga mencapai base case

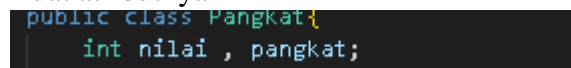
C. Percobaan 2

1. Buat class baru bernama pangkat



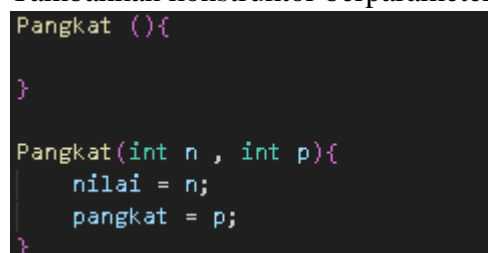
```
Pangkat.java X
```

2. Buat atributnya



```
public class Pangkat{
    int nilai , pangkat;
}
```

3. Tambahkan konstruktor berparameter



```
Pangkat (){
}

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

4. Buatlah method PangkatBF dan PangkatDC

```
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));
        }
    }
}
```

5. Buat class mainPangkat

J MainPangkat.java 1 X

6. Buat kode untuk menginputkan elemen

```
public static void main (String [] args){
    Scanner input = new Scanner(System.in);
    System.out.println(x:"Masukka jumlah elemen :");
    int elemen = input.nextInt();
}
```

7. Inisialisasi array of object

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

8. Panggil menggunakan return value

```
System.out.println(x:" Hasil pangkat Brute Force");
for (Pangkat p : png){
    System.out.println(p.nilai + "^" + p.pangkat + " : " + p.PangkatBF(p.nilai,p.pangkat));
}
System.out.println(x:" Hasil pangkat Divide Conquer");
for (Pangkat p : png){
    System.out.println(p.nilai + "^" + p.pangkat + " : " + p.PangkatDC(p.nilai,p.pangkat));
}
```

9. Lalu perhatikan hasilnya

```
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 Brute Force
2^3 : 8
4^5 : 1024
6^7 : 279936
Hasil pangkat Divide Conquer
2^3 : 8
4^5 : 1024
6^7 : 279936
PS D:\praktikumS2\ASD\praktikum asd 5> d:; cd 'd:
```

D. Pertanyaan

1. PangkatBF() Menggunakan perulangan (for) untuk mengalikan angka sebanyak n kali. Rumusnya $O(n)$ sementara pangkatDC() Menggunakan perulangan (for) untuk mengalikan angka sebanyak n kali. Rumusnya $O(\log n)$
2. Sudah yaitu ketika dua hasil recursive dikalikan kembali

```
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. Bisa ,hasilnya akan seperti ini
pangkat

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

pangkat main

```
System.out.println(x:" Hasil pangkat Brute Force");
for (Pangkat p : png){
    System.out.println(p.nilai + "^" + p.pangkat + " : " + p.PangkatBF());
}
```

4. Mengalikan hasil dengan nilai sebanyak pangkat kali menggunakan perulangan dan mengembalikan hasil akhir

Jika n lebih besar, pecah masalah menjadi dua bagian dengan $n / 2$., Lakukan rekursi untuk PangkatDC($a, n/2$)., Jika n ganjil, hasil akhirnya dikalikan a ., Mengembalikan hasil dari kombinasi rekursi.

E. Percobaan 3

1. Buat class baru bernama sum dan tambahkan konstruktor

```
J sum18.java X
double Keuntungan[];

sum18 (int el){
    Keuntungan = new double[el];
}
```

2. Tambahkan method totalBF() dan totalDC()

```
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. Buat class baru bernama mainsum

```
J mainSum18.java 1 X
```

4. Buat method main sekaligus buat instansiasi objek untuk memanggil atribut ataupun fungsi pada class Sum

```
public class mainSum18{
    Run | Debug
    public static void main (String [] args){
        Scanner input = new Scanner (System.in);
        System.out.print(s:"Masukkan jumlah Elemen : ");
        int elemen = input.nextInt();
    }
}
```

5. Buat objek class sum dan berikan atribut sm

```
sum18 sm = new sum18(elemen);
for(int i=0;i<elemen;i++){
    System.out.println(x:"Masukkan keuntungan ke =");
    sm.Keuntungan[i] = input.nextDouble();
}
```

6. Print hasilnya

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

7. Hasilnya akan jadi seperti ini

```
Masukkan jumlah Elemen : 5  
Masukkan keuntungan ke =  
10  
Masukkan keuntungan ke =  
20  
Masukkan keuntungan ke =  
30  
Masukkan keuntungan ke =  
40  
Masukkan keuntungan ke =  
50  
Total Keuntungan menggunakan Brute Force :150.0  
Total Keuntungan menggunakan Divide Conquer :150.0
```

F. Pertanyaan

1. Fungsi mid sangat penting dalam metode Divide Conquer karena menentukan titik pembagian array, dan membuat proses rekursi berjalan dengan benar, dan membantu dalam penggabungan hasil perhitungan dari dua bagian array.
2. Membagi array menjadi 2 bagian dan menghitung masing masing bagian tersebut dengan cara rekursif
3. Apabila nanti kedua bagian itu sudah mencapai base case maka nilainya akan ditambahkan melalui syntax ini
4. Base casenya adalah (l == r)
5. menggunakan pendekatan Divide and Conquer untuk menghitung jumlah elemen dalam array.