SAFRIZAL RAHMAN
19/ SIB-1G

JOBSHEET IV
BRUTE FORCE DAN DIVIDE CONQUER

4.1 Tujuan Praktikum

Setelah melakukan materi praktikum ini, mahasiswa mampu:

1. Mahasiswa mampu membuat algoritma bruteforce dan divide-conquer
2. Mahasiswa mampu menerapkan penggunaan algorima bruteforce dan divide-conquer

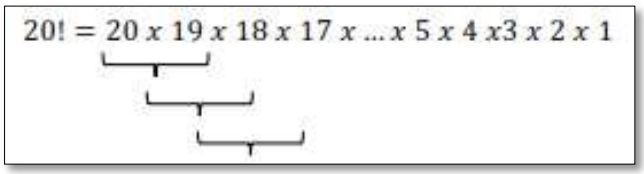
4.2 Menghitung Nilai Faktorial dengan Algoritma Brute Force dan Divide and Conquer

Perhatikan Diagram Class berikut ini :

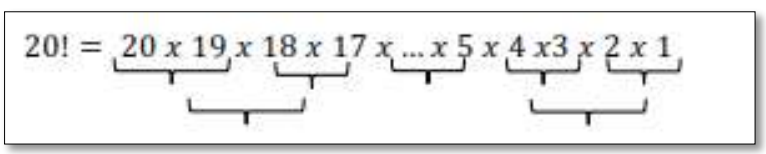
Faktorial
nilai: int
faktorialBF(): int faktorialDC(): int

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 :



Tahapan pencarian nilai faktorial dengan algoritma Divide and Conquer :



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

```
public int nilai;
```

   b) Tambahkan method `faktorialBF()` nilai

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

   c) Tambahkan method `faktorialDC()` nilai

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

4. Coba jalankan (Run) class `Faktorial` dengan membuat class baru `MainFaktorial`.
   a) Di dalam fungsi main sediakan komunikasi dengan user untuk menginputkan jumlah angka yang akan dicari nilai faktorialnya

```
Scanner sc = new Scanner(System.in);
System.out.println("=========================================================");
System.out.print("Masukkan jumlah elemen yang ingin dihitung : ");
int elemen = sc.nextInt();
```

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

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

c) Tampilkan hasil pemanggilan method `aktorialDC()` dan `faktorialBF()`

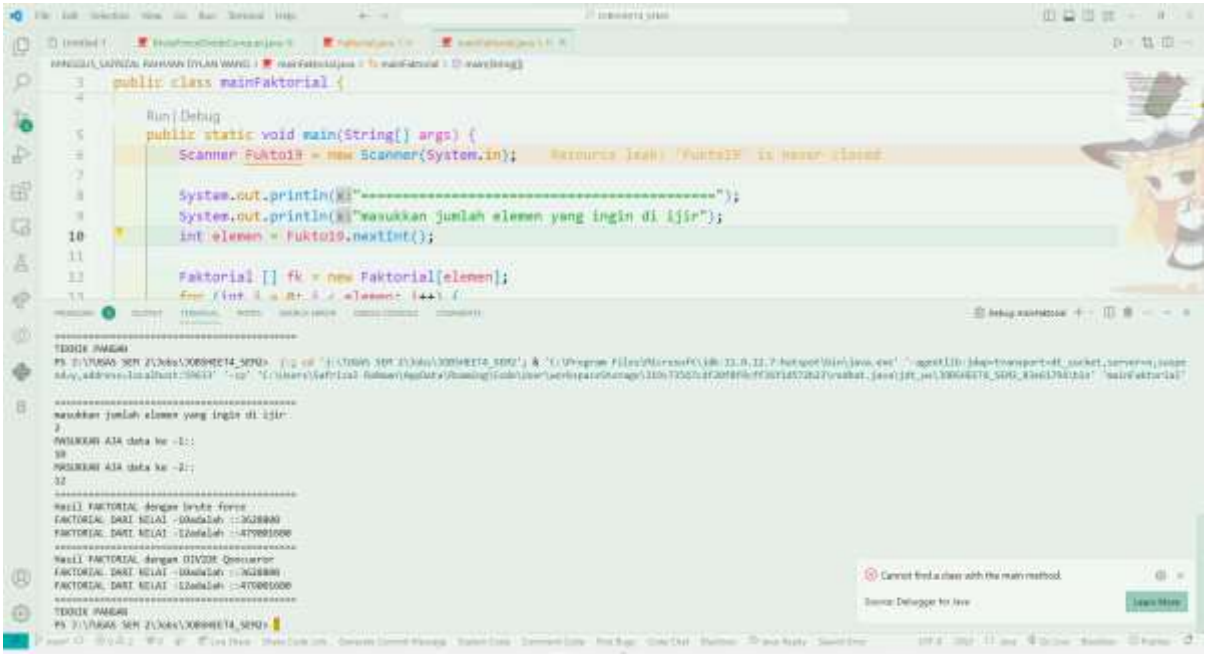
```
System.out.println("=====================================================");
System.out.println("Hasil Faktorial dengan Brute Force");
for (int i = 0; i < elemen; i++) {
    System.out.println("Faktorial dari nilai "+fk[i].nilai+" adalah : "+fk[i].faktorialBF(fk[i].nilai));
}
System.out.println("=====================================================");
System.out.println("Hasil Faktorial dengan Divide and Conquer");
for (int i = 0; i < elemen; i++) {
    System.out.println("Faktorial dari nilai "+fk[i].nilai+" adalah : "+fk[i].faktorialDC(fk[i].nilai));
}
System.out.println("=====================================================");
```

