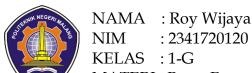
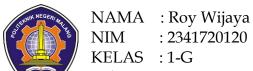


# Percobaan 1: Menghitung Nilai Faktorial dengan Algoritma Brute Force dan Divide and Conquer

```
package P4.BruteForceDivideConquer;
    public class Faktorial {
        public int nilai;
        int faktorialBF(int n) {
            int fakto = 1;
            for (int i = 1; i <= n; i++) {
                fakto = fakto * i;
11
            return fakto;
        }
13
        int faktorialDC(int n) {
            if (n == 1) {
15
                return 1;
            } else {
                int fakto = n * faktorialDC(n - 1);
                return fakto;
19
            }
        }
21
22
    }
23
```



```
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 perhitungan faktorial menggunakan Brute Forece adalah 120
Hasil perhitungan faktorial menggunakan Brute Forece adalah 40320
Hasil perhitungan faktorial menggunakan Brute Forece adalah 6
Hasil DIVIDE AND CONQUER
Hasil perhitungan faktorial menggunakan Divide and Conquer adalah 120
Hasil perhitungan faktorial menggunakan Divide and Conquer adalah 40320
Hasil perhitungan faktorial menggunakan Divide and Conquer adalah 6
PS C:\project\AlgoritmaStrukturData_1G_28\AlgoritmaStrukturData_1G_28>
```



#### Pertanyaan

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

Jawab: pada function faktorialDC menggunakan konsep fungsi recursif yaitu kode pada penggunaan if adalah base case ketika nilai n == 1 maka function tersebut mengembalikan nilai n dimana nilai n adalah 1 sedangkan kode pada penggunaan else adalah kode yang dijalankan ketika nilai n != 1 dan pada kondisi ini terjadi pemanggilan recursif dengan argumen n-1.

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

Jawab:

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

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

Jawab:

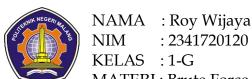
Perbedaan antara fakto \*= i; dan int fakto = n \* faktorialDC(n-1) jika dijabarkan adalah:

fakto \*= i => fakto = fakto \* i; dimana nilai i adalah 1 di dalam perulangan for pada function faktorialBF lalu perulangan tersebut akan berhenti ketika nilai i > n.

sedangkan

int fakto = n \* faktorialDC(n-1) => jika dimisalkan n adalah 5, maka

int fakto = 5 \* faktorialDC(5-1), dimana nilai n adalah argumen dari function faktorialDC, sehingga pada baris kode tersebut akan terjadi pemanggilan recursif dan akan berhenti ketika n == 1 (base case).



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

```
package <u>P</u>4.<u>B</u>rute<u>F</u>orce<u>D</u>ivide<u>C</u>onquer;
    public class Pangkat {
        public int nilai;
        public int 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 == 1) {
             } else {
                 if (n % 2 == 1) {
                     return (pangkatDC(a, n / 2) * pangkatDC(a, n / 2) * a);
                     return (pangkatDC(a, n / 2) * pangkatDC(a, n / 2));
        }
26 }
```



NAMA: Roy Wijaya NIM: 2341720120

KELAS: 1-G

MATERI : Brute Force and Divide Conquer (Jobsheet 4)

```
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 6 pangkat 3 adalah 64
```

#### Pertanyaan

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

Jawab:

Metode PangkatBF menggunakan pendekatan Brute Force untuk menghitung pangkat suatu bilangan yaitu dengan melakukan operasi perkalian sebanyak n kali terhadap bilangan a. Metode PangkatDC menggunakan pendekatan Divide and Conquer untuk memcah masalah menjadi sub-masalah yang lebih kecil yaitu dengan membagi bilangan n menjadi dua bagian yang lebih kecil.

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

Jawab:

Ya sudah termasuk, tahap combine terdapat pada code =>

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

Jawab:



NAMA: Roy Wijaya NIM: 2341720120

KELAS: 1-G

MATERI : Brute Force and Divide Conquer (Jobsheet 4)

```
package P4.BruteForceDivideConquer;

public class Pangkat {
    public int nilai;
    public int pangkat;

Pangkat(int nilai, int pangkat) {
        this.nilai = nilai;
        this.pangkat = pangkat;

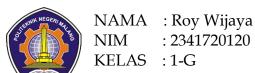
int pangkatBF(int a, int n) {
        int hasil = 1;
        for (int i = 0; i < n; i++) {
            hasil *= a;
        }
        return hasil;

}

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

}

https://documental interval interval
```



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 6 pangkat 2 adalah 36
Hasil dari 6 pangkat 3 adalah 64

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

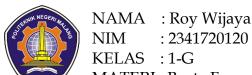
Jawab:



NAMA: Roy Wijaya NIM: 2341720120

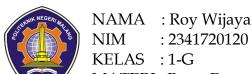
KELAS: 1-G

MATERI: Brute Force and Divide Conquer (Jobsheet 4)

