NAMA : AFRIZAL RAFLI KUSUMA WARDANA
NIM : 244107020007
NO ABSEN : 01
KELAS : TI-1F
MATERI : Brute Force dan Devide Conquer

LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

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

Class : faktorial

```java
public class faktorial {
    int faktorialBF(int n){
        int fakto = 1;
        int i = 1;
        while (i <= n) {
            fakto = fakto * i;
            i++;
        }
        return fakto;
    }
        int faktorialDC(int n){
        if (n == 1){
            return 1;
        } else {
            int fakto = n * faktorialDC(n-1);
            return fakto;
        }
    }
}
```

Class : Mainfaktorial

```java
import java.util.Scanner;
public class Mainfaktorial {
    public static void main(String[] args) {
    Scanner input = new Scanner(System.in);
    System.out.print(s:"Masukkan nilai: ");
    int nilai = input.nextInt();
    faktorial fk = new faktorial();
    System.out.println("Nilai faktorial " + nilai + "menggunakan BF: " + fk.faktorialBF(nilai));
    System.out.println("Nilai faktorial " + nilai + "menggunakan DF: " + fk.faktorialDC(nilai));
    }
}
```

Hasil:

```
Data\Roaming\Code\User\workspaceStorage\
Masukkan nilai: 5
Nilai faktorial 5menggunakan BF: 120
Nilai faktorial 5menggunakan DF: 120
PS D:\PROGDAS\2024-2025\Prk.ASD>
```

Question :

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

NAMA : AFRIZAL RAFLI KUSUMA WARDANA
NIM : 244107020007
NO ABSEN : 01
KELAS : TI-1F
MATERI : Brute Force dan Devide Conquer

Penggunaan if digunakan untuk mengecek apakah n sudah mencapai base case, jika kondisi terpenuhi (n = 1) maka akan langsung mengembalikan nilai 1 sebagai hasil faktorial 1.

Penggunaan else digunakan untuk untuk menangani kasus rekursif, yang mana jika kondisi if tidak terpenuhi atau n masih lebih besar dibanding 1 maka fungsi akan mengalikan n dengan nilai faktorial dari n-1 (rekursif), yang berlanjut hingga n menjadi 1 (base case terpenuhi.

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

Answer:

Kita juga bisa mengganti perulangan for menggunakan while loop

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

Hasil akan tetap sama setelah mengganti perulangan for dengan while:

```
Masukkan nilai: 3
Nilai faktorial 3menggunakan BF: 6
Nilai faktorial 3menggunakan DF: 6
```

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

Answer:

*fakto *= i;* adalah operasi yang mengalikan nilai yang tersimpan dalam fakto dengan nilai i dan kemudian menyimpan hasilnya kembali ke dalam fakto.

*int fakto = n * faktorialDC(n-1);* adalah deklarasi variabel fakto yang kemudian diinisialisasi dengan hasil perkalian antara n dan hasil dari rekursi faktorialDC(n-1). Yang mana menggunakan hasil rekursi faktorialDC(n-1) untuk menghitung faktorial dari n

4. Buat Kesimpulan tentang perbedaan cara kerja method faktorialBF() dan faktorialDC()!

Answer:

Perbedaan utama antara metode faktorialBF() dan faktorialDC() terletak pada cara mereka menyelesaikan perhitungan faktorial. Metode faktorialBF() menggunakan pendekatan iteratif dengan perulangan while, di mana nilai faktorial dihitung secara bertahap dengan mengalikan angka dari 1 hingga n. Pendekatan ini lebih efisien dalam penggunaan memori karena tidak memerlukan pemanggilan fungsi berulang.

Sementara itu, metode faktorialDC() menggunakan pendekatan rekursif berbasis Divide and Conquer, di mana setiap pemanggilan fungsi akan

NAMA : AFRIZAL RAFLI KUSUMA WARDANA
NIM : 244107020007
NO ABSEN : 01
KELAS : TI-1F
MATERI : Brute Force dan Devide Conquer

memecah masalah menjadi submasalah yang lebih kecil hingga mencapai kasus dasar n == 1.