d) Pastikan program sudah berjalan dengan baik!

```
5. import java.util.Scanner;
6.
7. public class mainFaktorial {
8.
9.     public static void main(String[] args) {
10.         Scanner Fukto19 = new Scanner(System.in);
11.
12.         System.out.println("========================
    ======================");
13.         System.out.println("masukkan jumlah elemen
    yang ingin di ijir");
14.         int elemen = Fukto19.nextInt();
15.
16.         Faktorial [] fk = new Faktorial[elemen];
17.         for (int i = 0; i < elemen; i++) {
18.             fk[i] = new Faktorial();
19.             System.out.println("MASUKKAN AJA data
    ke -"+ (i+1)+ "::");
20.             fk[i].nilai = Fukto19.nextInt();
21.
22.         }
```

```
23.
24.         System.out.println("==========================
  ==========================");
25.         System.out.println("Hasil FAKTORIAL dengan
  brute force");
26.         for (int i = 0; i < elemen; i++) {
27.             System.out.println("FAKTORIAL DARI
  NILAI -"+fk[i].nilai+ "adalah
  ::"+fk[i].faktorialBF(fk[i].nilai));
28.         }
29.
30.         System.out.println("==========================
  ==========================");
31.         System.out.println("Hasil FAKTORIAL dengan
  DIVIDE Qoncueror");
32.         for (int i = 0; i < elemen; i++) {
33.             System.out.println("FAKTORIAL DARI
  NILAI -"+fk[i].nilai+ "adalah
  ::"+fk[i].faktorialDC(fk[i].nilai));
34.         }
35.
36.         System.out.println("==========================
  ==========================");
37.         System.out.println("TEKNIK PANGAN");
38.         // for (int i = 0; i < elemen; i++) {
39.         //     System.out.println("FAKTORIAL DARI
  NILAI -"+fk[i].nilai+ "adalah
  ::"+fk[i].faktorialBF(fk[i].nilai));
40.         // }
41.
42.     }
43.     // int elemen = int.nextInt(0);
44.
45. }
```

46.



4.2.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.

```
run:
=====================================================
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
=====================================================
BUILD SUCCESSFUL (total time: 7 seconds)
```

4.2.3 Pertanyaan

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

   Dasar dari Algoritma Divide and Conquer untuk Perhitungan Faktorial: Dasar dari algoritma Divide and Conquer untuk penghitungan faktorial adalah memecah masalah menjadi submasalah yang lebih kecil hingga menjadi cukup sederhana untuk diselesaikan secara langsung. Untuk perhitungan faktorial, masalah menghitung n! (n faktorial) dibagi menjadi

masalah yang lebih kecil untuk menghitung (n-1)!, (n-2)!, dan seterusnya, hingga kita mencapai 1! atau 0!, yang keduanya didefinisikan sebagai 1. Solusi untuk submasalah ini kemudian digabungkan untuk memberikan solusi ke masalah asli.

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

   Implementasi Divide and Conquer dalam Perhitungan Faktorial: Dalam kode yang disediakan, algoritma Divide and Conquer diimplementasikan dalam metode faktorialDC(). Namun, penting untuk dicatat bahwa tiga tahap khas Divide and Conquer (bagi, taklukkan, gabungkan) tidak sepenuhnya berlaku di sini. Hal ini karena perhitungan faktorial pada dasarnya tidak melibatkan langkah 'gabungkan'. Berikut cara kerjanya dalam kode:

   Membagi: Ini dilakukan dengan memanggil faktorialDC() secara rekursif dengan argumen yang lebih kecil (n-1).

   Conquer: Ketika kasus dasar (n <= 1) tercapai, fungsi ini mengembalikan 1.

   Menggabungkan: Seperti yang telah disebutkan, tidak ada langkah 'gabungkan' secara eksplisit. Hasilnya secara implisit digabungkan dengan operasi perkalian dalam pemanggilan rekursif (n faktorialDC(n-1)).

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

   Ya, Anda dapat menggunakan jenis perulangan lain, seperti perulangan while, untuk mencapai hasil yang sama. Berikut adalah contoh bagaimana Anda dapat memodifikasi metode faktorialBF() untuk menggunakan perulangan while:

