LAPORAN TUGAS PRAKTIKUM
BRUTE FORCE DAN DIVIDE CONQUER

Oleh:

FAWWAZ ALIFIO FARSA

NIM. 2341720128

TI-1E / 10

D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

Praktikum Percobaan 1

Verifikasi Hasil Percobaan

```
fiota@LAPTOP-NATDJTJ6 MINGW64 /c/Pio's/College/Semester 2/Algoritma dan Struktur Data (master)
$  /usr/bin/env C:\\Program\ Files\\Java\\jdk-20\\bin\\java.exe -XX:+ShowCodeDetailsInExceptionM
essages -cp C:\\Users\\fiota\\AppData\\Roaming\\Code\\User\\workspaceStorage\\af0b9ed7e0f2d784a5
4a7bc4ade81d55\\redhat.java\\jdt_ws\\Algoritma\ dan\ Struktur\ Data_ead2f687\\bin Jobsheet4.Main
Faktorial
--------------------
Masukkan jumlah elemen: 3
Masukkan nilai data ke-1: 5
Masukkan nilai data ke-2: 8
Masukkan nilai data ke-3: 3
HASIL - BRUTE FORCE
Hasil penghitungan faktorial menggunakan Brute Force adalah 120
Hasil penghitungan faktorial menggunakan Brute Force adalah 40320
Hasil penghitungan faktorial menggunakan Brute Force adalah 6
HASIL - DIVIDE AND CONQUER
Hasil penghitungan faktorial menggunakan Divide and Conquer adalah 120
Hasil penghitungan faktorial menggunakan Divide and Conquer adalah 40320
Hasil penghitungan faktorial menggunakan Divide and Conquer adalah 6
```

Jawaban Pertanyaan Praktikum Percobaan 1

1. Pada bagian base line algoritma Divide Conquer terdapat perbedaan signifikan dalam penggunaan **if** dan **else**. Kondisi **if** memeriksa apakah nilai **n** sama dengan 1, dan jika ya, maka akan mengembalikan nilai 1 sebagai hasil faktorial. Kondisi **else** dilakukan ketika nilai **n** tidak sama dengan 1. Kondisi **else** akan menghitung **fakto** dengan mengalikan **n** dengan hasil rekursif dari **faktorialDC(n-1)**, dan pada akhirnya akan mengembalikan nilai **fakto** sebagai hasil faktorial.
2. Ya, perulangan pada method **faktorialBF()** dapat diubah selain menggunakan for, sebagai contoh alternatifnya dengan menggunakan while.

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

3. Perbedaan utamnya terdapat pada cara dalam memperbarui nilai **fakto**. Kode **fakto *= i;** tidak melibatkan rekursi, sedangkan kode **int fakto = n * faktorialDC(n-1);** menggunakan rekursi.

Praktikum Percobaan 2

Verifikasi Hasil Percobaan

```
fiota@LAPTOP-NATDJTJ6 MINGW64 /c/Pio's/College/Semester 2/Algoritma dan Struktur Data (master)
$  /usr/bin/env C:\\Program\ Files\\Java\\jdk-20\\bin\\java.exe -XX:+ShowCodeDetailsInExceptionM
essages -cp C:\\Users\\fiota\\AppData\\Roaming\\Code\\User\\workspaceStorage\\af0b9ed7e0f2d784a5
4a7bc4ade81d55\\redhat.java\\jdt_ws\\Algoritma\ dan\ Struktur\ Data_ead2f687\\bin Jobsheet4.Main
Pangkat
--------------------
Masukkan jumlah elemen yang dihitung: 2
Masukkan nilai yang hendak dipangkatkan: 6
Masukkan nilai pemangkat: 2
Masukkan nilai yang hendak dipangkatkan: 4
Masukkan nilai pemangkat: 3
HASIL PANGKAT - BRUTE FORCE
Hasil dari 6 pangkat 2 adalah 36
Hasil dari 4 pangkat 3 adalah 64
HASIL PANGKAT - DIVIDE AND CONQUER
Hasil dari 6 pangkat 2 adalah 36
Hasil dari 4 pangkat 3 adalah 64
```

Jawaban Pertanyaan Praktikum Percobaan 2

1. Perbedaan 2 method yaitu **pangkatBF()** menghitung pangkat secara langsung dengan perulangan dan mengalikan bilangan dasar dengan dirinya sendiri sebanyak nilai pangkat (**O(n)**), sedangkan **pangkatDC()** membagi masalah menjadi sub-masalah yang lebih kecil dan menghitung pangkat dengan rekursi (**O(log n)**).
2. Ya, terdapat tahap combine di dalam kode tersebut.

