

LAPORAN PRAKTIKUM
MATA KULIAH ALGORITMA DAN STRUKTUR DATA

PERTEMUAN 5 : BRUTE FORCE AND DIVIDE CONQUER



KAYLA RACHMAUDINA SATITI PUTRI

2341760103

D-IV SISTEM INFORMASI BISNIS

JURUSAN TEKNOLOGI INFORMASI POLITEKNIK
NEGERI MALANG

2024

JOBSHEET IV

BRUTE FORCE DAN DIVIDE CONQUER

4.1 Tujuan Praktikum

Setelah melakukan materi praktikum ini, mahasiswa mampu:

1. Mahasiswa mampu membuat algoritma brute force dan divide-conquer
2. Mahasiswa mampu menerapkan penggunaan algoritma brute force dan divide-conquer

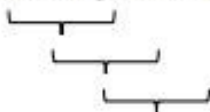
4.2 Menghitung Nilai Faktorial dengan Algoritma Brute Force dan Divide and Conquer

Perhatikan Diagram Class berikut ini :

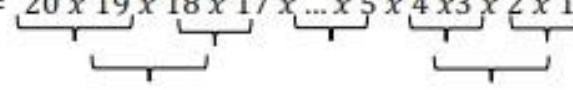
| Faktorial |
|--|
| nilai: int |
| faktorialBF(): int faktorialDC(): int |

Berdasarkan diagram class di atas, akan dibuat program class dalam Java. Untuk menghitung nilai faktorial suatu angka menggunakan 2 jenis algoritma, Brute Force dan Divide and Conquer. Jika digambarkan terdapat perbedaan proses perhitungan 2 jenis algoritma tersebut sebagai berikut :

Tahapan pencarian nilai faktorial dengan algoritma Brute Force :

$$20! = 20 \times 19 \times 18 \times 17 \times \dots \times 5 \times 4 \times 3 \times 2 \times 1$$


Tahapan pencarian nilai faktorial dengan algoritma Divide and Conquer :

$$20! = 20 \times 19 \times 18 \times 17 \times \dots \times 5 \times 4 \times 3 \times 2 \times 1$$


4.2.1 Langkah-langkah Percobaan

1. Buat Project baru, dengan nama "**BruteForceDivideConquer**". Buat package dengan nama minggu5.
2. Buatlah class baru dengan nama **Faktorial**

3. Lengkapi class **Faktorial** dengan atribut dan method yang telah digambarkan di dalam diagram class di atas, sebagai berikut:

a) Tambahkan atribut nilai

```
1 public class Faktorial15 {
2     public int nilai;
```

b) Tambahkan method faktorialBF() nilai

```
4 // menghitung faktorial dari n dengan cara mengalikan n dg semua bilangan positif dr 1 smp n
5 public int faktorialBF(int n) {
6     int fakto = 1; // dideklarasikan dengan nilai 1 krn faktorial dr 0 adlh 1, dan perkalian akan dimulai dr angka tsb
7     for (int i = 1; i <= n; i++) {
8         fakto = fakto * i;
9     }
10    return fakto;
11 }
12 }
```

c) Tambahkan method faktorialDC() nilai

```
13 // memecah masalah menjadi masalah yg lebih kecil, kemudian menyelesaikan masalah tsb secara rekursif
14 public int faktorialDC(int n) {
15     if (n == 1) {
16         return 1;
17     } else {
18         int fakto = n * faktorialDC(n-1);
19         return fakto;
20     }
21 }
22 }
```

4. Coba jalankan (Run) class **Faktorial** dengan membuat class baru **MainFaktorial**.

```
1 import java.util.Scanner;
2
3 public class FaktorialMain15 {
```

a) Di dalam fungsi main sediakan komunikasi dengan user untuk menginputkan jumlah angka yang akan dicari nilai faktorialnya

```
4 // faktorial = n!
5 public static void main(String[] args) {
6     Scanner sc = new Scanner(System.in);
7     System.out.println("=====");
8     System.out.print("Masukkan jumlah elemen yang ingin dihitung : ");
9     int elemen = sc.nextInt();
```

b) Buat Array of Objek pada fungsi main, kemudian inputkan beberapa nilai yang akan dihitung faktorialnya

```
10 Faktorial15[] fk = new Faktorial15[elemen];
11 for (int i = 0; i < elemen; i++) {
12     fk[i] = new Faktorial15();
13     System.out.print("Masukkan nilai data ke-" + (i + 1) + " : ");
14     fk[i].nilai = sc.nextInt();
15 }
```

