LAPORAN HASIL PRAKTIKUM

ALGORITMA DAN STRUKTUR DATA

JOBSHEET 5

OLEH :

IVAN RIZAL AHMADI

NIM. 2341760128

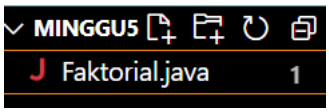
SIB-1F/13

D-IV SISTEM INFORMASI BISNIS

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

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

```
int nilai;
```

4. Tambahkan method faktorialBF

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

5. Tambahkan method faktorialDC

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

6. 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(x:"================================================");
System.out.print(s:"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 `faktorialDC()` dan `faktorialBF()`

```
System.out.println(x:"\nHasil Faktorial dengan Brute Force");
for (int i = 0; i < elemen; i++) {
    long factorial = fk[i].faktorialBF(fk[i].nilai);
    System.out.println("Faktorial dari nilai " + fk[i].nilai + " adalah : " + factorial);
}

System.out.println(x:"\nHasil Faktorial dengan Divide and Conquer");
for (int i = 0; i < elemen; i++) {
    long factorial = fk[i].faktorialDC(fk[i].nilai);
    System.out.println("Faktorial dari nilai " + fk[i].nilai + " adalah : " + factorial);
}
```

4.2.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda

(milik saya)

```
==================================================
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
==================================================
PS C:\Users\ivanr\OneDrive\Desktop\minggu5>
```

4.2.3 Pertanyaan

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

   Base line pada algoritma Divide and Conquer untuk mencari nilai faktorial adalah kondisi di mana masalah dipecah menjadi sub-masalah yang tidak dapat dipecah lagi. Dalam kasus ini, base line adalah ketika nilai n (input) sama dengan 1. Pada kondisi ini, nilai faktorial adalah 1.

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

- **Divide:** Membagi masalah menjadi dua sub-masalah yang lebih kecil. Sub-masalah pertama adalah menghitung faktorial n-1. Sub-masalah kedua adalah menghitung n.

  - **Conquer**: Memecahkan sub-masalah dengan menggunakan algoritma yang sama (rekursi). Menghitung nilai faktorial n-1 dan n.

  - **Combine**: Menggabungkan hasil dari dua sub-masalah untuk mendapatkan nilai faktorial n. Mengalikan nilai faktorial n-1 dengan n.

3. Apakah memungkinkan perulangan pada method `faktorialBF()` dirubah selain menggunakan for?Buktikan! = Meskipun kode yang diberikan menggunakan loop for, kita dapat memodifikasi faktorialBF untuk menggunakan loop while,

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

4. Tambahkan pegecekan waktu eksekusi kedua jenis method tersebut!

```
public long faktorialDC(int n) {
    long startTime = System.nanoTime();
    if (n == 1) {
        return 1;
    } else {
        long result = n * faktorialDC(n - 1);
        long endTime = System.nanoTime();
        long executionTime = endTime - startTime;
        System.out.println("Waktu Eksekusi Divide and Conquer: " + executionTime + " nanoseconds");
        return result;
    }
}
```

5. Buktikan dengan inputan elemen yang di atas 20 angka, apakah ada perbedaan waktu eksekusi? = ada perbedaan., Saya Menambahkan variabel startTime dan endTime untuk mencatat waktu awal dan akhir eksekusi setiap metode. Menghitung waktu eksekusi dengan rumus endTime - startTime. Mencetak waktu eksekusi setiap metode setelah menghitung faktorial.

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; /
public int pangkat;
```

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

```java
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()`

```java
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

   Selama ini belum ada kesalahan

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.

```java
Scanner sc = new Scanner(System.in);
System.out.println(x:"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.

```java
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

```
System.out.println(x:"Hasil Pangkat dengan Brute Force");
for (int i = 0; i < elemen; i++) {
    System.out.println("Nilai " + png[i].nilai + " pangkat " + png[i].pangkat + " adalah : " +
        png[i].pangkatBF(png[i].nilai, png[i].pangkat));
}
System.out.println(x:"=================================================================");
System.out.println(x:"Hasil Pangkat dengan Divide and Conquer");
for (int i = 0; i < elemen; i++) {
    System.out.println("Nilai " + png[i].nilai + " pangkat " + png[i].pangkat + " adalah : " +
        png[i].pangkatDC(png[i].nilai, png[i].pangkat));
}

