LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

**FILE NAME =ABSEN_NAME _CLASS_P1**

** Pertemuan mengikuti pertemuan ke berapa*

### *4.2* Menghitung Nilai Faktorial dengan Algoritma Brute Force dan Divide and Conquer *

*screenshoot code program hasil percobaan**

```
public class Faktorial04 {

public int nilai;


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

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

```
package P5;

import java.util.Scanner;

public class MainFaktorial04 {
  Run | Debug
  public static void main(String[] args) {

Scanner sc04 = new Scanner(System.in);

System.out.println(x:"----------------------");
System.out.print(s:"Masukan jumlah elemen: ");
int iJml = sc04.nextInt();
```

```
Faktorial04[] fk = new Faktorial04[iJml];
  for (int i = 0; i < iJml; i++) {
    fk[i] = new Faktorial04();
    System.out.println("Masukan nilai data ke-" +(i+1) + ":");
    int iNilai = sc04.nextInt();
    fk[i].nilai = iNilai;
  }

System.out.println(x:"HASIL - BRUTE FORCE");
for (int i = 0; i < iJml; i++) {
  System.out.println("Hasil perhitungan faktorial menggunakan Bru
}
System.out.println(x:"HASIL - DIVIDE AND CONQUER");
for (int i = 0; i < iJml; i++) {
  System.out.println("Hasil perhitungan faktorial menggunakan Div
}
```

⊗ 0 ⚠ 13  0  Java: Ready

```
Masukan jumlah elemen: 3
Masukan nilai data ke-1:
5
Masukan nilai data ke-2:
8
Masukan nilai data ke-3:
3
HASIL - BRUTE FORCE
Hasil perhitungan faktorial menggunakan Brute Force adalah 120
Hasil perhitungan faktorial menggunakan Brute Force adalah 40320
Hasil perhitungan faktorial menggunakan Brute Force adalah 6
HASIL - DIVIDE AND CONQUER
Hasil perhitungan faktorial menggunakan Divide and Conquer adalah 120
Hasil perhitungan faktorial menggunakan Divide and Conquer adalah 4032
Hasil perhitungan faktorial menggunakan Divide and Conquer adalah 6
```

## Question :

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

if digunakan untuk mengecek kasus dasar, sedangkan else dapat digunakan untuk menangani kasus rekursif atau operasi lainnya.

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

```
int faktorialBF(int n) {
  if (n == 0 || n == 1) {
      return 1;
  } else {
      return n * faktorialBF(n - 1);
  }
}

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

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

adalah cara untuk mengalikan fakto dengan i secara berulang untuk mendapatkan faktorial dari suatu bilangan

n * faktorialDC(n-1);

Metode faktorialDC() dipanggil secara rekursif dengan parameter n-1 hingga mencapai kasus dasar, di mana faktorial dari 1 adalah 1. Kemudian, hasil dari panggilan rekursif dikalikan dengan n. Ini merupakan pendekatan rekursif untuk menghitung faktorial.