c) Tampilkan hasil pemanggilan method faktorialDC() dan faktorialBF()

```

16      System.out.println("=====");
17      System.out.println("Hasil Faktorial dengan Brute Force");
18      for (int i = 0; i < elemen; i++) {
19          System.out.println("Faktorial dari nilai " + fk[i].nilai + " adalah : " + fk[i].faktorialBF(fk[i].nilai));
20      }
21      System.out.println("=====");
22      System.out.println("Hasil Faktorial dengan Divide and Conquer");
23      for (int i = 0; i < elemen; i++) {
24          System.out.println("Faktorial dari nilai " + fk[i].nilai + " adalah : " + fk[i].faktorialDC(fk[i].nilai));
25      }
26      System.out.println("=====");
27  }
28  }
    
```

4.2.2 Verifikasi Hasil Percobaan

```

=====
Masukkan jumlah elemen yang ingin dihitung : 3
Masukkan nilai data ke-1 : 5
Masukkan nilai data ke-2 : 8
Masukkan nilai data ke-3 : 3
=====
Hasil Faktorial dengan Brute Force
Faktorial dari nilai 5 adalah : 120
Faktorial dari nilai 8 adalah : 40320
Faktorial dari nilai 3 adalah : 6
=====
Hasil Faktorial dengan Divide and Conquer
Faktorial dari nilai 5 adalah : 120
Faktorial dari nilai 8 adalah : 40320
Faktorial dari nilai 3 adalah : 6
=====
    
```

4.2.3 Pertanyaan

1. Jelaskan mengenai base line Algoritma Divide Conquer untuk melakukan pencarian nilai faktorial!

Menggunakan rekursi untuk memecah masalah menjadi masalah yang lebih kecil, sampai akhirnya hanya tersisa faktorial dari 1. Kemudian, nilai faktorial dari angka tersebut diperoleh dengan cara mengalikan angka tersebut dengan hasil rekursif dari faktorial dari angka yang lebih kecil.

2. Pada implementasi Algoritma Divide and Conquer Faktorial apakah lengkap terdiri dari 3 tahapan divide, conquer, combine? Jelaskan masing-masing bagiannya pada kode program!

Kode program tersebut hanya terdiri dari tahapan divide dan conquer.

Divide : konsepnya ada dalam definisi fungsi faktorialDC yang menerima parameter n dan memanggil dirinya sendiri dengan parameter n-1 sampai n sama dengan 1.

Conquer : terlihat dalam blok kode yang mengeksekusi return n * faktorialDC(n-1). Setelah memecah masalah menjadi masalah-masalah yang lebih kecil, kode ini akan mengatasi setiap masalah terkecil tersebut dengan cara mengalikan n dengan hasil rekursif dari faktorial dari n-1

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

Bisa, dengan menggunakan perulangan while

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

4. Tambahkan pengecekan waktu eksekusi kedua jenis method tersebut!

```
1 import java.util.Date;
2 import java.util.Scanner;
3
4 public class FaktorialMain15 {
5     // faktorial = n!
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8         System.out.println("=====");
9         System.out.print("Masukkan jumlah elemen yang ingin dihitung : ");
10        int elemen = sc.nextInt();
11        Faktorial15[] fk = new Faktorial15[elemen];
12        for (int i = 0; i < elemen; i++) {
13            fk[i] = new Faktorial15();
14            System.out.print("Masukkan nilai data ke-" + (i + 1) + " : ");
15            fk[i].nilai = sc.nextInt();
16        }
17        System.out.println("=====");
18        System.out.println("Hasil Faktorial dengan Brute Force");
19        for (int i = 0; i < elemen; i++) {
20            long startTime = new Date().getTime();
21            long endTime = new Date().getTime();
22            System.out.println("Faktorial dari nilai " + fk[i].nilai + " adalah : " + fk[i].faktorialBF(fk[i].nilai));
23            System.out.println("Waktu eksekusi: " + (endTime - startTime) + " ms");
24        }
25        System.out.println("=====");
26        System.out.println("Hasil Faktorial dengan Divide and Conquer");
27        for (int i = 0; i < elemen; i++) {
28            long startTime = new Date().getTime();
29            long endTime = new Date().getTime();
30            System.out.println("Faktorial dari nilai " + fk[i].nilai + " adalah : " + fk[i].faktorialDC(fk[i].nilai));
31            System.out.println("Waktu eksekusi: " + (endTime - startTime) + " ms");
32        }
33        System.out.println("=====");
34    }
35 }
```

```
=====
Masukkan jumlah elemen yang ingin dihitung : 3
Masukkan nilai data ke-1 : 5
Masukkan nilai data ke-2 : 8
Masukkan nilai data ke-3 : 3
=====
Hasil Faktorial dengan Brute Force
Faktorial dari nilai 5 adalah : 120
Waktu eksekusi: 0 ms
Faktorial dari nilai 8 adalah : 40320
Waktu eksekusi: 0 ms
Faktorial dari nilai 3 adalah : 6
Waktu eksekusi: 0 ms
=====
Hasil Faktorial dengan Divide and Conquer
Faktorial dari nilai 5 adalah : 120
Waktu eksekusi: 0 ms
Faktorial dari nilai 8 adalah : 40320
Waktu eksekusi: 0 ms
Faktorial dari nilai 3 adalah : 6
Waktu eksekusi: 0 ms
=====
```