System.out.println(x:"=================================================================");
```

4.3.2 Verifikasi Hasil Percobaan System.out.println('

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
=============================================================
```

```
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() (**Brute Force) melakukan perpangkatan dengan cara melakukan iterasi sebanyak n kali dan mengalikan nilai a pada setiap iterasi. Dan Ini adalah pendekatan langsung yang sederhana untuk menghitung pangkat.
   -**pangkatDC()** (Divide and Conquer) menggunakan pendekatan rekursif untuk memecah masalah perpangkatan menjadi masalah yang lebih kecil. Jika n adalah bilangan ganjil, maka nilai dikembalikan dengan memanggil dirinya sendiri dua kali dan mengalikan hasilnya dengan nilai a. Jika n adalah bilangan genap, maka nilai dikembalikan dengan memanggil dirinya sendiri dua kali dan mengalikan kedua hasilnya.

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 =

- jika nilai pangkat n adalah ganjil, maka method akan memanggil dirinya sendiri untuk menghitung n/2, lalu mengalikan hasilnya dengan hasilnya lagi, dan kemudian dikalikan dengan a.
- Jika nilai pangkat n adalah genap, method akan memanggil dirinya sendiri untuk menghitung n/2, lalu hasilnya akan dikalikan dengan hasilnya lagi.

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

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

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

- **Tahap** combine pada algoritma Divide and Conquer sebenarnya sudah termasuk dalam pemanggilan rekursif yang terjadi pada method pangkatDC(). Pada tahap ini, submasalah yang lebih kecil telah dipecah dan dihitung, kemudian hasilnya digabungkan kembali untuk mendapatkan hasil akhir.

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

- Sebelumnya**, atribut nilai** dan **pangkat diinisialisasi** secara terpisah setelah pembuatan objek. Setelah modifikasi, pengisian atribut dilakukan saat pembuatan objek menggunakan konstruktor dengan parameter.

  Potongan kode yang dimodifikasi:
  ```
  png[i] = new pangkat(nilai, pangkat);
  ```
- **Objek png** telah diubah menjadi atribut statis di kelas Mainpangkat agar dapat diakses dari metode statis lainnya di kelas yang sama. Ini memungkinkan untuk mengakses objek png dari metode main dan calculatePowers tanpa membuat hal baru.

  Potongan kode yang dimodifikasi:
  ```
  public static pangkat[] png;
  ```
- **Metode calculatePowers** diubah untuk menggunakan atribut statis png yang telah diperbarui. Ini memungkinkan untuk mengakses nilai dan pangkat dari setiap objek pangkat yang telah dibuat.

  Potongan kode yang dimodifikasi:
  ```
  public static void calculatePowers(int elemen) {
      System.out.println(x:"Hasil Pangkat dengan Brute Force");
  ```

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

```
============================================================
Masukkan jumlah elemen yang ingin dihitung :
3
Masukkan nilai yang akan dipangkatkan ke-1 : 2
Masukkan nilai pemangkat ke-1 : 3
Masukkan nilai yang akan dipangkatkan ke-2 : 4
Masukkan nilai pemangkat ke-2 : 2
Masukkan nilai yang akan dipangkatkan ke-3 : 3
Masukkan nilai pemangkat ke-3 : 4
============================================================
Pilih metode:
1. Pangkat Brute Force
2. Pangkat Divide and Conquer
Pilihan Anda: 2
Hasil Pangkat dengan Divide and Conquer
Nilai 2 pangkat 3 adalah : 8
Nilai 4 pangkat 2 adalah : 16
Nilai 3 pangkat 4 adalah : 81
PS C:\Users\ivanr\OneDrive\Desktop\minggu5>
```

Saya memodifikasi program tersebut yang Dimana memungkinkan pengguna untuk memilih metode perhitungan pangkat yang ingin digunakan (Brute Force atau Divide and Conquer). Setelah pengguna memasukkan nilai-nilai, mereka diminta untuk memilih metode. Kemudian, berdasarkan pilihan pengguna, program akan menjalankan metode yang sesuai dan menampilkan hasilnya. Hal ini memberikan fleksibilitas dan kontrol kepada pengguna dalam memilih metode perhitungan pangkat yang paling sesuai dengan kebutuhan mereka.

```
    System.out.println(x:"============================================================");

    System.out.println(x:"Pilih metode:");
    System.out.println(x:"1. Pangkat Brute Force");
    System.out.println(x:"2. Pangkat Divide and Conquer");
    System.out.print(s:"Pilihan Anda: ");
    int pilihan = sc.nextInt();

    switch (pilihan) {
        case 1:
            calculatePowersBF(png, elemen);
            break;
        case 2:
            calculatePowersDC(png, elemen);
            break;
        default:
            System.out.println(x:"Pilihan tidak valid.");
    }
}
```

Untuk membuat tambahan menu ini saya menggunakan logika switch dan juga if else, agar output yang dihasilkan bisa ada pemilihan menu nya.

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;

Sum13(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);
        double rsum = totalDC(arr, mid + 1, r);
        return lsum + rsum;
    }
    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(x:"========================================================================");
System.out.println(x:"Program Menghitung Keuntungan Total (Satuan Juta. Misal 5,9)");

System.out.print(s:"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.