### *4.3* Menghitung Hasil Pangkat dengan Algoritma Brute Force dan Divide and Conque *

*screenshoot code program hasil percobaan**

```java
public class Pangkat04 {
  public int nilai, pangkat;


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

int pangkatDC (int a, int n){
  if (n==0) {
    return 1;
  }else{
      int nilai = pangkatDC(a, n/2);
      if (n%2==1) {
        return nilai * nilai * a;
      } else {
        return nilai * nilai;
      }



  }
}
}
```

```java
public class MainPangkat04 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc04 = new Scanner(System.in);

        System.out.println(x:"----------------------");
        System.out.print(s:"Masukan jumlah elemen yang dihitung: ");
        int iLmn = sc04.nextInt();

        Pangkat04[] png = new Pangkat04[iLmn];
        for (int i = 0; i < png.length; i++) {
            png[i] = new Pangkat04();
            System.out.println(x:"Masukan nilai yang hendak dipangkatkan: ");
            png[i].nilai = sc04.nextInt(); // Simpan nilai yang dibaca ke dalam objek
            System.out.println(x:"Masukan nilai pemangkat: ");
            png[i].pangkat = sc04.nextInt(); // Simpan pangkat yang dibaca ke dalam objek
        }

        System.out.println(x:"HASIL PANGKAT - BRUTE FORCE");
        for (int i = 0; i < iLmn; i++) {
            System.out.println(
                    "Hasil dari " + png[i].nilai + " pangkat " + png[i].pangkat + " adalah " +
                            png[i].pangkatBF(png[i].nilai, png[i].pangkat)
            );
        }

        System.out.println(x:"HASIL PANGKAT - DIVIDE CONQUER");
        for (int i = 0; i < iLmn; i++) {
            System.out.println(
                    "Hasil dari " + png[i].nilai + " pangkat " + png[i].pangkat + " adalah " +
                            png[i].pangkatDC(png[i].nilai, png[i].pangkat)
```

```
Masukan jumlah elemen yang dihitung: 2
Masukan nilai yang hendak dipangkatkan
6
Masukan nilai pemangkat:
2
Masukan nilai yang hendak dipangkatkan
4
Masukan nilai pemangkat:
3
HASIL PANGKAT - BRUTE FORCE
Hasil dari 6 pangkat 2 adalah 36
Hasil dari 4 pangkat 3 adalah 64
HASIL PANGKAT - DIVIDE CONQUER
Hasil dari 6 pangkat 2 adalah 36
Hasil dari 4 pangkat 3 adalah 64
```

## Question :

1. Jelaskan mengenai perbedaan 2 method yang dibuat yaitu `PangkatBF()` dan `PangkatDC()`!

   Answer: perbedaan utama antara kedua metode ini terletak pada pendekatan yang digunakan untuk menghitung pangkat suatu bilangan. Metode Brute Force menggunakan pendekatan iteratif sederhana, sedangkan metode Divide and Conquer menggunakan pendekatan rekursif yang membagi permasalahan menjadi submasalah yang lebih kecil

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

   Answer : Sudah

```
int pangkatDC(int a, int n){
    if(n==0){
        return 1;
    }else{
        int hasil = pangkatDC(a, n / 2);
        if(n%2== 0) {
            return (hasil * hasil);

        }else{
            return (hasil * hasil * a);
        }
    }
}
```

3. Modifikasi kode program tersebut, anggap proses pengisian atribut dilakukan dengan konstruktor.

```
public Pangkat04(int nilai, int pangkat) {
    this.nilai = nilai;
    this.pangkat = pangkat;
}
```

4. Tambahkan menu agar salah satu method yang terpilih saja yang akan dijalankan menggunakan switch-case!

```
switch (Pilih) {
 case 1:
 System.out.println(x:"HASIL PANGKAT - BRUTE FORCE");
 for (int i = 0; i < iLmn; i++) {
     System.out.println(
             "Hasil dari " + png[i].nilai + " pangkat " + png[i].pangkat + " adalah " +
                     png[i].pangkatBF(png[i].nilai, png[i].pangkat)
     );
 }
   break;
 case 2:
 System.out.println(x:"HASIL PANGKAT - DIVIDE CONQUER");
 for (int i = 0; i < iLmn; i++) {
     System.out.println(
             "Hasil dari " + png[i].nilai + " pangkat " + png[i].pangkat + " adalah " +
                     png[i].pangkatDC(png[i].nilai, png[i].pangkat)
```