5. Buktikan dengan inputan elemen yang di atas 20 angka, apakah ada perbedaan waktu eksekusi?

```
=====
Masukkan jumlah elemen yang ingin dihitung : 2
Masukkan nilai data ke-1 : 23
Masukkan nilai data ke-2 : 25
=====
Hasil Faktorial dengan Brute Force
Faktorial dari nilai 23 adalah : 862453760
Waktu eksekusi: 0 ms
Faktorial dari nilai 25 adalah : 2076180480
Waktu eksekusi: 0 ms
=====
Hasil Faktorial dengan Divide and Conquer
Faktorial dari nilai 23 adalah : 862453760
Waktu eksekusi: 0 ms
Faktorial dari nilai 25 adalah : 2076180480
Waktu eksekusi: 0 ms
=====
```

4.3 Menghitung Hasil Pangkat dengan Algoritma Brute Force dan Divide and Conquer

Pada praktikum ini kita akan membuat program class dalam Java. Untuk menghitung nilai pangkat suatu angka menggunakan 2 jenis algoritma, Brute Force dan Divide and Conquer.

4.3.1 Langkah-langkah Percobaan

1. Di dalam paket `minggu5`, buatlah class baru dengan nama `Pangkat`. Dan di dalam class `Pangkat` tersebut, buat atribut angka yang akan dipangkatkan sekaligus dengan angka pemangkatnya

```
1 public class Pangkat15 {
2     public int nilai, pangkat;
3 }
```

2. Pada class `Pangkat` tersebut, tambahkan method `PangkatBF()`

```
3
4     public int pangkatBF(int a, int n) {
5         int hasil = 1;
6         for (int i = 0; i < n; i++) {
7             hasil = hasil * a;
8         }
9         return hasil;
10    }
```

3. Pada class `Pangkat` juga tambahkan method `PangkatDC()`

```

12     public int pangkatDC(int a, int n) {
13         if (n == 0) {
14             return 1;
15         } else {
16             if (n % 2 == 1) { // bilangan ganjil
17                 return (pangkatDC(a, n/2) * pangkatDC(a, n/2) * a);
18             } else { // bilangan genap
19                 return (pangkatDC(a, n/2) * pangkatDC(a, n/2));
20             }
21         }
22     }

```

4. Perhatikan apakah sudah tidak ada kesalahan yang muncul dalam pembuatan class Pangkat
5. Selanjutnya buat class baru yang di dalamnya terdapat method main. Class tersebut dapat dinamakan MainPangkat. Tambahkan kode pada class main untuk menginputkan jumlah nilai yang akan dihitung pangkatnya.

```

1  import java.util.Scanner;
2
3  public class PangkatMain15 {
4      public static void main(String[] args) {
5          Scanner sc = new Scanner(System.in);
6          System.out.println("=====");
7          System.out.print("Masukkan jumlah elemen yang ingin dihitung : ");
8          int elemen = sc.nextInt();
9

```

6. Nilai pada tahap 5 selanjutnya digunakan untuk instansiasi array of objek. Di dalam Kode berikut ditambahkan proses pengisian beberapa nilai yang akan dipangkatkan sekaligus dengan pemangkatnya.

```

9
10     Pangkat15[] png = new Pangkat15[elemen];
11
12     for (int i = 0; i < elemen; i++) {
13         png[i] = new Pangkat15();
14         System.out.print("Masukkan nilai yang akan dipangkatkan ke-" + (i + 1) + " : ");
15         png[i].nilai = sc.nextInt();
16         System.out.print("Masukkan nilai pemangkat ke-" + (i + 1) + " : ");
17         png[i].pangkat = sc.nextInt();
18     }
19

