LAPORAN PRAKTIKUM

ALGORITMA DAN STRUKTUR DATA

PERTEMUAN KE 5

BRUTE FORCE DAN DIVIDE CONQUER

Dosen Pengampuh : Bu. Triana Fatmawati, S.T., M.T.

Muhammad Afiq Firdaus

2341760189 / SIB-1E

PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

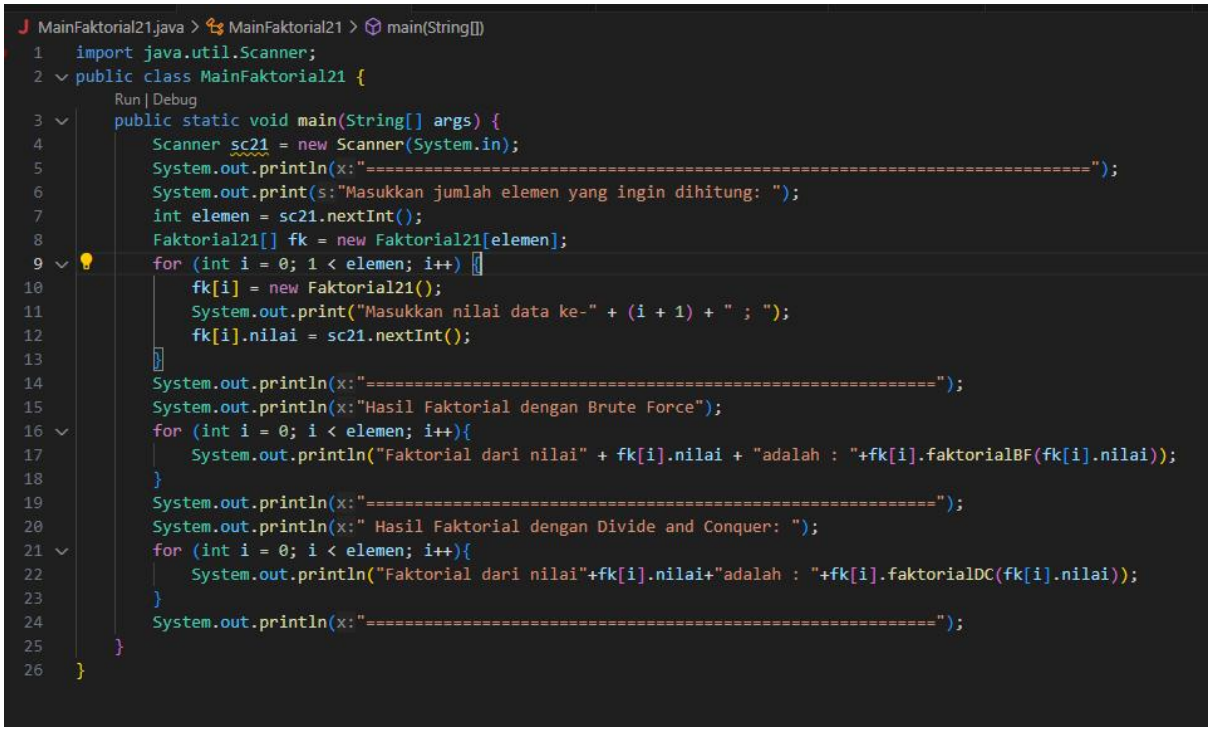
2024

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

Berikut adalah hasil coding Faktorial berdasarkan jobsheet.