```
Sum13 sm = new Sum13(elm);
System.out.println(x:"============================================================");
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)

```
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.keuntungan));
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.keuntungan, 1:0, sm.elemen - 1));
System.out.println(x:"============================================================");
```

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 Conquer
Total keuntungan perusahaan selama 5 bulan adalah = 40.339999999999996
============================================================
PS C:\Users\ivanr\OneDrive\Desktop\minggu5>
```

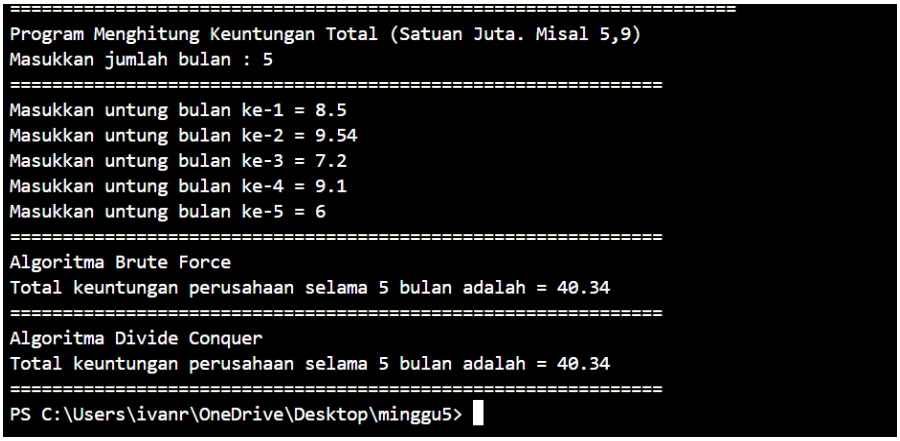
4.4.3 Pertanyaan

1. Berikan ilustrasi perbedaan perhitungan keuntungan dengan method `TotalBF()` ataupun
    - `TotalDC()` TotalBF() (Brute Force): Dalam metode ini, keuntungan total dihitung dengan cara menambahkan semua keuntungan bulanan secara berurutan. Misalnya, jika keuntungan bulanan adalah [5, 7, 3, 8], maka keuntungan total akan dihitung dengan cara: 5 + 7 + 3 + 8 = 23.

    - TotalDC() (Divide and Conquer): Dalam metode ini, keuntungan total dihitung dengan membagi masalah menjadi submasalah yang lebih kecil, menghitung keuntungan total dari setengah bagian pertama, kedua, dan seterusnya, kemudian menggabungkan hasilnya. Misalnya, jika keuntungan bulanan adalah [5, 7, 3, 8], maka keuntungan total akan dihitung dengan cara: (5 + 7) + (3 + 8) = 23.
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= menggunakan String.format() atau DecimalFormat untuk memformat output agar memiliki jumlah digit di belakang koma yang konsisten.

```
String.format(format:"%.2f", sm.totalBF(sm.keuntungan)));
```

```
String.format(format:"%.2f", sm.totalDC(sm.keuntungan, 1:0, sm.elemen - 1)));
```



3. Mengapa terdapat formulasi *return value* berikut?Jelaskan! = return value adalah jumlah total keuntungan yang dihitung dalam metode masing-masing. Ini penting agar nilai total keuntungan dapat digunakan atau dicetak di tempat lain dalam program.

4. Kenapa dibutuhkan variable `mid` pada method `TotalDC()`? = Variabel mid digunakan dalam metode TotalDC() karena metode Divide and Conquer membagi masalah menjadi submasalah yang lebih kecil dengan membagi array ke dalam dua bagian. Variabel mid menandai tengah dari array, yang digunakan untuk membagi array menjadi dua bagian yang lebih kecil.

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!

```java
import java.util.Scanner;

public class MainSum13 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.println(x:"================================================================");
        System.out.println(x:"Program Menghitung Keuntungan Total untuk Beberapa Perusahaan");
        System.out.println(x:"(Satuan Juta. Misal 5,9)");

        System.out.print(s:"Masukkan jumlah perusahaan: ");
        int numPerusahaan = sc.nextInt();

        System.out.print(s:"Masukkan jumlah bulan: ");
        int numBulan = sc.nextInt();


        double[][] profits = new double[numPerusahaan][numBulan];