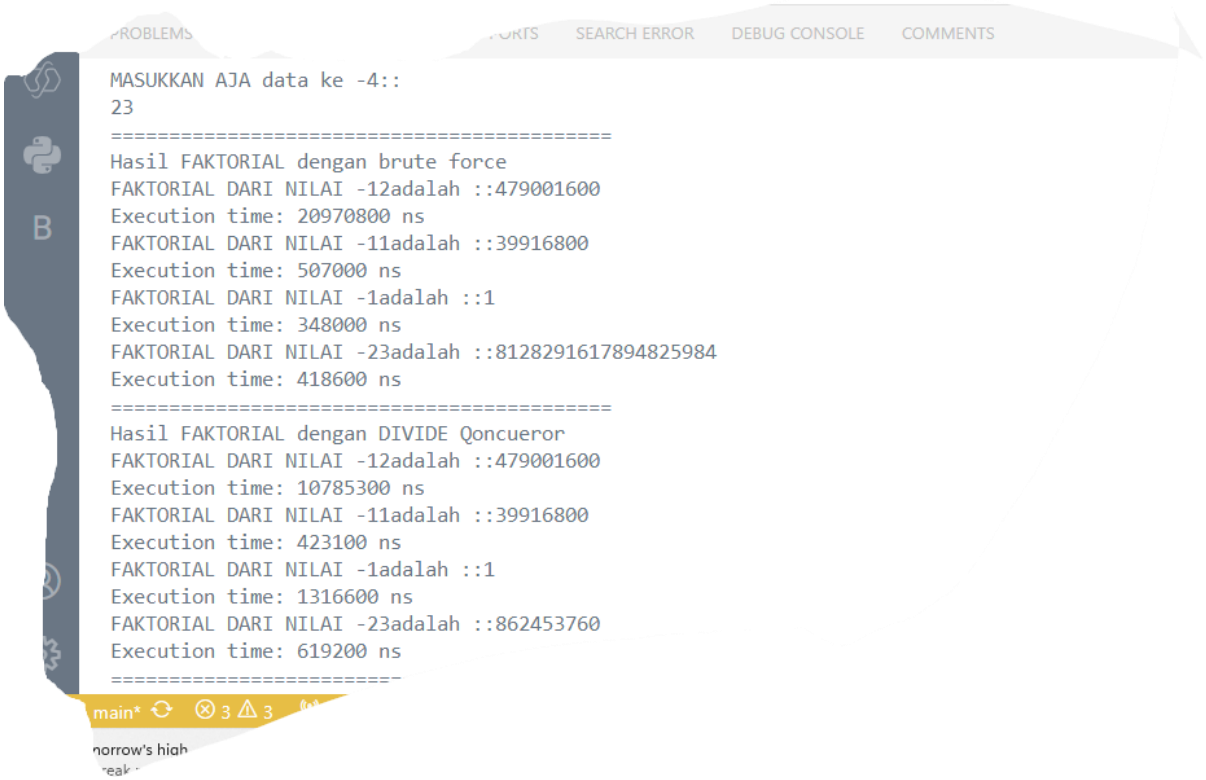
```
public long faktorialBF(int n) {
    long result = 1;
    int i = 1;
    while (i <= n) {
        result = i;
        i++;
    }
    return result;
}
```

Kode ini melakukan hal yang sama dengan metode faktorialBF() yang asli: kode ini menghitung faktorial dari n dengan mengalikan hasilnya secara berulang-ulang dengan setiap bilangan bulat dari 1 hingga n.

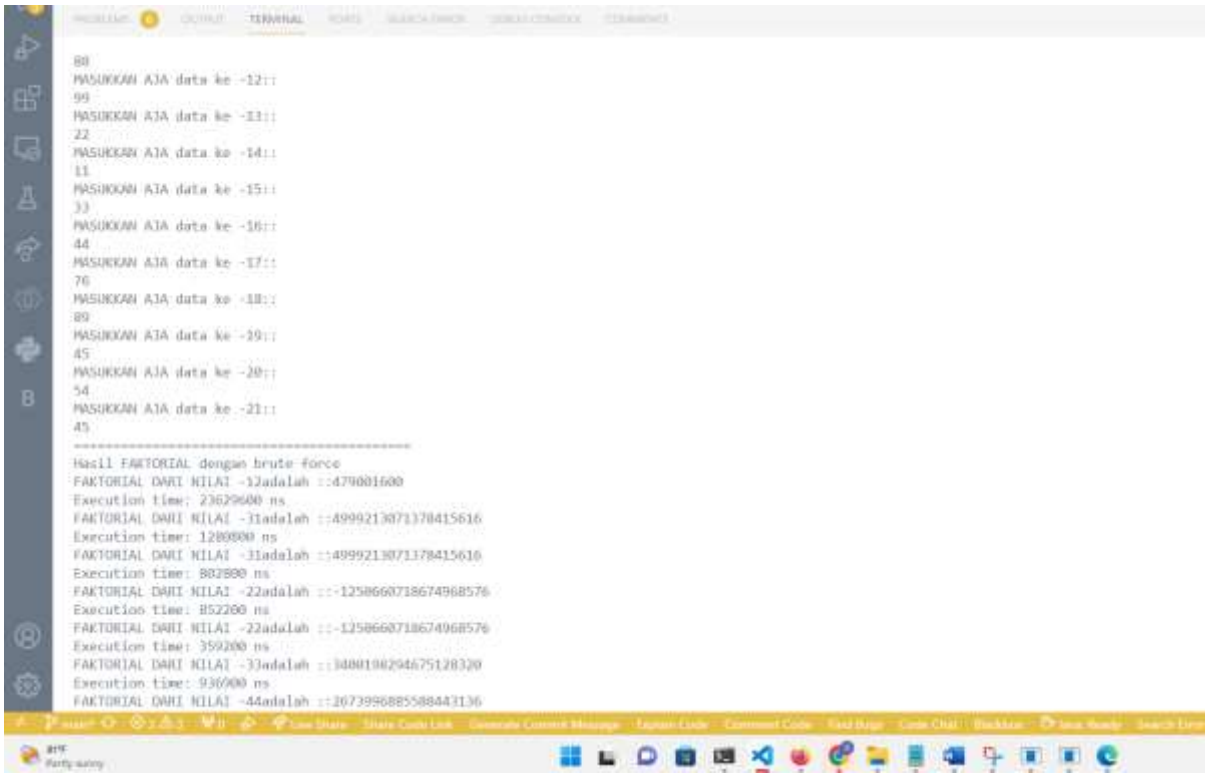
4. Tambahkan pegecekan waktu eksekusi kedua jenis method tersebut!

```
5.public class mainFaktorial {
6.    public static void main(String[] args) {
7.        Scanner Fukto19 = new Scanner(System.in);
8.
9.        System.out.println("========================
  ====================");
10.         System.out.println("masukkan jumlah elemen
  yang ingin di ijir");
11.         int elemen = Fukto19.nextInt();
12.
13.         Faktorial [] fk = new Faktorial[elemen];
14.         for (int i = 0; i < elemen; i++) {
15.             fk[i] = new Faktorial();
16.             System.out.println("MASUKKAN AJA data
  ke -"+ (i+1)+ "::");
17.             fk[i].nilai = Fukto19.nextInt();
18.         }
19.
20.         System.out.println("======================
  ======================");
21.         System.out.println("Hasil FAKTORIAL dengan
  brute force");
22.         for (int i = 0; i < elemen; i++) {
23.             long start = System.nanoTime();
24.             System.out.println("FAKTORIAL DARI
  NILAI -"+fk[i].nilai+ "adalah
  ::"+fk[i].faktorialBF(fk[i].nilai));
25.             long end = System.nanoTime();
26.             System.out.println("Execution time: "
  + (end - start) + " ns");
27.         }
28.
```

```
29.         System.out.println("==========================
  ==========================");
30.         System.out.println("Hasil FAKTORIAL dengan
  DIVIDE Qoncueror");
31.         for (int i = 0; i < elemen; i++) {
32.             long start = System.nanoTime();
33.             System.out.println("FAKTORIAL DARI
  NILAI -"+fk[i].nilai+ "adalah
  ::"+fk[i].faktorialDC(fk[i].nilai));
34.             long end = System.nanoTime();
35.             System.out.println("Execution time: "
  + (end - start) + " ns");
36.         }
37.
38.         System.out.println("==========================
  ==========================");
39.         System.out.println("TEKNIK PANGAN");
40.     }
41. }
```