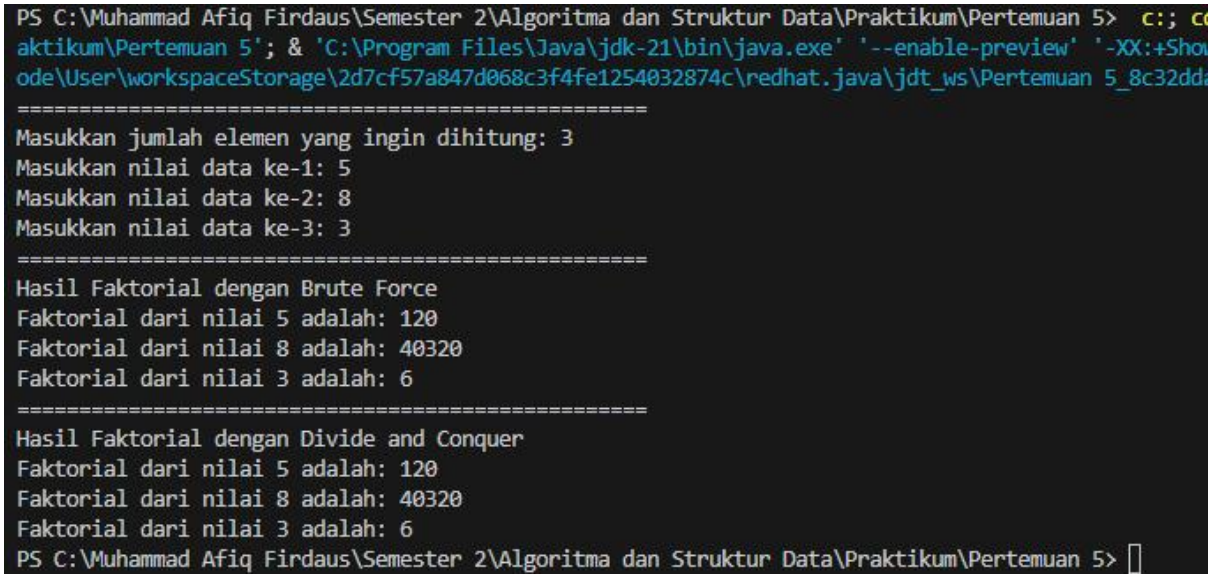
```java
public class Faktorial21 {
    public int nilai;

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

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

Berikut adalah code MainFaktorial berdasarkan pada jobsheet



```java
import java.util.Scanner;
public class MainFaktorial21 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc21 = new Scanner(System.in);
        System.out.println("============================================================================");
        System.out.print("Masukkan jumlah elemen yang ingin dihitung: ");
        int elemen = sc21.nextInt();
        Faktorial21[] fk = new Faktorial21[elemen];
        for (int i = 0; 1 < elemen; i++) {
            fk[i] = new Faktorial21();
            System.out.print("Masukkan nilai data ke-" + (i + 1) + " ; ");
            fk[i].nilai = sc21.nextInt();
        }
        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("============================================================");
    }
}
```

Berikut adalah hasil run dari code diatas :



Pertanyaan
1. Jelaskan mengenai base line Algoritma Divide Conquer untuk melakukan pencarian nilai faktorial!
2. Pada implementasi Algoritma Divide and Conquer Faktorial apakah lengkap terdiri dari 3 tahapan divide, conquer, combine? Jelaskan masing-masing bagiannya pada kode program!
3. Apakah memungkinkan perulangan pada method faktorialBF() dirubah selain menggunakan for?Buktikan!
4. Tambahkan pegecekan waktu eksekusi kedua jenis method tersebut!
5. Buktikan dengan inputan elemen yang di atas 20 angka, apakah ada perbedaan waktu eksekusi?

Jawaban :

1. Algoritma Divide-and-Conquer untuk mencari nilai faktor membagi masalah

menjadi submasalah yang lebih kecil, menyelesaikan setiap submasalah secara terpisah, dan menggabungkan solusi untuk mendapatkan solusi akhir.
Sebagai bagian dari pencarian faktorial, algoritma pertama-tama memeriksa apakah nilai yang diinginkan adalah 1. Jika demikian, ia segera mengembalikan nilai 1. Jika tidak, algoritma membagi permasalahan menjadi permasalahan yang lebih kecil dengan mengalikan nilai yangg ditemukan dengan faktorial dari nilai sebelumnya. Proses ini diulangi hingga kasus dasar tercapai, dimana nilai yang diinginkan adalah 1.
Hasil dari setiap langkah rekursi kemudian digabungkan untuk mendapatkan nilai faktor akhir. Oleh karena itu, algoritma ini menggunakan pendekatan rekursif untuk menyelesaikan masalah, menggabungkan solusi hingga sampai pada solusi akhir.

2. Pada implementasi Algoritma Divide and Conquer Faktorial apakah lengkap

terdiri dari 3 tahapan divide, conquer, combine? Jelaskan masing-masing bagiannya pada kode program!
Pada percobaan 1 hanya terdapat 2 tahapan yaitu Divide dan Conquer karena tahapan Combine tidak selalu eksplisit dalam implementasi faktorial menggunakan algoritma Divide and Conquer.

- Devide

Terjadi ketika pada fungsi faktorialDC(int n), kita membagi masalah menjadi masalah yang lebih kecil dengan mengurangi nilai faktorial yang dicari (n) satu per satu hingga mencapai kasus dasar (n==1).

- Conquer

Terjadi di dalam blok else pada fungsi faktorialDC(int n). Pada tahap ini, kita menyelesaikan setiap submasalah secara rekursif dengan mengalikan nilai faktorial yang sedang dicari (n) dengan nilai faktorial dari masalah yang lebih kecil (faktorialDC(n-1)). Setiap langkah rekursif ini akan terus berulang sampai mencapai kasus dasar.

3. Apakah memungkinkan perulangan pada method faktorialBF() dirubah selain

menggunakan for?Buktikan!

Perulangan pada method faktorialBF() dapat diubah menjadi rekursif dengan menggunakan if else seperti berikut :