   ```
   return (pangkatDC(a, n/2) * pangkatDC(a, n/2));
   ```

3. Sebelum modifikasi

   ```
   for (int i = 0; i < elemen; i++) {
       png[i] = new Pangkat();
       System.out.print(s:"Masukkan nilai yang hendak dipangkatkan: ");
       int nilai = sc.nextInt();
       png[i].nilai = nilai;
       System.out.print(s:"Masukkan nilai pemangkat: ");
       int pangkat = sc.nextInt();
       png[i].pangkat = pangkat;
   }
   ```

   Setelah modifikasi agar pengisian atribut dilakukan dengan konstruktor:

   ```
   for (int i = 0; i < elemen; i++) {
       System.out.print(s:"Masukkan nilai yang hendak dipangkatkan: ");
       int nilai = sc.nextInt();
       System.out.print(s:"Masukkan nilai pemangkat: ");
       int pangkat = sc.nextInt();
       png[i] = new Pangkat(nilai, pangkat);
   }
   ```

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

```
switch (pilihan) {
    case 1:
        System.out.println(x:"HASIL PANGKAT - BRUTE FORCE");
        for (int i = 0; i < elemen; 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 AND CONQUER");
        for (int i = 0; i < elemen; i++) {
            System.out.println("Hasil dari " + png[i].nilai + " pangkat
            " + png[i].pangkat + " adalah " + png[i].pangkatDC(png[i].
            nilai, png[i].pangkat));
        }
        break;
    default:
        System.out.println(x:"Pilihan tidak valid!");
}
```