        System.out.println(x:"============================================================");


        for (int i = 0; i < numPerusahaan; i++) {
            System.out.println("Untuk Perusahaan ke-" + (i + 1));
            for (int j = 0; j < numBulan; j++) {
                System.out.print("Masukkan untung bulan ke-" + (j + 1) + " = ");
                profits[i][j] = sc.nextDouble();
            }
            System.out.println(x:"============================================================");
        }


        for (int i = 0; i < numPerusahaan; i++) {
            double totalProfit = 0;
            for (int j = 0; j < numBulan; j++) {
                totalProfit += profits[i][j];
            }
            System.out.println("Total keuntungan perusahaan ke-" + (i + 1) + " selama " + numBulan + " bulan adalah: "
                    + totalProfit);
        }

        sc.close();
    }
}
```

- Program ini meminta pengguna untuk memasukkan jumlah perusahaan dan jumlah bulan.

- Kemudian, program membuat sebuah array dua dimensi bernama profits yang digunakan untuk menyimpan keuntungan untuk setiap perusahaan dan setiap bulan.

- Setelah itu, program menggunakan loop bersarang untuk meminta pengguna memasukkan keuntungan untuk setiap perusahaan dan setiap bulan.

- Setelah semua keuntungan dimasukkan, program menghitung total keuntungan untuk setiap perusahaan dengan menjumlahkan keuntungan untuk setiap bulan.

- Akhirnya, program mencetak total keuntungan untuk setiap perusahaan.

- Dengan demikian, program ini memungkinkan pengguna untuk menghitung total keuntungan untuk beberapa perusahaan selama beberapa bulan dengan cara yang sederhana dan langsung.

Lalu untuk hasil output nya adalah sebagai berikut =

```
================================================================
Program Menghitung Keuntungan Total untuk Beberapa Perusahaan
(Satuan Juta. Misal 5,9)
Masukkan jumlah perusahaan: 3
Masukkan jumlah bulan: 3
============================================================
Untuk Perusahaan ke-1
Masukkan untung bulan ke-1 = 222
Masukkan untung bulan ke-2 = 222
Masukkan untung bulan ke-3 = 222
============================================================
Untuk Perusahaan ke-2
Masukkan untung bulan ke-1 = 222
Masukkan untung bulan ke-2 = 222
Masukkan untung bulan ke-3 = 22
============================================================
Untuk Perusahaan ke-3
Masukkan untung bulan ke-1 = 222
Masukkan untung bulan ke-2 = 222
Masukkan untung bulan ke-3 = 222
============================================================
Total keuntungan perusahaan ke-1 selama 3 bulan adalah: 666.0
Total keuntungan perusahaan ke-2 selama 3 bulan adalah: 466.0
Total keuntungan perusahaan ke-3 selama 3 bulan adalah: 666.0
```

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 .*

```java
public class Akarbilangan13 {

    public static int bruteForceSqrt(int x) {
        if (x == 0 || x == 1) {
            return x;
        }
        int result = 1;
        while (result * result <= x) {
            result++;
        }
        return result - 1;
    }

    public static int divideConquerSqrt(int x) {
        if (x == 0 || x == 1) {
            return x;
        }
        return divideConquerBantu(x, kiri:0, x);
    }

    private static int divideConquerBantu(int x, int kiri, int kanan) {
        if (kiri <= kanan) {
            int tengah = kiri + (kanan - kiri) / 2;
            int tengahKuadrat = tengah * tengah;
            if (tengahKuadrat == x) {
                return tengah;
            } else if (tengahKuadrat < x) {
                return divideConquerBantu(x, tengah + 1, kanan);
            } else {
                return divideConquerBantu(x, kiri, tengah - 1);
            }
        }
        return kanan;
    }
}
```

```java
public class MainAkarbilangan13 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.println(x:"============================================================");
        System.out.println(x:"Program Menghitung Akar Bilangan (Bulatkan ke Bawah jika Bukan Kuadrat Sempurna)");
        System.out.println(x:"Masukkan bilangan yang ingin dihitung akarnya: ");
        int angka = sc.nextInt();

        int hasilBruteForce = Akarbilangan13.bruteForceSqrt(angka);

        int hasilDivideConquer = Akarbilangan13.divideConquerSqrt(angka);

        System.out.println("Bilangan: " + angka);
        System.out.println("Akar dengan Algoritma Brute Force: " + hasilBruteForce);
        System.out.println("Akar dengan Algoritma Divide Conquer: " + hasilDivideConquer);

        sc.close();
    }
}
```

Dalam implementasi ini, kelas AkarBilangan bertanggung jawab untuk menyediakan metode untuk menghitung akar bilangan dengan algoritma brute force dan divide conquer. Sedangkan kelas MainAkarBilangan bertanggung jawab untuk menjalankan program utama dan berinteraksi dengan pengguna untuk memasukkan bilangan yang ingin dihitung akarnya. Lalu untuk hasil nya adalah seperti di bawah ini.

```
Program Menghitung Akar Bilangan (Bulatkan ke Bawah jika Bukan Kuadrat Sempurna)
Masukkan bilangan yang ingin dihitung akarnya:
43
Bilangan: 43
Akar dengan Algoritma Brute Force: 6
Akar dengan Algoritma Divide Conquer: 6
PS C:\Users\ivanr\OneDrive\Desktop\minggu5>
```