```
    // Setelah pertanyaan
    if(n==0 || n==1){
        return 1;
    }else{
        return n * faktorialBF(n-1);
    }
    }
```

4. Tambahkan pegecekan waktu eksekusi kedua jenis method tersebut!

```
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 5>  c:; c
aktikum\Pertemuan 5'; & 'C:\Program Files\Java\jdk-21\bin\java.exe' '--enable-preview' '-XX:+Sho
ode\User\workspaceStorage\2d7cf57a847d068c3f4fe1254032874c\redhat.java\jdt_ws\Pertemuan 5_8c32dd
================================================
Masukkan jumlah elemen yang ingin dihitung: 3
Masukkan nilai data ke-1: 8
Masukkan nilai data ke-2: 5
Masukkan nilai data ke-3: 3
================================================
Hasil Faktorial dengan Brute Force
Faktorial dari nilai 8 adalah: 40320 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 5 adalah: 120 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 3 adalah: 6 (Waktu eksekusi: 0 ms)
================================================
Hasil Faktorial dengan Divide and Conquer
Faktorial dari nilai 8 adalah: 40320 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 5 adalah: 120 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 3 adalah: 6 (Waktu eksekusi: 0 ms)
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 5>
```

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

```
================================================
Hasil Faktorial dengan Brute Force
Faktorial dari nilai 8 adalah: 40320 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 9 adalah: 362880 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 12 adalah: 479001600 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 3 adalah: 6 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 4 adalah: 24 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 7 adalah: 5040 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 10 adalah: 3628800 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 13 adalah: 1932053504 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 15 adalah: 2004310016 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 16 adalah: 2004189184 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 17 adalah: -288522240 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 19 adalah: 109641728 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 20 adalah: -2102132736 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 22 adalah: -522715136 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 1 adalah: 1 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 3 adalah: 6 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 4 adalah: 24 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 5 adalah: 120 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 25 adalah: 2076180480 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 29 adalah: -1241513984 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 23 adalah: 862453760 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 30 adalah: 1409286144 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 18 adalah: -898433024 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 16 adalah: 2004189184 (Waktu eksekusi: 0 ms)
```

```
================================================
Hasil Faktorial dengan Divide and Conquer
Faktorial dari nilai 8 adalah: 40320 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 9 adalah: 362880 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 12 adalah: 479001600 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 3 adalah: 6 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 4 adalah: 24 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 7 adalah: 5040 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 10 adalah: 3628800 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 13 adalah: 1932053504 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 15 adalah: 2004310016 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 16 adalah: 2004189184 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 17 adalah: -288522240 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 19 adalah: 109641728 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 20 adalah: -2102132736 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 22 adalah: -522715136 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 1 adalah: 1 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 3 adalah: 6 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 4 adalah: 24 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 5 adalah: 120 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 25 adalah: 2076180480 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 29 adalah: -1241513984 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 23 adalah: 862453760 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 30 adalah: 1409286144 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 18 adalah: -898433024 (Waktu eksekusi: 0 ms)
Faktorial dari nilai 16 adalah: 2004189184 (Waktu eksekusi: 0 ms)
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktil
```

Percobaan 2

Berikut adalah code Pangkat21 berdasarkan pada jobsheet

```java
public class Pangkat21 {
    public int nilai, pangkat;

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

    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));
        }
    }
}
```

Berikut adalah MainPangkat berdasarkan pada Jobsheet

```java
import java.util.Scanner;
public class MainPangkat21 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("==============================");
        System.out.println("Masukkan jumlah elemen yang ingin dihitung: ");
        int elemen = sc.nextInt();