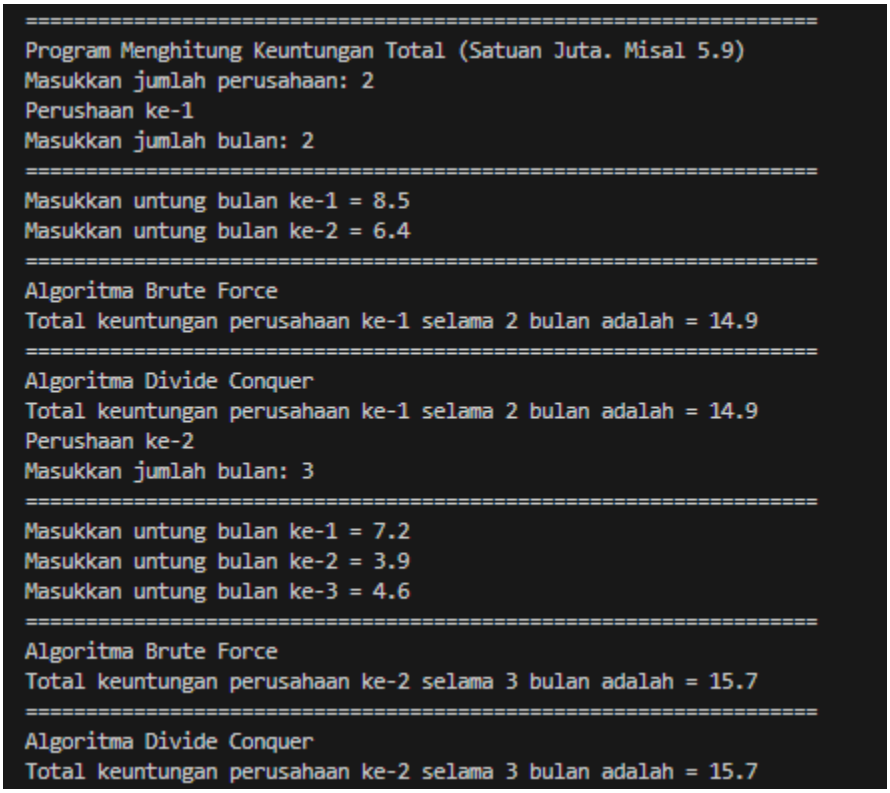
Praktikum Percobaan 3

Verifikasi Hasil Percobaan

```
fiota@LAPTOP-NATDJTJ6 MINGW64 /c/Pio's/College/Semester 2/Algoritma dan Struktur Data (master)
$  /usr/bin/env C:\\Program\ Files\\Java\\jdk-20\\bin\\java.exe -XX:+ShowCodeDetailsInExceptionM
essages -cp C:\\Users\\fiota\\AppData\\Roaming\\Code\\User\\workspaceStorage\\af0b9ed7e0f2d784a5
4a7bc4ade81d55\\redhat.java\\jdt_ws\\Algoritma\ dan\ Struktur\ Data_ead2f687\\bin Jobsheet4.Main
Sum
================================================================
Program Menghitung Keuntungan Total (Satuan Juta. Misal 5.9)
Masukkan jumlah bulan: 5
================================================================
Masukkan untung bulan ke - 1 = 8.5
Masukkan untung bulan ke - 2 = 9.54
Masukkan untung bulan ke - 3 = 7.2
Masukkan untung bulan ke - 4 = 9.1
Masukkan 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
```

Jawaban Pertanyaan Praktikum Percobaan 3

1. Formulasi kode **return lsum + arr[mid] + rsum;** adalah dengan menghitung jumlah elemen di sub-array kiri (**lsum**) yang diperoleh dari panggilan rekursif **totalDC(arr, l, mid-1)** dan sub-array kanan (**rsum**) yang diperoleh dari panggilan rekursif **totalDC(arr, mid+1, r)**, dan kemudian menggabungkan jumlah sub-array ini dengan elemen tengah (**arr[mid]**) untuk mendapatkan jumlah total seluruh sub-array.
2. Variabel **mid** dalam fungsi **totalDC()** dibutuhkan karena merupakan inti dari pendekatan yang digunakan untuk menentukan titik tengah dari sub-array yang sedang diproses.
3. Output setelah modifikasi:



Tugas

1. Menentukan:
   a. **top_acceleration** tertinggi menggunakan Divide and Conquer

```java
public int highestTopAccelerationDC(Showroom[] showroom, int low, int high)
{
    if (low == high) {
        return showroom[low].top_acceleration;
    }

    int mid = low + (high - low) / 2;
    int leftHighest = highestTopAccelerationDC(showroom, low, mid);
    int rightHighest = highestTopAccelerationDC(showroom, mid + 1, high);

    return Math.max(leftHighest, rightHighest);
}
```

   b. **top_acceleration** terendah menggunakan Divide and Conquer

```java
public int lowestTopAccelerationDC(Showroom[] showroom, int low, int high) {
    if (low == high) {
        return showroom[low].top_acceleration;
    }

    int mid = low + (high - low) / 2;
    int leftLowest = lowestTopAccelerationDC(showroom, low, mid);
    int rightLowest = lowestTopAccelerationDC(showroom, mid + 1, high);

    return Math.min(leftLowest, rightLowest);
}
```

   c. Rata – rata **top_power** dari seluruh mobil menggunakan Brute Force

```java
public double averageTopPowerBF(Showroom[] showroom) {
    int totalPower = 0;
    for (Showroom car : showroom) {
        totalPower += car.top_power;
    }
    return (double) totalPower / showroom.length;
}
```

Output:

```
fiota@LAPTOP-NATDJTJ6 MINGW64 /c/Pio's/College/Semester 2/Algoritma dan Struktur Data (master)
$  cd c:\\Pio\'s\\College\\Semester\ 2\\Algoritma\ dan\ Struktur\ Data ; /usr/bin/env C:\\Progra
d784a54a7bc4ade81d55\\redhat.java\\jdt_ws\\Algoritma\ dan\ Struktur\ Data_ead2f687\\bin Jobsheet
Highest Top Acceleration (DC): 6816
Lowest Top Acceleration (DC): 3700
Average Top Power (BF): 633.125
```