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



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 nil ai 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

```
    public int nilai,pangkat;
```

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

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

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

```
public int pangkatDC(int a,int n){
    if (n==0) {
        return  1;
    }
    else
    {
        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));
    }
}
```

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.

```
Scanner sc = new Scanner(System.in);
System.out.println("=========================================================");
System.out.print("Masukkan jumlah elemen yang ingin dihitung : ");
int elemen = sc.nextInt();
```

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.

```
Pangkat [] png = new Pangkat[elemen];

for (int i = 0; i < elemen; i++) {
    png[i] = new Pangkat();
    System.out.print("Masukkan nilai yang akan dipangkatkan ke-"+(i+1)+" : ");
    png[i].nilai = sc.nextInt();
    System.out.print("Masukkan nilai pemangkat ke-"+(i+1)+" : ");
    png[i].pangkat = sc.nextInt();
}
```

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

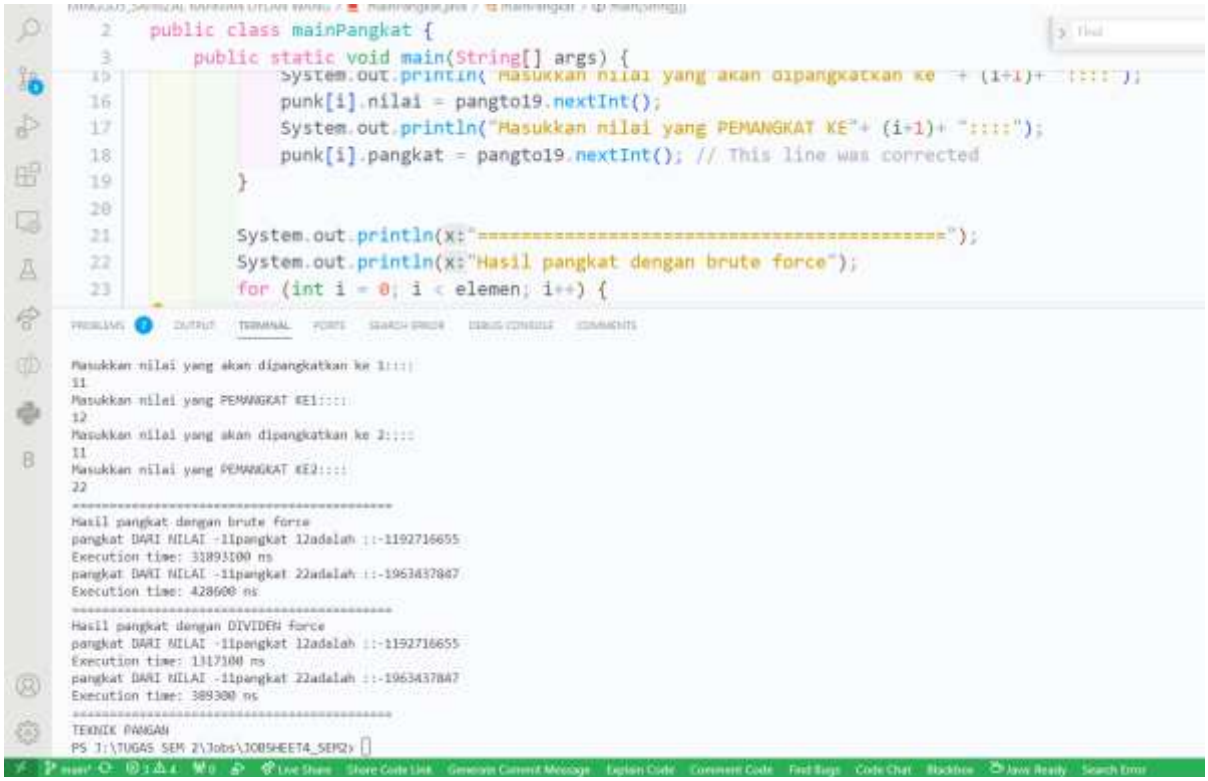
```
import java.util.Scanner;
public class mainPangkat {
    public static void main(String[] args) {

        Scanner pangto19 = new Scanner(System.in);

        System.out.println("============================
==================");
        System.out.println("masukkan jumlah elemen
yang ingin di ijir");
        int elemen = pangto19.nextInt();

        pangkatpangkatan [] punk = new
pangkatpangkatan[elemen];

        for (int i = 0; i < elemen;i++) {
            punk[i] = new pangkatpangkatan();
```

```
            System.out.println("Masukkan nilai yang
akan dipangkatkan ke "+ (i+1)+ "::::");
            punk[i].nilai = pangto19.nextInt();
            System.out.println("Masukkan nilai yang
PEMANGKAT KE"+ (i+1)+ "::::");
            punk[i].pangkat = pangto19.nextInt(); //
This line was corrected
        }

        System.out.println("==========================
================");
        System.out.println("Hasil pangkat dengan brute
force");
        for (int i = 0; i < elemen; i++) {
            long start = System.nanoTime();
            System.out.println("pangkat DARI NILAI -
"+punk[i].nilai+"pangkat " +punk[i].pangkat +"adalah
::"+punk[i].pangkatBF(punk[i].nilai,punk[i].pangkat));
            long end = System.nanoTime();
            System.out.println("Execution time: " +
(end - start) + " ns");
        }
        System.out.println("==========================
================");
        System.out.println("Hasil pangkat dengan
DIVIDEN force");
        for (int i = 0; i < elemen; i++) {
            long start = System.nanoTime();
            System.out.println("pangkat DARI NILAI -
"+punk[i].nilai+"pangkat " +punk[i].pangkat +"adalah
::"+punk[i].pangkatDC(punk[i].nilai,punk[i].pangkat));
            long end = System.nanoTime();