        Pangkat21[] png = new Pangkat21[elemen];

        for (int i = 0; i < elemen; i++) {
            png[i] = new Pangkat21();
            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();
        }
        System.out.println("==============================");
        System.out.println("Hasil pangkat dengan Brute Force");
        for (int i = 0; i < elemen; i++) {
            System.out.println("Nilai " + p  int i - MainPangkat21.main(String[])  gkat + "adalah : "
                    + png[i].pangkatBF(png[i].nilai, png[i].pangkat));
        }
        System.out.println("==============================");
        System.out.println("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("==============================");
    }
}
```

Verifikasi hasil percobaan

```
PS C:\Muham>  c:; cd 'c:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Pr
-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\C
\Pertemuan 5_8c32dda\bin' 'MainPangkat21'
================================
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 2adalah : 36
Nilai 4 pangkat 3adalah : 64
================================
Hasil Pangkat dengan Divide and Conquer
Nilai 6pangkat2adalah : 36
Nilai 4pangkat3adalah : 64
================================
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 5
```

Pertanyaan

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

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

4. Modifikasi kode program tersebut, anggap proses pengisian atribut dilakukan dengan konstruktor.
5. Tambahkan menu agar salah satu method yang terpilih saja yang akan dijalankan!

Jawaban :

1.

- PangkatBF()

a. Menggunakan iterasi sederhana dengan looping for untuk mengalikan bilangan a sebanyak n kali
b. Algoritma ini efektif untuk nilai n yang relatif kecil, tetapi memiliki kompleksitas waktu O(n) karena membutuhkan waktu linear sesuai dengan nilai eksponen n