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

Class : mainpangkat1

```java
import java.util.Scanner;
Codeium: Refactor | Explain
public class mainpangkat1 {
    Run | Debug | Run main | Debug main | Codeium: Refactor | Explain | Generate Javadoc | ×
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print(s:"Masukkan jumlahh elemen: ");
        int elemen = input.nextInt();
        pangkat1 [] png = new pangkat1[elemen];
        for (int i = 0; i < elemen; i++) {
            System.out.print("Masukkan nilai basis elemen ke-" + (i+1) + ": ");
            int basis = input.nextInt();
            System.out.print("Masukkan nilai pangkat elemen ke- " + (i+1) + ": ");
            int pangkat = input.nextInt();
            png[i] =  new pangkat1(basis,pangkat);
        }
        System.out.println(x:"HASIL PANGKAT BRUTE FORCE: ");
        for (pangkat1 p : png) {
            System.out.println(p.nilai +"^"+p.pangkat+" : "+p.pangkatBF(p.nilai,p.pangkat));
        }
        System.out.println(x:"HASIL PANGKAT DIVIDE CONQUER: ");
        for (pangkat1 p : png) {
            System.out.println(p.nilai +"^"+p.pangkat+" : "+p.pangkatDC(p.nilai,p.pangkat));
        }
    }
}
```

Class : pangkat1

```java
public class pangkat1 {
    int nilai, pangkat;
    pangkat1(int n, int p) {
        nilai = n;
        pangkat = p;
    }
    Codeium: Refactor | Explain | Generate Javadoc | ×
    int pangkatBF(int a, int n){
        int hasil = 1;
        for (int i = 0; i < n; i++) {
            hasil = hasil * a;
        }
        return hasil;
    }
    Codeium: Refactor | Explain | Generate Javadoc | ×
    int pangkatDC(int a, int n){
        if (n==1){
            return a;
        } 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));
            }
        }
    }
}
```

Hasil:

NAMA	: AFRIZAL RAFLI KUSUMA WARDANA
NIM	: 244107020007
NO ABSEN	: 01
KELAS	: TI-1F
MATERI	: Brute Force dan Devide Conquer

```
PS C:\Users\dedyb\OneDrive\Documents\KULIAH SMST 2\Praktikum Algoritma dan Struktur Data\PrakASD_1F_07>  c:; cd 'c:\Users\dedyb\OneDrive\Documents\KULIAH SMST 2\Praktikum Algoritma dan Struktur Data\PrakASD_1F_07'; & 'C:\Program Files\Java\jdk-21\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dedyb\OneDrive\Documents\KULIAH SMST 2\Praktikum Algoritma dan Struktur Data\PrakASD_1F_07\bin' 'P5.Pangkat.MainPangkat07'
========================================
Masukan jumlah elemen yang dihitung: 2
Masukan nilai yang ingin dipangkatkan: 6
Masukan nilai pemangkat             : 2
Masukan nilai yang ingin dipangkatkan: 4
Masukan nilai pemangkat             : 3
HASIL PANGKAT - BRUTE FORCE
Hasil dari 6 pangkat 2 adalah : 36
Hasil dari 4 pangkat 3 adalah : 64
HASIL PANGKAT - DIVIDE CONQUER
Hasil dari 6 pangkat 2 adalah : 36
Hasil dari 4 pangkat 3 adalah : 64
PS C:\Users\dedyb\OneDrive\Documents\KULIAH SMST 2\Praktikum Algoritma dan Struktur Data\PrakASD_1F_07>
```

Question :

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

   Answer:

   PangkatBF adalah metode brute force, yang mana bilangan a dikalikan dengan dirinya sendiri sebanyak n kali dalam sebuah perulangan.