```

7. Kemudian, panggil hasil nya dengan mengeluarkan return value dari method PangkatBF() dan

```

20     System.out.println("=====");
21     System.out.println("Hasil pangkat dengan Brute Force");
22     for (int i = 0; i < elemen; i++) {
23         System.out.println("Nilai " + png[i].nilai + " pangkat " + png[i].pangkat + " adalah : " + png[i].pangkatBF(png[i].nilai, png[i].pangkat));
24     }
25
26     System.out.println("=====");
27     System.out.println("Hasil pangkat dengan Divide and Conquer");
28     for (int i = 0; i < elemen; i++) {
29         System.out.println("Nilai " + png[i].nilai + " pangkat " + png[i].pangkat + " adalah : " + png[i].pangkatDC(png[i].nilai, png[i].pangkat));
30     }
31 }
32 }

```

4.3.2 Verifikasi Hasil Percobaan

Pastikan output yang ditampilkan sudah benar seperti di bawah ini.

```
=====
Masukkan jumlah elemen yang ingin dihitung : 2
Masukkan nilai yang akan dipangkatkan ke-1 : 6
Masukkan nilai pemangkat ke-1 : 2
Masukkan nilai yang akan dipangkatkan ke-2 : 4
Masukkan nilai pemangkat ke-2 : 3
=====
Hasil pangkat dengan Brute Force
Nilai 6 pangkat 2 adalah : 36
Nilai 4 pangkat 3 adalah : 64
=====
Hasil pangkat dengan Divide and Conquer
Nilai 6 pangkat 2 adalah : 36
Nilai 4 pangkat 3 adalah : 64
```

4.3.3 Pertanyaan

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

`PangkatBF()` digunakan untuk pangkat yang kecil atau algoritma yang mudah dimengerti dan diimplementasikan.

2. Pada method `PangkatDC()` terdapat potongan program sebagai berikut:

```
if(n%2==1)//bilangan ganjil
    return (pangkatDC(a,n/2)*pangkatDC(a,n/2)*a);
else//bilangan genap
    return (pangkatDC(a,n/2)*pangkatDC(a,n/2));
```

Jelaskan arti potongan kode tersebut

Digunakan untuk membagi pangkat menjadi dua bagian yang lebih kecil dan menghitung pangkat dari bagian-bagian tersebut secara rekursif.

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

Tahap *combine* dalam kode tersebut terjadi ketika hasil pangkat dari dua bagian yang lebih kecil digabungkan untuk menghasilkan pangkat akhir.

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


```

1 public class Pangkat15 {
2     public int nilai, pangkat;
3
4     // menambahkan konstruktor untuk pertanyaan no.4
5     public Pangkat15(int nil, int pgkt) {
6         nilai = nil;
7         pangkat = pgkt;
8     }
9
10    public int pangkatBF(int a, int n) {
11        int hasil = 1;
12        for (int i = 0; i < n; i++) {
13            hasil = hasil * a;
14        }
15        return hasil;
16    }
17    public int pangkatDC(int a, int n) {
18        if (n == 0) {
19            return 1;
20        } else {
21            if (n % 2 == 1) { // bilangan ganjil
22                return (pangkatDC(a, n/2) * pangkatDC(a, n/2) * a);
23            } else { // bilangan genap
24                return (pangkatDC(a, n/2) * pangkatDC(a, n/2));
25            }
26        }
27    }
28 }

```

```

1 import java.util.Scanner;
2
3 public class PangkatMain15 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         // Membaca jumlah elemen
8         System.out.print("Masukkan jumlah elemen yang ingin dihitung : ");
9         int elemen = sc.nextInt();
10
11        // Membuat array untuk menyimpan objek Pangkat15
12        Pangkat15[] png = new Pangkat15[elemen];
13
14        for (int i = 0; i < elemen; i++) {
15            png[i] = new Pangkat15(i, i);
16            System.out.print("Masukkan nilai yang akan dipangkatkan ke-" + (i + 1) + " : ");
17            png[i].nilai = sc.nextInt();
18            System.out.print("Masukkan nilai pemangkat ke-" + (i + 1) + " : ");
19            png[i].pangkat = sc.nextInt();
20        }