## 4.4 Menghitung Sum Array dengan Algoritma Brute Force dan Divide and Conquer

```
public class Sum04 {
  int elemen;
  double keuntungan[], total;


  Sum04(int elemen){
    this.elemen = elemen;
    this.keuntungan = new double[elemen];
    this.total = 0;
  }

double totalBF (double arr[]){
   for (int i = 0; i < elemen; i++) {
    total = total + arr[i];
   }
   return total;
}

double totalDC (double arr[], int l, int r) {
  if (l==r) {
    return arr[l];
  } else if(l < r){
    int mid = (l+r) /2;
    double lsum = totalDC(arr, l, mid);
    double rsum = totalDC(arr, mid+1, r);
    return lsum + rsum;
```

```
System.out.println(x:"------------------------");
System.out.println(x:"Program menghitung keuntungan total (satuan juta. misal 5.9 )");
System.out.print(s:"Masukan jumlah bulan: ");
int elm = sc04.nextInt();

Sum04 sm = new Sum04(elm);
System.out.println(x:"============================");
for (int i = 0; i < sm.elemen; i++) {
  System.out.println("Masukan untung bulan ke - "+(i+1)+" = ");
  sm.keuntungan[i] = sc04.nextDouble();
}

System.out.println(x:"============================");
System.out.println(x:"Algoritma Brute Force");
System.out.println("Total keuntungan perusahaan selama " + sm.elemen + " bulan adalah = " + sm.totalBF(sm.ke
System.out.println(x:"============================");
System.out.println(x:"Algoritma Divide Conquer");
System.out.println("Total keuntungan perusahaan selama " + sm.elemen + " bulan adalah = " + sm.totalDC(sm.ke
```

```
Program menghitung keuntungan total (satuan juta. misal 5.9 )
Masukan jumlah bulan: 5
============================
Masukan untung bulan ke - 1 =
8.5
Masukan untung bulan ke - 2 =
9.54
Masukan untung bulan ke - 3 =
7.2
Masukan untung bulan ke - 4 =
9.1
Masukan untung bulan ke - 5 =
6
============================
Algoritma Brute Force
Total keuntungan perusahaan selama 5 bulan adalah = 40.339999999999996
============================
Algoritma Divide Conquer
Total keuntungan perusahaan selama 5 bulan adalah = 40.339999999999996
Java: Ready
```

## Question :

1. Mengapa terdapat formulasi *return value* berikut?Jelaskan!
formulasi return lsum + rsum + arr[mid]; digunakan untuk menghitung total keseluruhan array menggunakan pendekatan Divide and Conquer.
2. Kenapa dibutuhkan variable `mid` pada method `TotalDC()`?
variabel mid digunakan dalam metode totalDC() agar kita dapat membagi array ke dalam submasalah yang lebih kecil dengan cara yang seimbang dalam pendekatan Divide and Conquer. Hal ini membantu meningkatkan efisiensi algoritma dan memastikan bahwa submasalah diproses dengan benar pada setiap langkah rekursif.
3. Program perhitungan keuntungan suatu perusahaan ini hanya untuk satu perusahaan saja. Bagaimana cara menghitung sekaligus keuntungan beberapa bulan untuk beberapa perusahaan.(Setiap perusahaan bisa saja memiliki jumlah bulan berbeda-beda)? Buktikan

dengan program!

```
for (int i = 0; i < JmlPerusahaan; i++) {
  System.out.println(x:"============================");
  System.out.println("Perusahaan " + (i + 1));
  System.out.print(s:"Masukan jumlah bulan: ");
  int bulan = sc.nextInt();
  sc.nextLine();
  perusahaan[i] = new Sum04(bulan);

  for (int j = 0; j < perusahaan[i].elemen; j++) {
    System.out.println("Masukan untung bulan ke - " + (j + 1) + " = ");
    perusahaan[i].keuntungan[j] = sc.nextDouble();
  }
}


System.out.println(x:"============================");
System.out.println(x:"Total Keuntungan setiap Perusahaan");
for (int i = 0; i < JmlPerusahaan; i++) {
  System.out.println(x:"============================");
  System.out.println("Perusahaan " + (i + 1));
  System.out.println(x:"Algoritma Brute Force");
  System.out.println("Total keuntungan perusahaan selama " + perusahaan[
  System.out.println(x:"Algoritma Divide Conquer");
  System.out.println("Total keuntungan perusahaan selama " + perusahaan[
}
```

```
============================
Total Keuntungan setiap Perusahaan
============================
Perusahaan 1
Algoritma Brute Force
Total keuntungan perusahaan selama 4 bulan adalah = 23.3
Algoritma Divide Conquer
Total keuntungan perusahaan selama 4 bulan adalah = 23.3
============================
Perusahaan 2
Algoritma Brute Force
Total keuntungan perusahaan selama 2 bulan adalah = 9.0
Algoritma Divide Conquer
Total keuntungan perusahaan selama 2 bulan adalah = 9.0
============================
Perusahaan 3
Algoritma Brute Force
Total keuntungan perusahaan selama 6 bulan adalah = 29.200000000000003
Algoritma Divide Conquer
Total keuntungan perusahaan selama 6 bulan adalah = 29.200000000000003
```