- PangkatDC()

a. Membagi masalah menjadi submasalah yang lebih kecil dan menyelesaikannya secara rekursif.

b. Algoritma ini memanfaatkan sifat matematis bahwa a^n dapat dipecah menjadi a^(n/2) * a^(n/2) untuk eksponen n genap dan (a^(n/2) * a^(n/2) * a) untuk eksponen n ganjil.
c. Algoritma ini memiliki kompleksitas waktu yang lebih baik daripada Brute Force, yaitu O(log n), karena melakukan pembagian masalah secara rekursif menjadi submasalah yang lebih kecil

2.

- if (n % 2 == 1) { // Bilangan Ganjil

Pernyataan ini memeriksa apakah eksponen n adalah bilangan ganjil. Jika sisa pembagian n dengan 2 sama dengan 1, itu berarti n adalah bilangan ganjil.

- return (pangkatDC(a, n/2) * pangkatDC(a, n/2) * a)

Jika eksponen n adalah bilangan ganjil, maka masalah dibagi menjadi dua bagian: a^(n/2) dan a^(n/2). Kedua submasalah ini diselesaikan secara rekursif menggunakan metode pangkatDC(). Kemudian, hasilnya dikalikan dengan a, karena eksponen n ganjil menambahkan satu faktor a tambahan.

- else { // Bilangan Genap

Ini adalah bagian dari struktur if-else yang menangani kasus ketika eksponen n adalah bilangan genap

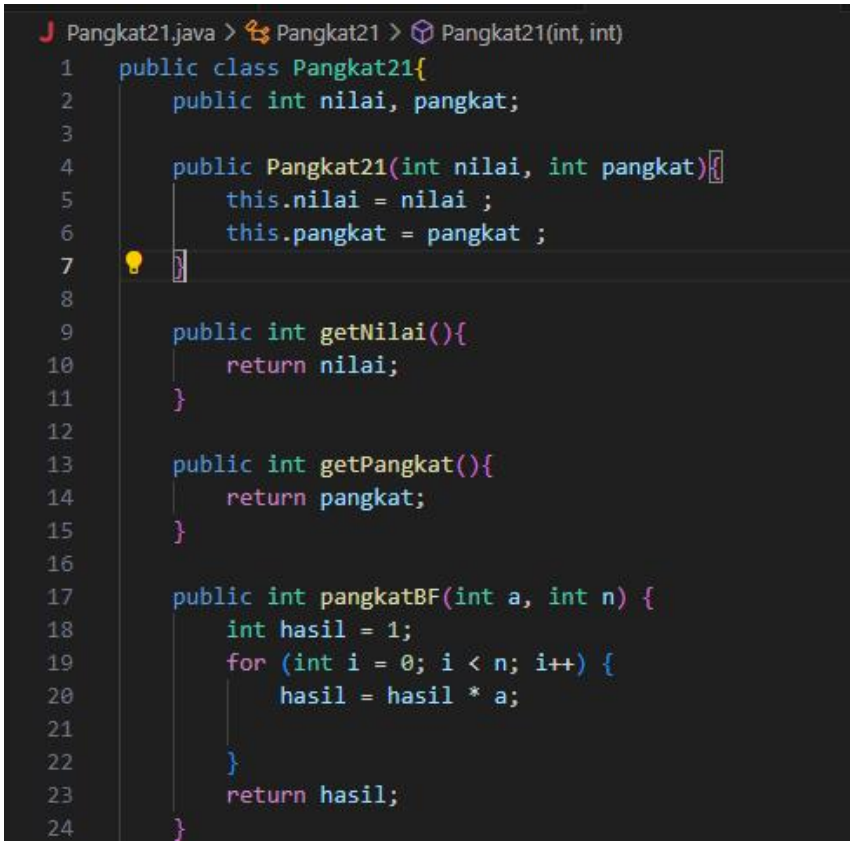
- return (pangkatDC(a, n/2) * pangkatDC(a, n/2));

Jika eksponen n adalah bilangan genap, maka masalah dibagi menjadi dua bagian yang sama: a^(n/2) dan a^(n/2). Kedua submasalah ini diselesaikan secara rekursif menggunakan metode pangkatDC(), dan hasilnya dikalikan bersama. Dengan cara ini, pendekatan Divide and Conquer secara rekursif memecah masalah menjadi submasalah yang lebih kecil dan menyelesaikannya secara efisien.

3.

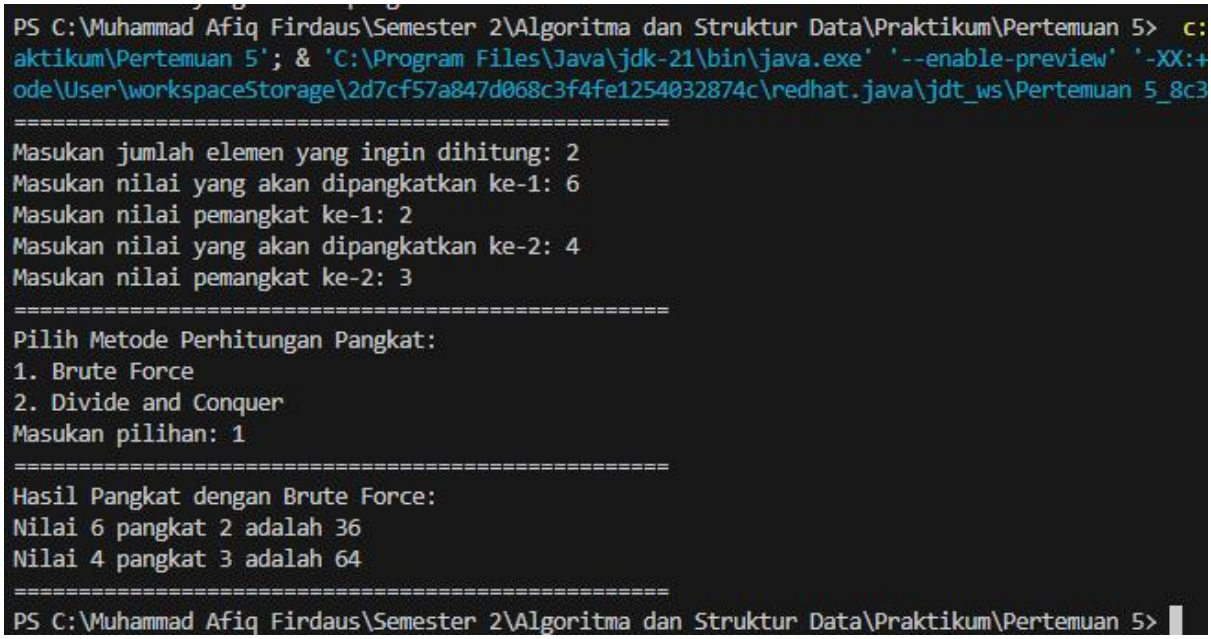
```
if(n%2==1)//bilangan ganjil
    return (pangkatDC(a,n/2)*pangkatDC(a,n/2)*a);
else//bilangan genap
    return (pangkatDC(a,n/2)*pangkatDC(a,n/2));
```

4.