            System.out.println("Execution time: " +
(end - start) + " ns");
```

```
        }
        System.out.println("==========================
================");
        System.out.println("TEKNIK PANGAN");
    }
    //     System.out.println("======================
==================");
    //     System.out.println("masukkan jumlah elemen
yang ingin di ijir");
    //     int elemen = pangto19.nextInt();

    //     pangkatpangkatan [] punk = new
pangkatpangkatan[elemen];

    //     for (int i = 0; i < elemen;i++) {
    //         punk[i] = new pangkatpangkatan();
    //         System.out.println("Masukkan nilai yang
akan dipangkatkan ke "+ (i+1)+ "::::");
    //         punk[i].nilai = pangto19.nextInt();
    //         System.out.println("Masukkan nilai yang
PEMANGKAT KE"+ (i+1)+ "::::");
    //         punk[i].nilai = pangto19.nextInt();
    //         //#endregion
    //     }

    //     System.out.println("======================
==================");
    //     System.out.println("Hasil pangkat dengan
brute force");
    //     for (int i = 0; i < elemen; i++) {
    //         long start = System.nanoTime();
    //         System.out.println("pangkat DARI NILAI
-"+punk[i].nilai+"pangkat " +punk[i].pangkat +"adalah
::"+punk[i].pangkatBF(punk[i].nilai,punk[i].pangkat));
```

```
    //          long end = System.nanoTime();
    //          System.out.println("Execution time: " +
(end - start) + " ns");
    //      }
    //      System.out.println("=======================
====================");
    //      System.out.println("Hasil pangkat dengan
DIVIDEN force");
    //      for (int i = 0; i < elemen; i++) {
    //          long start = System.nanoTime();
    //          System.out.println("pangkat DARI NILAI
-"+punk[i].nilai+"pangkat " +punk[i].pangkat +"adalah
::"+punk[i].pangkatDC(punk[i].nilai,punk[i].pangkat));
    //          long end = System.nanoTime();
    //          System.out.println("Execution time: " +
(end - start) + " ns");
    //      }
    //      System.out.println("=======================
====================");
    //      System.out.println("TEKNIK PANGAN");
    // }
    }
```

4.3.2 Verifikasi Hasil Percobaan



Pastikan output yang ditampilkan sudah benar seperti di bawah ini.

```
run:
=====================================================
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
=====================================================
BUILD SUCCESSFUL (total time: 10 seconds)
```

4.3.3 Pertanyaan

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

   PangkatBF() vs PangkatDC(): Metode `PangkatBF () ` menghitung pangkat dari sebuah angka menggunakan pendekatan brute force. Metode ini mengalikan angka dasar dengan sendirinya untuk beberapa kali yang ditentukan oleh eksponen. Di sisi lain, `PangkatDC()` menggunakan pendekatan Bagi dan Taklukkan untuk menghitung pangkat suatu bilangan. Metode ini

memecah masalah menjadi submasalah yang lebih kecil dan menggabungkan hasilnya untuk mendapatkan jawaban akhir. Metode ini lebih efisien karena mengurangi jumlah perkalian.