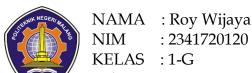


### Percobaan 3: Menghitung Sum Array dengan Algoritma Brute Force dan Divide and Conquer

```
package <u>P</u>4.<u>B</u>rute<u>F</u>orce<u>D</u>ivide<u>C</u>onquer;
import java.util.Scanner;
public class MainSum {
   public static void main(String[] args) {
       Scanner sc = new Scanner(System.in);
       System.out.println("========");
      System.out.println("Program Menghitung Keuntungan Total (Satuan Juta. Misal 5.9)");
System.out.print("Masukkan jumlah bulan: ");
       Sum sm = new Sum(elm);
       System.out.println("========
     for (int i = 0; i < sm.elemen; i++) {
         System.out.print("Masukkan untung bulan ke-" + (i + 1) + " = ");
           sm.keuntungan[i] = sc.nextDouble();
     System.out.println("=========");
       System.out.println("Algoritma Brute Force");
       System.out.println(
               "Total keuntungan perusahaan selama " + sm.elemen + " bulan adalah = " + sm.totalBF(sm.keuntungan));
       System.out.println("========="");
       System.out.println("Algoritma Divide Conquer");
      System.out.println("Total keuntungan perusahaan selama " + sm.elemen + " bulan adalah = "
               + Math.round(sm.totalDC(sm.keuntungan, 0, sm.elemen - 1) * 100.0) / 100.0);
```



```
1 package <u>P</u>4.<u>B</u>rute<u>F</u>orce<u>D</u>ivide<u>C</u>onquer;
3 public class Sum {
        int elemen;
        double keuntungan[];
        double total;
        Sum(int elemen) {
             this.elemen = elemen;
             this.keuntungan = new double[elemen];
             this.total = 0;
        }
        double totalBF(double arr[]) {
             for (int i = 0; i < elemen; i++) {</pre>
                 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 - 1);
                 double rsum = totalDC(arr, mid + 1, r);
                 return lsum + rsum + arr[mid];
             }
             return 0;
        }
32 }
```



### Pertanyaan

1. Mengapa terdapat formulasi return value berikut? Jelaskan!

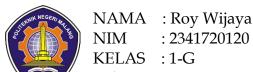
```
return lsum+rsum+arr[mid];
```

Jawab:

Return lsum + rsum + arr[mid] digunakan untuk menggabungkan solusi dari dua submasalah dengan elemen pada titik tengah untuk mendapatkan solusi total keseluruhan yang sesuai dengan konsep divide and conquer.

- 2. Kenapa dibutuhkan variable mid pada method TotalDC()? jawab: variabel mid digunakan untuk membagi array yang sedang diproses menjadi dua bagian sehingga code dapat diproses secara efisien
- 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! jawab:

```
Program Menghitung Keuntungan Total (Satuan Juta. Misal 5.9)
Banyak Perusahaan: 2
Perusahaan ke 1
Masukkan jumlah bulan: 3
Masukkan untung bulan ke-1 = 3,1
Masukkan untung bulan ke-1 = 5,3
Masukkan untung bulan ke-1 = 2,2
Algoritma Brute Force
_____
Algoritma Divide Conquer
Total keuntungan perusahaan selama 3 bulan adalah = 10.6
Perusahaan ke 2
Masukkan jumlah bulan: 4
Masukkan untung bulan ke-2 = 2,5
Masukkan untung bulan ke-2 = 2,1
Masukkan untung bulan ke-2 = 2,5
Masukkan untung bulan ke-2 = 2,1
                 -----
Algoritma Brute Force
Total keuntungan perusahaan selama 4 bulan adalah = 9.2
______
Algoritma Divide Conquer
Total keuntungan perusahaan selama 4 bulan adalah = 9.2
```



```
package P4.BruteForceDivideConquer;
  import java.util.Scanner;
  public class MainSum {
     public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
         System.out.println("-----");
        System.out.println("Program Menghitung Keuntungan Total (Satuan Juta. Misal 5.9)");
       System.out.print("Banyak Perusahaan: ");
       int jml = sc.nextInt();
       Sum sm[] = new Sum[jm1];
        for (int i = 0; i < jml; i++) {
            System.out.println("Perusahaan ke " + (i + 1));
            System.out.print("Masukkan jumlah bulan: ");
            int elm = sc.nextInt();
            sm[i] = new Sum(elm);
            System.out.println("=========");
            for (int j = 0; j < sm[i].elemen; <math>j++) {
                System.out.print("Masukkan untung bulan ke-" + (i + 1) + " = ");
                sm[i].keuntungan[j] = sc.nextDouble();
            System.out.println("=========="");
            System.out.println("Algoritma Brute Force");
            System.out.println(
                   "Total keuntungan perusahaan selama " + sm[i].elemen + " bulan adalah = "
                          + sm[i].totalBF(sm[i].keuntungan));
            System.out.println("===========");
            System.out.println("Algoritma Divide Conquer");
            System.out.println("Total keuntungan perusahaan selama " + sm[i].elemen + " bulan adalah = "
                  + Math.round(sm[i].totalDC(sm[i].keuntungan, 0, sm[i].elemen - 1) * 100.0) / 100.0);
```

LINK GITHUB: https://github.com/RoyW12/AlgoritmaStrukturData\_1G\_28