   PangkatDC menggunakan metode divide conquer, yang mana masalah dibagi menjadi submasalah yang lebih kecil. Jika n genap, pangkat bilangan dapat dihitung sebagai hasil perkalian dari pangkat bilangan setengah dan dirinya sendiri. Jika n ganjil, pangkat bilangan dapat dihitung sebagai hasil perkalian dari pangkat bilangan setengah, dirinya sendiri, dan a.

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

   Answer:

   Tahap "combine" terjadi pada saat menggabungkan solusi dari submasalah yang lebih kecil untuk membentuk solusi akhir. Dalam metode pangkatDC(), tahap combine terjadi ketika hasil pangkat dari bilangan setengah dikalikan dengan dirinya sendiri atau dengan a, tergantung pada apakah pangkat n genap atau ganjil.

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

   Tahap "combine" terjadi ketika hasil submasalah digabungkan kembali untuk menghasilkan solusi pangkat yang akhir.

3. Pada method pangkatBF()terdapat parameter untuk melewatkan nilai yang akan dipangkatkan dan pangkat berapa, padahal di sisi lain di class Pangkat telah ada atribut nilai dan pangkat, apakah menurut Anda method tersebut tetap relevan untuk memiliki parameter? Apakah bisa jika method tersebut dibuat dengan tanpa parameter? Jika bisa, seperti apa method pangkatBF() yang tanpa parameter?

NAMA : AFRIZAL RAFLI KUSUMA WARDANA
NIM : 244107020007
NO ABSEN : 01
KELAS : TI-1F
MATERI : Brute Force dan Devide Conquer

Answer:
Method pangkatBF(int a, int n) memiliki parameter a dan n, padahal dalam class pangkat1 sudah ada atribut nilai dan pangkat. Ini berarti method tersebut sebenarnya tidak perlu menerima parameter karena bisa langsung menggunakan atribut yang tersedia dalam objek. method pangkatBF() bisa dibuat tanpa parameter dengan menggunakan atribut nilai dan pangkat langsung.

4. Tarik tentang cara kerja method pangkatBF() dan pangkatDC()!
Answer:

Method pangkatBF() menggunakan pendekatan iteratif untuk menghitung pangkat. Cara kerjanya sebagai berikut:

1. Inisialisasi variabel hasil dengan 1 sebagai nilai awal.
2. Menggunakan perulangan for yang berjalan sebanyak n kali, di mana n adalah pangkat yang diberikan.
3. Pada setiap iterasi, hasil dikalikan dengan a (basis) untuk menghitung hasil perpangkatan secara bertahap.
4. Setelah perulangan selesai, nilai hasil dikembalikan sebagai output.

Method pangkatDC() menggunakan pendekatan rekursif berbasis Divide and Conquer untuk menghitung pangkat. Cara kerjanya sebagai berikut:

1. Kasus dasar:
    - Jika n == 1, maka langsung mengembalikan a (karena a^1 = a).
2. Kasus rekursif:
    - Jika n adalah bilangan genap, maka a^n dihitung sebagai: an=(an/2)×(an/2)
    - Jika n adalah bilangan ganjil, maka a^n dihitung sebagai:

      an=(an/2)x(an/2)xa
3. Rekursi terus berjalan hingga mencapai n == 1, setelah itu hasilnya dikalikan dan dikembalikan secara bertahap.

NAMA	: AFRIZAL RAFLI KUSUMA WARDANA
NIM	: 244107020007
NO ABSEN	: 01
KELAS	: TI-1F
MATERI	: Brute Force dan Devide Conquer

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

Class : Sum

```java
public class Sum {
    double keuntungan[];
    Sum (int el){
        keuntungan = new double[el];
    }
    Codeium: Refactor | Explain | Generate Javadoc | ×
    double totalBF(){
        double total = 0;
        for (int i = 0; i < keuntungan.length; i++) {
            total += keuntungan[i];
        }
        return total;
    }
    Codeium: Refactor | Explain | Generate Javadoc | ×
    double totalDC(double arr[], int l, int r){
        if (l==r){
            return arr[l];
        }
        int mid = (l+r)/2;
        double lsum = totalDC(arr, l, mid);
        double rsum = totalDC(arr, mid+1, r);
        return lsum + rsum;
    }
}
```