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

Potongan Kode di PangkatDC(): Potongan kode ini memeriksa apakah eksponen `n` genap atau ganjil. Jika `n` ganjil, maka ia akan menghitung pangkat setengah dari `n`, mengkuadratkannya, lalu mengalikannya dengan basis `a`. Jika `n` genap, maka ia akan menghitung pangkat setengah dari `n` dan mengkuadratkannya. Pendekatan ini mengurangi jumlah perkalian.

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

Ya, langkah penggabungan disertakan di dalam kode. Hasil dari sub-sub masalah digabungkan ketika metode ini mengembalikan `(pangkatDC(a, n/2)pangkatDC(a, n/2)a)` atau `(pangkatDC(a, n/2)pangkatDC(a, n/2))`.

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

`pangkatpangkat` untuk menyertakan konstruktor yang menginisialisasi atribut `nilai` dan `pangkat`. Berikut adalah cara untuk melakukannya:

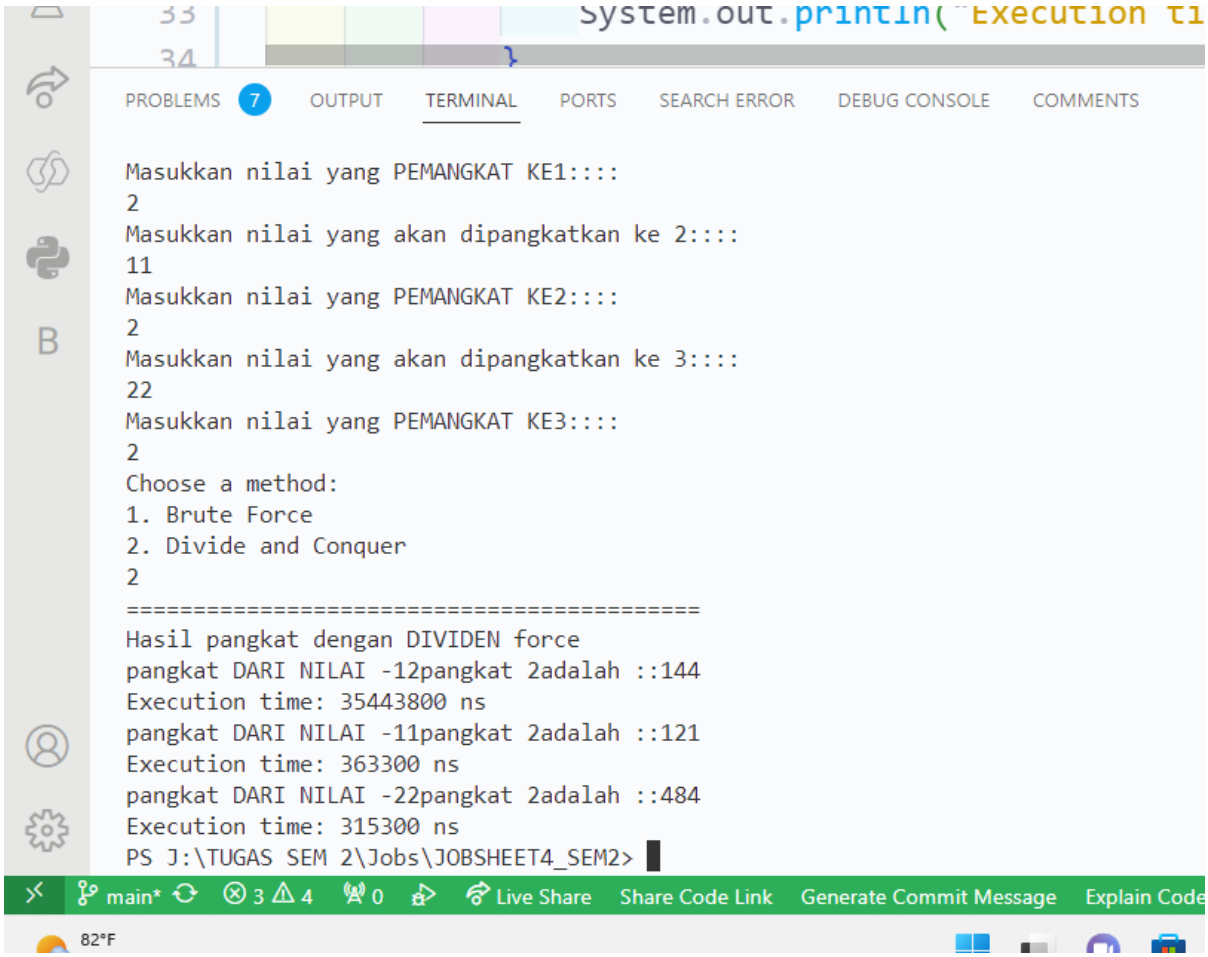
```
public pangkatpangkatan(int nilai, int pangkat) {
     this.nilai = nilai;
     this.pangkat = pangkat;
 }
```