## 4.5 Latihan Praktikum

1. Sebuah showroom memiliki daftar mobil dengan data sesuai tabel di bawah ini

| merk | tipe | tahun | top_acceleration | top_power |
|---|---|---|---|---|
| BMW | M2 Coupe | 2016 | 6816 | 728 |
| Ford | Fiesta ST | 2014 | 3921 | 575 |
| Nissan | 370Z | 2009 | 4360 | 657 |
| Subaru | BRZ | 2014 | 4058 | 609 |
| Subaru | Impreza WRX STI | 2013 | 6255 | 703 |
| Toyota | AE86 Trueno | 1986 | 3700 | 553 |
| Toyota | 86/GT86 | 2014 | 4180 | 609 |
| Volkswagen | Golf GTI | 2014 | 4180 | 631 |

Tentukan:
a) top_acceleration tertinggi menggunakan Divide and Conquer!
b) top_acceleration terendah menggunakan Divide and Conquer!
c) Rata-rata top_power dari seluruh mobil menggunakan Brute Force!

```java
public class Showroom04 {
  public static class Mobil {
    String merk;
    String tipe;
    int tahun;
    int top_acceleration;
    int top_power;

    public Mobil(String merk, String tipe, int tahun, int top_acceleration, int top_power) {
        this.merk = merk;
        this.tipe = tipe;
        this.tahun = tahun;
        this.top_acceleration = top_acceleration;
        this.top_power = top_power;
    }
}


public static int MaxTopAcceleration(Mobil[] mobil, int L, int H) {
    if (L == H) {
        return mobil[L].top_acceleration;
    }

    int mid = (L + H) / 2;
    int leftMax = MaxTopAcceleration(mobil, L, mid);
    int rightMax = MaxTopAcceleration(mobil, mid + 1, H);

    return Math.max(leftMax, rightMax);
}
```

```java
public static int MinTopAcceleration(Mobil[] mobil, int L, int H) {
    if (L == H) {
        return mobil[L].top_acceleration;
    }

    int mid = (L + H) / 2;
    int leftMin = MinTopAcceleration(mobil, L, mid);
    int rightMin = MinTopAcceleration(mobil, mid + 1, H);

    return Math.min(leftMin, rightMin);
}


public static double rataTopPower(Mobil[] mobil) {
    int totalTopPower = 0;
    for (Mobil car : mobil) {
        totalTopPower += car.top_power;
    }
    return (double) totalTopPower / mobil.length;
}
```

```java
public static void main(String[] args) {

  Showroom04.Mobil[] mobil = {
    new Showroom04.Mobil(merk:"BMW", tipe:"M2 Coupe", tahun:2016, top_acceleration:6816, top_power:728),
    new Showroom04.Mobil(merk:"Ford", tipe:"Fiesta ST", tahun:2014, top_acceleration:3921, top_power:575),
    new Showroom04.Mobil(merk:"Nissan", tipe:"370Z", tahun:2009, top_acceleration:4360, top_power:657),
    new Showroom04.Mobil(merk:"Subaru", tipe:"BRZ", tahun:2014, top_acceleration:4058, top_power:609),
    new Showroom04.Mobil(merk:"Subaru", tipe:"Impreza WRX STI", tahun:2013, top_acceleration:6255, top_power:703),
    new Showroom04.Mobil(merk:"Toyota", tipe:"AE86 Trueno", tahun:1986, top_acceleration:3700, top_power:553),
    new Showroom04.Mobil(merk:"Toyota", tipe:"86/GT86", tahun:2014, top_acceleration:4180, top_power:609),
    new Showroom04.Mobil(merk:"Volkswagen", tipe:"Golf GTI", tahun:2014, top_acceleration:4180, top_power:631)
};

int maxTopAcceleration = Showroom04.MaxTopAcceleration(mobil, l:0, mobil.length - 1);
System.out.println("Top Acceleration Tertinggi: " + maxTopAcceleration);


int minTopAcceleration = Showroom04.MinTopAcceleration(mobil, l:0, mobil.length - 1);
System.out.println("Top Acceleration Terendah: " + minTopAcceleration);


double averageTopPower = Showroom04.rataTopPower(mobil);
System.out.println("Rata-rata Top Power: " + averageTopPower);
}
```

```
howroom04'
Top Acceleration Tertinggi: 6816
Top Acceleration Terendah: 3700
Rata-rata Top Power: 633.125
```