Class : MainSum

```java
import java.util.Scanner;
Codeium: Refactor | Explain
public class MainSum{
    Run main | Debug main | Run | Debug | Codeium: Refactor | Explain | Generate Javadoc | ×
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print(s:"Masukkan jumlah elemen: ");
        int elemen = input.nextInt();
        Sum sm = new Sum(elemen);
        for(int i = 0; i < elemen; i++) {
            System.out.print("Masukkan keuntungan ke- " + (i+1) + ": ");
            sm.keuntungan[i] = input.nextInt();
        }
        System.out.println("Total keuntungan menggunakan Bruteforce: " + sm.totalBF());
        System.out.println("Total keuntungan menggunakan Divide Conquer: " + sm.totalDC(sm.keuntungan, l:0, elemen-1));
    }
}
```

Question :

1. Kenapa dibutuhkan variable mid pada method TotalDC()?
   Answer:
   Memisahkan array, variabel mid digunakan untuk membagi array menjadi dua bagian secara rekursif, yaitu bagian kiri dan bagian kanan.

NAMA : AFRIZAL RAFLI KUSUMA WARDANA
NIM : 244107020007
NO ABSEN : 01
KELAS : TI-1F
MATERI : Brute Force dan Devide Conquer

Jadi, return value ini adalah total dari semua elemen di sebelah kiri, di sebelah kanan, dan elemen pada mid, yang merupakan total keseluruhan dari array tersebut yang mengginakan rekursif dengan metode divide and conquer.

2. Untuk apakah statement di bawah ini dilakukan dalam TotalDC()?

```
double lsum = totalDC(arr, l, mid);
double rsum = totalDC(arr, mid+1, r);
```

Answer:

lsum menghitung jumlah elemen dari indeks l hingga mid, sedangkan rsum menghitung jumlah dari mid+1 hingga r. Ini adalah bagian dari teknik Divide and Conquer yang memecah masalah menjadi bagian yang lebih kecil.

3. Kenapa diperlukan penjumlahan hasil lsum dan rsum seperti di bawah ini?
Answer:
Setelah hasil perhitungan dari dua bagian diperoleh (lsum dan rsum), kedua hasil tersebut dijumlahkan untuk mendapatkan total keseluruhan**.**

4. Apakah base case dari totalDC()?
Answer:
Base case dari totalDC() adalah ketika l == r. Dalam kondisi ini, hanya ada satu elemen yang tersisa dalam array, sehingga langsung dikembalikan tanpa perlu pemrosesan lebih lanjut

5. Tarik Kesimpulan tentang cara kerja totalDC()
Answer:
totalDC() bekerja dengan menggunakan metode Divide and Conquer, yang terdiri dari tiga tahap utama:

- **Divide**: Memecah array menjadi dua bagian dengan menentukan nilai tengah (mid).
- **Conquer**: Memanggil rekursi untuk menghitung jumlah dari masing-masing bagian.
- **Combine**: Menjumlahkan hasil dari dua bagian yang telah dihitung sebelumnya untuk mendapatkan total keseluruhan.

NAMA	: AFRIZAL RAFLI KUSUMA WARDANA
NIM	: 244107020007
NO ABSEN	: 01
KELAS	: TI-1F
MATERI	: Brute Force dan Devide Conquer

4.5 Latihan Praktikum

Class Mahasiswa1:

```java
public class Mahasiswa1 {
    String nama;
    String nim;
    int tahunMasuk;
    int nilaiUTS;
    int nilaiUAS;

    Mahasiswa1(String nama, String nim, int tahunMasuk, int nilaiUTS, int nilaiUAS) {
        this.nama = nama;
        this.nim = nim;
        this.tahunMasuk = tahunMasuk;
        this.nilaiUTS = nilaiUTS;
        this.nilaiUAS = nilaiUAS;
    }
}
```