```

```

Masukkan jumlah elemen yang ingin dihitung : 2
Masukkan nilai yang akan dipangkatkan ke-1 : 6
Masukkan nilai pemangkat ke-1 : 2
Masukkan nilai yang akan dipangkatkan ke-2 : 4
Masukkan nilai pemangkat ke-2 : 3
=====
Hasil pangkat dengan Brute Force
Nilai 6 pangkat 2 adalah : 36
Nilai 4 pangkat 3 adalah : 64
=====
Hasil pangkat dengan Divide and Conquer
Nilai 6 pangkat 2 adalah : 36
Nilai 4 pangkat 3 adalah : 64

```

5. Tambahkan menu agar salah satu method yang terpilih saja yang akan dijalankan!

```

21
22     System.out.println("=====");
23     System.out.println("1. Brute Force\n2. Divide and Conquer");
24     System.out.println("=====");
25     System.out.print("Pilih metode yang anda inginkan : ");
26     int metode = sc.nextInt();
27     switch (metode) {
28         case 1:
29             System.out.println("=====");
30             System.out.println("Hasil pangkat dengan Brute Force");
31             for (int i = 0; i < elemen; i++) {
32                 System.out.println("Nilai " + png[i].nilai + " pangkat " + png[i].pangkat + " adalah : " + png[i].pangkatBF(png[i].nilai, png[i].pangkat));
33             }
34             break;
35         case 2:
36             System.out.println("=====");
37             System.out.println("Hasil pangkat dengan Divide and Conquer");
38             for (int i = 0; i < elemen; i++) {
39                 System.out.println("Nilai " + png[i].nilai + " pangkat " + png[i].pangkat + " adalah : " + png[i].pangkatDC(png[i].nilai, png[i].pangkat));
40             }
41             break;
42     }
43     System.out.println("=====");
44 }
45 }

```

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

Di dalam percobaan ini, kita akan mempraktekkan bagaimana proses *divide, conquer*, dan *combine* diterapkan pada studi kasus penjumlahan keuntungan suatu perusahaan dalam beberapa bulan.

4.4.1 Langkah-langkah Percobaan

1. Pada paket minggu 5. Buat class baru yaitu class `Sum`. Di dalam class tersebut terdapat beberapa atribut jumlah elemen array, array, dan juga total. Tambahkan pula konstruktor pada class `Sum`.

```

1  public class Sum15 {
2      public int elemen;
3      public double keuntungan[];
4      public double total;
5
6      Sum15(int elemen) {
7          this.elemen = elemen;
8          this.keuntungan = new double[elemen];
9          this.total = 0;
10     }

```

2. Tambahkan method `TotalBF()` yang akan menghitung total nilai array dengan cara *iterative*.

```

12     double totalBF(double arr[]) {
13         for (int i = 0; i < elemen; i++) {
14             total = total + arr[i];
15         }
16         return total;
17     }

```

3. Tambahkan pula method `TotalDC()` untuk implementasi perhitungan nilai total array menggunakan algoritma Divide and Conquer

```

19     double totalDC(double arr[], int l, int r) {
20         if (l == r) {
21             return arr[l];
22         } else if (l < r) {
23             int mid = (l+r) / 2;
24             double lsum = totalDC(arr, l, mid-1);
25             double rsum = totalDC(arr, mid+1, r);
26             return lsum + rsum + arr[mid];
27         }
28         return 0;
29     }
30 }
    
```

4. Buat class baru yaitu `MainSum`. Di dalam kelas ini terdapat method `main`. Pada method ini user dapat menuliskan berapa bulan keuntungan yang akan dihitung. Dalam kelas ini sekaligus dibuat instansiasi objek untuk memanggil atribut ataupun fungsi pada class `Sum`

```

1  import java.util.Scanner;
2
3  public class MainSum15 {
4      public static void main(String[] args) {
5          Scanner sc = new Scanner(System.in);
6          System.out.println("=====");
7          System.out.println("Program menghitung keuntungan total (satuan juta. misal 5.9)");
8          System.out.print("Masukkan jumlah bulan : ");
9          int elm = sc.nextInt();
    
```