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

```
6.    System.out.println("Choose a method:");
7.        System.out.println("1. Brute Force");
8.        System.out.println("2. Divide and Conquer");
9.        int choice = pangto19.nextInt();
10.
11.         if (choice == 1) {
12.             System.out.println("==================
  ========================");
```

```
13.             System.out.println("Hasil pangkat
  dengan brute force");
14.             for (int i = 0; i < elemen; i++) {
15.                 long start = System.nanoTime();
16.                 System.out.println("pangkat DARI
  NILAI -"+punk[i].nilai+"pangkat " +punk[i].pangkat
  +"adalah
  ::"+punk[i].pangkatBF(punk[i].nilai,punk[i].pangkat)
  );
17.                 long end = System.nanoTime();
18.                 System.out.println("Execution
  time: " + (end - start) + " ns");
19.             }
20.             // Run brute force method
21.         } else if (choice == 2) {
22.             System.out.println("==================
  ==========================");
23.             System.out.println("Hasil pangkat
  dengan DIVIDEN force");
24.             for (int i = 0; i < elemen; i++) {
25.                 long start = System.nanoTime();
26.                 System.out.println("pangkat DARI
  NILAI -"+punk[i].nilai+"pangkat " +punk[i].pangkat
  +"adalah
  ::"+punk[i].pangkatDC(punk[i].nilai,punk[i].pangkat)
  );
27.                 long end = System.nanoTime();
28.                 System.out.println("Execution
  time: " + (end - start) + " ns");
29.             }
30.             // Run divide and conquer method
31.         } else {
32.             System.out.println("Invalid choice");
```

```
33.             System.out.println("==================
  ==========================");
34.             System.out.println("TEKNIK PANGAN");
35.         }
```



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 minggu5. Buat class baru yaitu class `Sum`. DI salam class tersebut terdapat beberapa atribut jumlah elemen array, array, dan juga total. Tambahkan pula konstruktor pada class `Sum`.

```
public int elemen;
public double keuntungan[];
public double total;
```

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

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

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

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

```
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;
}
```

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`

```
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 : ");
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.

```
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();
}
```

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

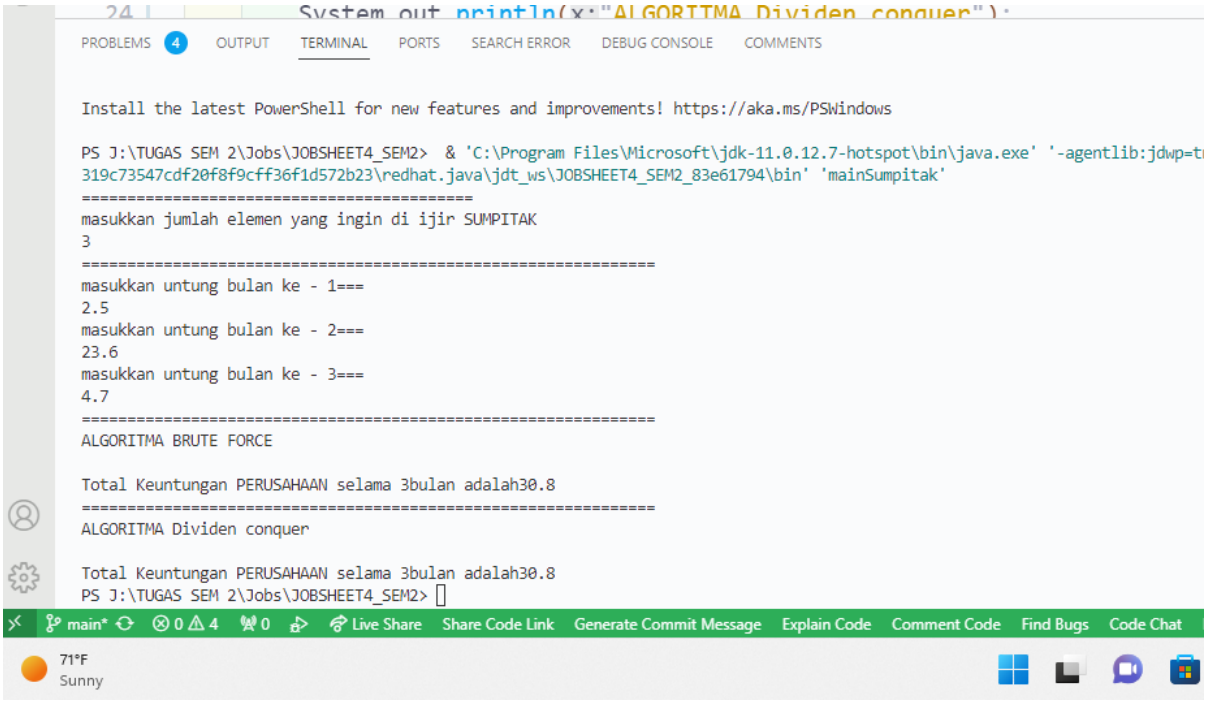
4.4.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.