```java
Pangkat21.java > Pangkat21 > Pangkat21(int, int)
public class Pangkat21{
    public int nilai, pangkat;

    public Pangkat21(int nilai, int pangkat){
        this.nilai = nilai ;
        this.pangkat = pangkat ;
    }

    public int getNilai(){
        return nilai;
    }

    public int getPangkat(){
        return pangkat;
    }

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

        }
        return hasil;
    }
```

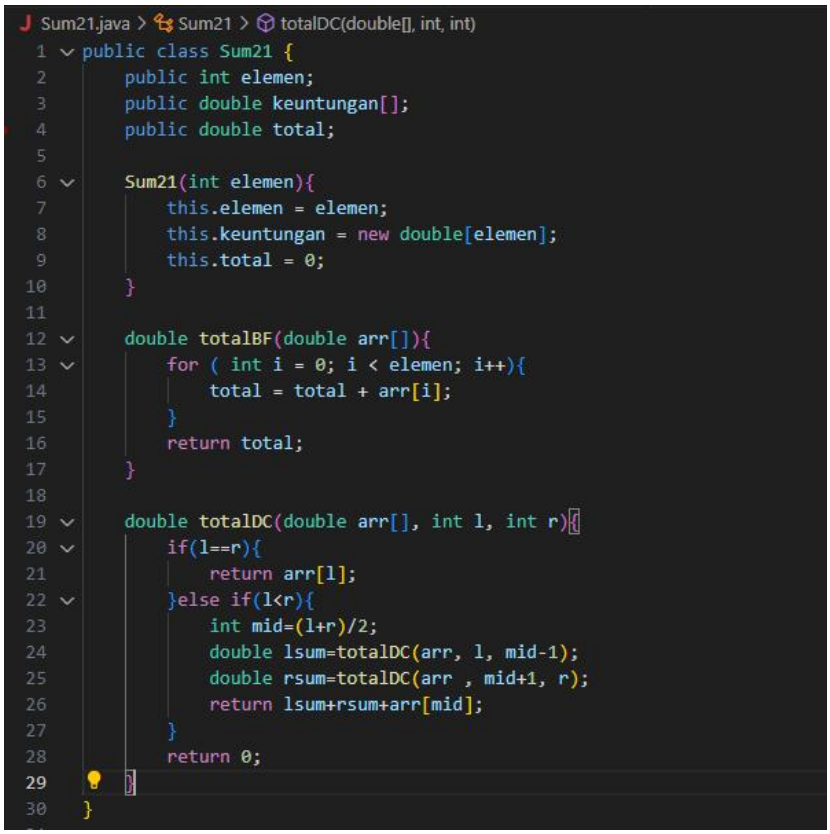
5.



```
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 5>  c:
aktikum\Pertemuan 5'; & 'C:\Program Files\Java\jdk-21\bin\java.exe' '--enable-preview' '-XX:+
ode\User\workspaceStorage\2d7cf57a847d068c3f4fe1254032874c\redhat.java\jdt_ws\Pertemuan 5_8c3
================================================
Masukan jumlah elemen yang ingin dihitung: 2
Masukan nilai yang akan dipangkatkan ke-1: 6
Masukan nilai pemangkat ke-1: 2
Masukan nilai yang akan dipangkatkan ke-2: 4
Masukan nilai pemangkat ke-2: 3
================================================
Pilih Metode Perhitungan Pangkat:
1. Brute Force
2. Divide and Conquer
Masukan pilihan: 1
================================================
Hasil Pangkat dengan Brute Force:
Nilai 6 pangkat 2 adalah 36
Nilai 4 pangkat 3 adalah 64
================================================
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 5>
```

Percobaan 3

Berikut adalah code Sum berdasarkan pada Jobsheet



```java
Sum21.java > Sum21 > totalDC(double[], int, int)
public class Sum21 {
    public int elemen;
    public double keuntungan[];
    public double total;

    Sum21(int elemen){
        this.elemen = elemen;
        this.keuntungan = new double[elemen];
        this.total = 0;
    }

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

Berikut adalah code MainSum berdasarkan pada Jobsheet

```java
MainSum21.java > ...
import java.util.Scanner;

public class MainSum21 {
    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 (Satuan Juta. Misal 5.9)");
        System.out.print(s:"Masukkan jumlah bulan : ");
        int elm = sc.nextInt();

        Sum21 sm = new Sum21(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();
        }

        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, l:0, sm.elemen - 1));
    }
}
```

Berikut adalah verifikasi hasil percobaan

```
PS C:\Muhammad Afiq F>  c:; cd 'c:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Pr
 '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\C
ava\jdt_ws\Pertemuan 5_8c32dda\bin' 'MainSum21'
============================
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
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 5>
```

Pertanyaan

1. Berikan ilustrasi perbedaan perhitungan keuntungan dengan method TotalBF() ataupun TotalDC()
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.
3. Mengapa terdapat formulasi *return value* berikut?Jelaskan!