5. Karena yang akan dihitung adalah total nilai keuntungan, maka ditambahkan pula pada method `main` mana array yang akan dihitung. Array tersebut merupakan atribut yang terdapat di class `Sum`, maka dari itu dibutuhkan pembuatan objek `Sum` terlebih dahulu.

```

10
11     Sum15 sm = new Sum15(elm);
12     System.out.println("=====");
13     for (int i = 0; i < sm.elemen; i++) {
14         System.out.print("Masukkan untung bulan ke-" + (i+1) + " : ");
15         sm.keuntungan[i] = sc.nextDouble();
16     }
17
    
```

6. Tampilkan hasil perhitungan melalui objek yang telah dibuat untuk kedua cara yang ada (Brute Force dan Divide and Conquer)

```

17
18     System.out.println("=====");
19     System.out.println("Algoritma Brute Force");
20     System.out.println("Total keuntungan perusahaan selama " + sm.elemen + " bulan adalah : " + sm.totalBF(sm.keuntungan));
21     System.out.println("=====");
22     System.out.println("Algoritma Divide and Conquer");
23     System.out.println("Total keuntungan perusahaan selama " + sm.elemen + " bulan adalah : " + sm.totalDC(sm.keuntungan, 0, sm.elemen - 1));
24 }
25 }
    
```

4.4.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.

```
=====
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 and Conquer
Total keuntungan perusahaan selama 5 bulan adalah : 40.34
```

4.4.3 Pertanyaan

1. Berikan ilustrasi perbedaan perhitungan keuntungan dengan method `TotalBF()` ataupun `TotalDC()`

TotalBF (Brute Force):

Metode ini menghitung total keuntungan dengan menjumlahkan satu per satu elemen dalam array. Iterasi dimulai dari indeks 0 hingga elemen terakhir (elemen-1). Di setiap iterasi, nilai elemen ditambahkan ke variabel total.

TotalDC (Divide and Conquer):

Metode ini membagi masalah menjadi sub-masalah yang lebih kecil. Pertama, ia memeriksa apakah indeks kiri (l) sama dengan indeks kanan (r). Jika iya, berarti hanya ada 1 elemen dan langsung dikembalikan. Jika indeks kiri lebih kecil dari kanan, maka hitung titik tengah (mid) dari interval indeks $(l + r)$ dibagi 2. Rekursif panggil fungsi `totalDC` untuk menghitung total keuntungan di bagian kiri array (l sampai mid-1). Rekursif panggil fungsi `totalDC` untuk menghitung total keuntungan di bagian kanan array (mid+1 sampai r). Kemudian menjumlahkan total keuntungan dari kedua bagian (lsum dan rsum) dan menambahkan keuntungan di elemen tengah (`arr[mid]`).

2. Perhatikan output dari kedua jenis algoritma tersebut bisa jadi memiliki hasil berbeda di belakang koma. Bagaimana membatasi output di belakang koma agar menjadi standar untuk kedua jenis algoritma tersebut.

```

17
18     System.out.println("=====");
19     System.out.println("Algoritma Brute Force");
20     double formattedBF = Double.parseDouble(sm.formatDouble(sm.totalBF(sm.keuntungan)));
21     System.out.println("Total keuntungan perusahaan selama " + sm.elemen + " bulan adalah : " + formattedBF);
22     // System.out.println("Total keuntungan perusahaan selama " + sm.elemen + " bulan adalah : " + sm.totalBF(sm.keuntungan));
23     System.out.println("=====");
24     System.out.println("Algoritma Divide and Conquer");
25     // System.out.println("Total keuntungan perusahaan selama " + sm.elemen + " bulan adalah : " + sm.totalDC(sm.keuntungan, 0, sm.elemen - 1));
26     double formattedDC = Double.parseDouble(sm.formatDouble(sm.totalDC(sm.keuntungan, 0, sm.elemen-1)));
27     System.out.println("Total keuntungan perusahaan selama " + sm.elemen + " bulan adalah : " + formattedDC);
28 }
29

```

```

1 public String formatDouble(double number) {
2     DecimalFormat df = new DecimalFormat("#.##");
3     return df.format(number);
4 }

```

```

=====
Algoritma Brute Force
Total keuntungan perusahaan selama 5 bulan adalah : 40.34
=====
Algoritma Divide and Conquer
Total keuntungan perusahaan selama 5 bulan adalah : 40.34