```
import java.util.Scanner;

public class mainSumpitak {
    public static void main(String[] args) {
        Scanner pangkat19 = new Scanner(System.in);

        System.out.println("==========================
=================");
        System.out.println("masukkan jumlah elemen
yang ingin di ijir SUMPITAK");
        int elm = pangkat19.nextInt();

        sumpitak sumpitak = new sumpitak(elm);
        System.out.println("==========================
====================================");
        for (int i = 0; i < sumpitak.elemen; i++) {
            System.out.println("masukkan untung bulan
ke - "+(i+1)+"===");
            sumpitak.keuntungan[i] =
pangkat19.nextDouble();
        }

        System.out.println("==========================
====================================");
        System.out.println("ALGORITMA BRUTE FORCE");
        System.out.println("");
```

```
        System.out.println("Total Keuntungan
PERUSAHAAN selama "+ sumpitak.elemen + "bulan adalah"
+ sumpitak.totalBF(sumpitak.keuntungan));

        System.out.println("==========================
====================================");
        System.out.println("ALGORITMA Dividen
conquer");
        System.out.println("");
        System.out.println("Total Keuntungan
PERUSAHAAN selama "+ sumpitak.elemen + "bulan adalah"
+ sumpitak.totalDC(sumpitak.keuntungan, 0,
sumpitak.elemen-1));

    }
    //     Scanner pangkat19 = new Scanner(System.in);

    //     System.out.println("======================
====================");
    //     System.out.println("masukkan jumlah elemen
yang ingin di ijir SUMPITAK");
    //     int elm = pangkat19.nextInt();

    //     sumpitak sumpitak = new sumpitak(elm);
    //     System.out.println("======================
========================================");
    //     for (int i = 0; i < sumpitak.elemen; i++) {
    //         System.out.println("masukkan untung
bulan ke - "+(i+1)+"===");
    //         sumpitak.keuntungan[i] =
pangkat19.nextDouble();
    //     }
```

```
    //      System.out.println("=======================
=========================================");
    //      System.out.println("ALGORITMA BRUTE
FORCE");
    //      System.out.println("Total Keuntungan
PERUSAHAAN selama "+ sumpitak.elemen + "bulan adalah"
+ sumpitak.totalBF(sumpitak.keuntungan));
    //      System.out.println("ALGORITMA Dividen
conquer");
    //      System.out.println("Total Keuntungan
PERUSAHAAN selama "+ sumpitak.elemen + "bulan adalah"
+ sumpitak.totalDC(sumpitak.keuntungan, 0,
sumpitak.elemen-1));

    // }
}
```



```
run:
============================================================
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 = 40.34
BUILD SUCCESSFUL (total time: 11 seconds)
```

4.4.3 Pertanyaan

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

   Perbedaan TotalBF() dan TotalDC(): Metode `TotalBF()` menghitung total keuntungan dengan cara menjumlahkan semua elemen secara langsung, satu per satu. Ini seperti menambahkan setiap angka pada baris angka. Sebaliknya, metode `TotalDC()` menggunakan pendekatan Divide and Conquer. Ia membagi array keuntungan menjadi dua bagian yang sama, menghitung total masing-masing bagian secara rekursif, dan kemudian menambahkan dua total tersebut.

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.

   Untuk membatasi jumlah digit di belakang koma, Anda bisa menggunakan fungsi pembulatan atau format dalam bahasa pemrograman Anda. Misalnya, dalam Java, Anda bisa menggunakan `String.format("%.2f", yourValue)` untuk membatasi output hingga dua angka di belakang koma.

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

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

   Formulasi `return lsum+rsum+arr[mid];` digunakan dalam metode `TotalDC()`. `lsum` adalah total dari setengah bagian kiri array, `rsum` adalah total dari setengah bagian kanan

array, dan `arr[mid]` adalah elemen tengah array. Formulasi ini mengembalikan total dari seluruh array.

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

   Variabel `mid` digunakan untuk membagi array menjadi dua bagian yang hampir sama besar. Ini adalah bagian penting dari pendekatan Divide and Conquer.

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!

   Untuk menghitung keuntungan beberapa perusahaan sekaligus, Anda bisa membuat array dari objek `sumpitak`, di mana setiap objek mewakili satu perusahaan. Anda bisa mengulangi proses pengisian data dan perhitungan keuntungan untuk setiap perusahaan.

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.*

```
import java.util.Scanner;
public class mainsquar {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int x = scanner.nextInt();
        scanner.close();
        // int x = 10;
        System.out.println("Square root of " + x + "
using Brute Force: " + SquareRoot.sqrtBF(x));
        System.out.println("Square root of " + x + "
using Divide and Conquer: " + SquareRoot.sqrtDC(x));
    }
}
```

```
public class SquareRoot {
```

```
    public static int sqrtBF(int x) {
        if (x < 0) return -1; // Invalid input
        int i = 0;
        while (i * i <= x) {
            i++;
        }
        return i - 1; // Return the floor value of the
square root
    }

    // Divide and Conquer method
    public static int sqrtDC(int x) {
        if (x < 0) return -1; // Invalid input
        int start = 0, end = x;
        while (start <= end) {
            int mid = start + (end - start) / 2;
            if (mid * mid == x) {
                return mid;
            } else if (mid * mid < x) {
                start = mid + 1;
            } else {
                end = mid - 1;
            }
        }
        return end; // Return the floor value of the
square root
    }
}
```


int start = 0, end = x;
while (start <= end) {
int mid = start + (end - start) / 2;
PROBLEMS 1 OUTPUT TERMINAL PORTS SEARCH ERROR DEBUG CONSOLE COMMENTS
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
Square root of 10 using Brute Force: 3
Square root of 10 using Divide and Conquer: 3
Enter a number: 100
Square root of 100 using Brute Force: 10
Square root of 100 using Divide and Conquer: 10