Class DataMhs:

```java
public class DataMhs {
    Mahasiswa1[] mahasiswa;
    int maxUTS, minUTS;
    double rataRataUAS;

    DataMhs(Mahasiswa1[] mahasiswa) {
        this.mahasiswa = mahasiswa;
    }

    int hitungMaxUTS(int l, int r) {
        if (l == r) {
            if (maxUTS < mahasiswa[l].nilaiUTS) maxUTS = mahasiswa[l].nilaiUTS;
        } else {
            int mid = (l + r) / 2;
            hitungMaxUTS(l, mid);
            hitungMaxUTS(mid + 1, r);
        }
        return maxUTS;
    }

    int hitungMinUTS(int l, int r) {
        if (l == r) {
            if (minUTS > mahasiswa[l].nilaiUTS) minUTS = mahasiswa[l].nilaiUTS;
        } else {
            int mid = (l + r) / 2;
            hitungMinUTS(l, mid);
            hitungMinUTS(mid + 1, r);
        }
        return minUTS;
    }
    int hitungRataRataUAS() {
        double total = 0;
        for (Mahasiswa1 m : mahasiswa) {
            total += m.nilaiUAS;
        }
        rataRataUAS = total / mahasiswa.length;
        return (int) rataRataUAS;
    }
}
```

NAMA	: AFRIZAL RAFLI KUSUMA WARDANA
NIM	: 244107020007
NO ABSEN	: 01
KELAS	: TI-1F
MATERI	: Brute Force dan Devide Conquer

Class MainDatMhs:

```
P.5 > J MainDatMhs.java > Language Support for Java(TM) by Red Hat > MainDatMhs > main(String[])
1  public class MainDatMhs {
2      public static void main (String [] args) {
3          Mahasiswa1[] data = {
4              new Mahasiswa1(nama:"Ahmad", nim:"220101001", tahunMasuk:2022, nilaiUTS:78, nilaiUAS:82),
5              new Mahasiswa1(nama:"Budi", nim:"220101002", tahunMasuk:2022, nilaiUTS:85, nilaiUAS:88),
6              new Mahasiswa1(nama:"Cindy", nim:"220101003", tahunMasuk:2021, nilaiUTS:90, nilaiUAS:87),
7              new Mahasiswa1(nama:"Dian", nim:"220101004", tahunMasuk:2021, nilaiUTS:76, nilaiUAS:79),
8              new Mahasiswa1(nama:"Eko", nim:"220101005", tahunMasuk:2023, nilaiUTS:92, nilaiUAS:95),
9              new Mahasiswa1(nama:"Fajar", nim:"220101006", tahunMasuk:2020, nilaiUTS:88, nilaiUAS:85),
10             new Mahasiswa1(nama:"Gina", nim:"220101007", tahunMasuk:2023, nilaiUTS:80, nilaiUAS:83),
11             new Mahasiswa1(nama:"Hadi", nim:"220101008", tahunMasuk:2020, nilaiUTS:82, nilaiUAS:84)
12         };
13
14         DataMhs dm = new DataMhs(data);
15
16         dm.maxUTS = data[0].nilaiUTS;
17         dm.minUTS = data[0].nilaiUTS;
18         dm.hitungMaxUTS(i:0, data.length - 1);
19         dm.hitungMinUTS(i:0, data.length - 1);
20         dm.hitungRataRataUAS();
21
22         System.out.println("Nilai UTS tertinggi: " + dm.maxUTS);
23         System.out.println("Nilai UTS terendah: " + dm.minUTS);
24         System.out.println("Rata-rata nilai UAS: " + dm.rataRataUAS);
25     }
26 }
```

Output:

```
Nilai UTS tertinggi: 92
Nilai UTS terendah: 76
Rata-rata nilai UAS: 85.375
```

Github:

https://github.com/rizalrfli/Prk.ASD/tree/main/P.5