```

3. Mengapa terdapat formulasi *return value* berikut?Jelaskan!

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

Untuk mencari sub-array dengan total nilai terbesar

4. Kenapa dibutuhkan variable `mid` pada method `TotalDC()` ?

Untuk menyimpan jumlah elemen dalam `dc`, sehingga dapat digunakan untuk membagi jumlah semua nilai dalam `dc` dan menghitung rata-rata

5. 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!

```

1  import java.util.Scanner;
2
3  public class MainSumIS {
4      public static void main(String[] args) {
5          try (Scanner sc = new Scanner(System.in)) {
6              System.out.println("=====");
7              System.out.println("Program Menghitung Keuntungan Total (Satuan Juta. Misal 5,9)");
8
9              System.out.print("Masukkan jumlah bulan : ");
10             int elm = sc.nextInt();
11
12             SumIS[] sm = new SumIS[elm];
13
14             for (int i = 0; i < elm; i++) {
15                 System.out.print("Masukkan jumlah bulan untuk perusahaan ke-" + (i + 1) + " : ");
16                 while (!sc.hasNextInt()) {
17                     System.out.println("Invalid input. Please enter an integer value.");
18                     sc.next();
19                 }
20                 int month = sc.nextInt();
21                 sm[i] = new SumIS(month);
22
23                 System.out.println("Masukkan keuntungan untuk perusahaan ke-" + (i + 1));
24                 for (int j = 0; j < sm[i].elemen; j++) {
25                     do {
26                         System.out.print("Masukkan untung bulan ke-" + (j + 1) + " : ");
27                         while (!sc.hasNextDouble()) {
28                             System.out.println("Invalid input. Please enter a decimal value.");
29                             sc.next();
30                         }
31                         sm[i].keuntungan[j] = sc.nextDouble();
32                         sc.nextLine();
33                     } while (sm[i].keuntungan[j] < 0);
34                 }
35             }
36
37             System.out.println("=====");
38             for (int i = 0; i < elm; i++) {
39                 System.out.println("Perusahaan ke-" + (i + 1));
40                 System.out.println("Algoritma Brute Force");
41                 System.out.println("Total keuntungan perusahaan selama " + sm[i].elemen + " bulan adalah = " + String.format("%.2f", sm[i].totalBF(sm[i].keuntungan)));
42                 System.out.println("Algoritma Divide Conquer");
43                 System.out.println("Total keuntungan perusahaan selama " + sm[i].elemen + " bulan adalah = " + String.format("%.2f", sm[i].totalDC(sm[i].keuntungan, 0, sm[i].elemen - 1)));
44                 System.out.println("=====");
45             }
46         }
47     }
48 }

```

4.5 Latihan Praktikum

Buatlah kode program untuk menghitung nilai akar dari suatu bilangan dengan algoritma Brute Force dan Divide Conquer! *Jika bilangan tersebut bukan merupakan kuadrat sempurna, bulatkan angka ke bawah.*

```

1  public class Akar15 {
2      public double num;
3
4      public double akar15BF(double num) {
5          double low = 0, high = num, mid;
6          while (low <= high) {
7              mid = low + (high - low) / 2;
8              if (mid * mid == num) {
9                  return mid;
10             } else if (mid * mid < num) {
11                 low = mid + 0.00001; // utk mencegah infinite loop
12             } else {
13                 high = mid - 0.00001; // utk mencegah infinite loop
14             }
15         }
16         return low;
17     }
18
19     public double akar15DC(double num, double low, double high) {
20         double mid = low + (high - low) / 2;
21         if (high - low < 0.00001) {
22             return mid;
23         }
24         if (mid * mid == num) {
25             return mid;
26         } else if (mid * mid < num) {
27             return akar15DC(num, mid, high);
28         } else {
29             return akar15DC(num, low, mid);
30         }
31     }
32 }

```

```

1  import java.util.Scanner;
2
3  public class AkarMain15 {
4      public static void main(String[] args) {
5          Scanner sc = new Scanner(System.in);
6          System.out.println("=====");
7          System.out.print("Masukkan bilangan yang ingin dihitung akarnya : ");
8          double num = sc.nextDouble();
9
10         Akar15 akar = new Akar15();
11         System.out.println("=====");
12         System.out.println("Hasil akar dengan Brute Force");
13         System.out.println("Akar dari " + num + " adalah : " + akar.akar15BF(num));
14
15         System.out.println("=====");
16         System.out.println("Hasil akar dengan Divide and Conquer");
17         System.out.println("Akar dari " + num + " adalah : " + akar.akar15DC(num, 0, num));
18     }
19 }

```