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

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

Jawaban

1.

- totalBF()

a. Pada metode ini akan menelusuri seluruh elemen dalam array keuntungan
b. Setiap elemen ditambahkan ke total secara berurutan dengan kompleksitas waktu O(n), di mana n adalah jumlah elemen dalam array

- totalDC()

a. Metode ini membagi masalah menjadi submasalah yang lebih kecil, menyelesaikan submasalah tersebut secara rekursif, dan menggabungkan hasilnya
b. Ketika mencapai kasus dasar (basis), yaitu ketika kisaran (range) dari l dan r hanya satu elemen, maka nilai dari elemen tersebut dikembalikan
c. Jika kisaran lebih dari satu elemen, kisaran dibagi menjadi dua bagian pada

tengahnya (mid), dan dua submasalah dipecahkan secara rekursif untuk setiap bagian kiri dan kanan
d. Kemudian, total dari kiri, kanan, dan elemen tengah dihitung dan digabungkan untuk memberikan total keseluruhan
e. Proses ini memiliki kompleksitas waktu O(n log n) karena membagi masalah menjadi dua setengah pada setiap tingkat rekursi.
Jadi, TotalBF() sederhana dan langsung menambahkan keuntungan secara berurutan, sementara TotalDC() membagi masalah menjadi submasalah lebih kecil, menyelesaikannya secara rekursif, dan kemudian menggabungkan hasilnya untuk mendapatkan total keseluruhan

2.

```java
import java.util.Scanner;

public class MainSum21 {
    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 (Satuan Juta. Misal 5.9)");
        System.out.print(s:"Masukkan jumlah bulan : ");
        int elm = sc.nextInt();

        Sum21 sm = new Sum21(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();
        }

        System.out.println(x:"========================================");
        System.out.println(x:"Algoritma Brute Force");
        double totalBF = sm.totalBF(sm.keuntungan);
        System.out.println("Total keuntungan perusahaan selama " + sm.elemen + " bulan adalah = " + String.format(format:"%.2f", totalBF));
        System.out.println(x:"========================================");
        System.out.println(x:"Algoritma Divide Conquer");
        double totalDC = sm.totalDC(sm.keuntungan, l:0, sm.elemen - 1);
        System.out.println("Total keuntungan perusahaan selama " + sm.elemen + " bulan adalah = " + String.format(format:"%.2f", totalDC));
    }
}
// Modifikasi
```


3.

- lsum

Ini adalah hasil dari pemanggilan rekursif totalDC() pada setengah bagian kiri array (dari indeks l hingga mid - 1). Metode ini mengembalikan total keuntungan dari setengah bagian kiri

- rsum

Ini adalah hasil dari pemanggilan rekursif totalDC() pada setengah bagian kanan array (dari indeks mid + 1 hingga r). Metode ini mengembalikan total keuntungan dari setengah bagian kanan

- arr[mid]

Ini adalah nilai keuntungan pada indeks mid, yang merupakan nilai keuntungan tengah dari array.
Jadi, return lsum + rsum + arr[mid]; menggabungkan total keuntungan dari setengah bagian kiri, setengah bagian kanan, dan nilai keuntungan tengah, yang merupakan total keseluruhan keuntungan dari seluruh array. Ini adalah langkah combine dalam algoritma Divide and Conquer, di mana hasil dari dua submasalah yang lebih kecil digabungkan bersama-sama untuk membentuk solusi untuk masalah yang lebih besar.

4. Variabel mid dibutuhkan dalam metode totalDC() karena itu merupakan

indeks yang menandai titik tengah dari array yang sedang diproses. Dalam pendekatan Divide and Conquer, array dibagi menjadi dua bagian setiap kali metode rekursif dipanggil. Variabel mid digunakan untuk menandai pembagian tersebut

5.

```
aktikum\Pertemuan 5'; & 'C:\Program Files\Java\jdk-21\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDe
ode\User\workspaceStorage\2d7cf57a847d068c3f4fe1254032874c\redhat.java\jdt_ws\Pertemuan 5_8c32dda\bin'
============================
Program Menghitung Keuntungan Total (Satuan Juta. Misal 5.9)
Masukkan jumlah perusahaan : 2
Masukkan jumlah bulan untuk perusahaan ke-1 : 3
Masukkan jumlah bulan untuk perusahaan ke-2 : 7
============================
Masukkan untung bulan ke - 1 untuk perusahaan ke-1 : 8,5
Masukkan untung bulan ke - 2 untuk perusahaan ke-1 : 7,4
Masukkan untung bulan ke - 3 untuk perusahaan ke-1 : 6,5
Masukkan untung bulan ke - 1 untuk perusahaan ke-2 : 9,3
Masukkan untung bulan ke - 2 untuk perusahaan ke-2 : 8,4
Masukkan untung bulan ke - 3 untuk perusahaan ke-2 : 6,8
Masukkan untung bulan ke - 4 untuk perusahaan ke-2 : 6,9
Masukkan untung bulan ke - 5 untuk perusahaan ke-2 : 6,1
Masukkan untung bulan ke - 6 untuk perusahaan ke-2 : 4,5
Masukkan untung bulan ke - 7 untuk perusahaan ke-2 : 6,2
========================================
Perusahaan ke-1
Algoritma Brute Force
Total keuntungan perusahaan selama 3 bulan adalah = 22,40
========================================
Algoritma Divide Conquer
Total keuntungan perusahaan selama 3 bulan adalah = 22,40
========================================
Perusahaan ke-2
Algoritma Brute Force
Total keuntungan perusahaan selama 7 bulan adalah = 48,20
========================================
Algoritma Divide Conquer
Total keuntungan perusahaan selama 7 bulan adalah = 48,20
========================================
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 5>
```

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.*

MainLatihan21.java > MainLatihan21

```java
import java.util.Scanner;

public class MainLatihan21 {
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println(x:"================================");
        System.out.println(x:"        Perhitungan Akar");
        System.out.println(x:"================================");
        System.out.print(s:"Masukkan bilangan: ");
        double number = scanner.nextDouble();
        Latihan21 latihan = new Latihan21();

        // Brute Force
        double bruteResult = latihan.bruteForceSqrt(number);
        System.out.println(x:"================================");
        System.out
                .println("Nilai akar dari  " + number + " adalah: " + bruteResult + " Dengan menggunakan Brute Force");
        System.out.println(x:"================================");

        // Divide Conquer
        double divideConquerResult = latihan.divideConquerSqrt(number);
        System.out.println(
                "Nilai akar dari " + number + " adalah: " + divideConquerResult + " Dengan menggunakan Divide Conquer");
        System.out.println(x:"================================");

        scanner.close();
    }
}
```

Latihan21.java > Latihan21

```java
public class Latihan21 {
    // Brute Force
    public double bruteForceSqrt(double x) {
        if (x == 0 || x == 1) {
            return x;
        }

        double result = 1;
        while (result * result <= x) {
            result++;
        }
        return (int) result - 1; // Bulatkan ke bawah
    }

    // Divide Conquer
    public double divideConquerSqrt(double x) {
        if (x == 0 || x == 1) {
            return x;
        }
        return divideConquerHelper(x, start:0, x);
    }

    public double divideConquerHelper(double x, double start, double end) {
        double mid = start + (end - start) / 2;
        double midSquare = mid * mid;

        if (midSquare == x || (midSquare < x && (mid + 0.001) * (mid + 0.001) > x)) {
            return (int) mid; // Bulatkan ke bawah
        } else if (midSquare < x) {
            return divideConquerHelper(x, mid, end);
        } else {
            return divideConquerHelper(x, start, mid);
        }
    }
```

```
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 5>  c:; cd 'c:\M
aktikum\Pertemuan 5'; & 'C:\Program Files\Java\jdk-21\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDet
ode\User\workspaceStorage\2d7cf57a847d068c3f4fe1254032874c\redhat.java\jdt_ws\Pertemuan 5_8c32dda\bin'
================================
        Perhitungan Akar
================================
Masukkan bilangan: 36
================================
Nilai akar dari  36.0 adalah: 6.0 Dengan menggunakan Brute Force
================================
Nilai akar dari 36.0 adalah: 5.0 Dengan menggunakan Divide Conquer
================================
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 5>
```