LAPORAN HASIL PRAKTIKUM
ALGORITMA STRUKTUR DATA
JOBSHEET 5

NAMA : JIRO AMMAR WAFI

NIM : 244107020190

KELAS : 1-E

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

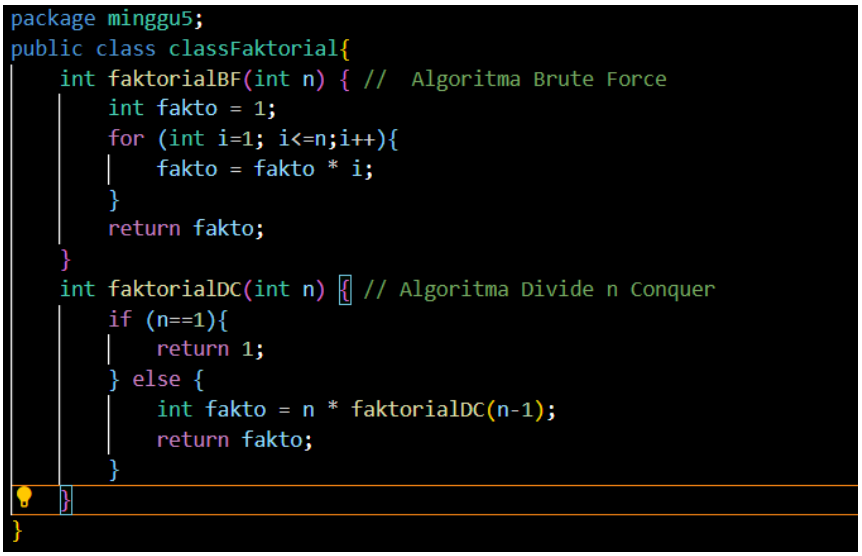
POLINEMA

2025

PERCOBAAN 1

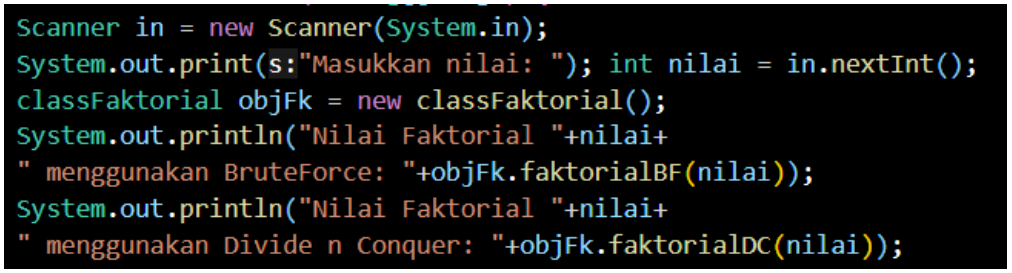
Menghitung Nilai Faktorial dengan Algoritma Brute Force dan Divide and Conquer

1. Membuat package dan class yang berisi method sesuai dengan diagram:



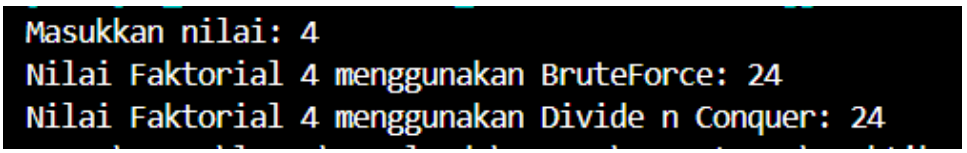
```
package minggu5;
public class classFaktorial{
    int faktorialBF(int n) { //  Algoritma Brute Force
        int fakto = 1;
        for (int i=1; i<=n;i++){
            fakto = fakto * i;
        }
        return fakto;
    }
    int faktorialDC(int n) { // Algoritma Divide n Conquer
        if (n==1){
            return 1;
        } else {
            int fakto = n * faktorialDC(n-1);
            return fakto;
        }
    }
}
```

2. Membuat line untuk meminta input pengguna, membuat objek dari class, dan memanggil method yang berada di class.



```
Scanner in = new Scanner(System.in);
System.out.print(s:"Masukkan nilai: "); int nilai = in.nextInt();
classFaktorial objFk = new classFaktorial();
System.out.println("Nilai Faktorial "+nilai+
" menggunakan BruteForce: "+objFk.faktorialBF(nilai));
System.out.println("Nilai Faktorial "+nilai+
" menggunakan Divide n Conquer: "+objFk.faktorialDC(nilai));
```

3. Verifikasi Kode:



```
Masukkan nilai: 4
Nilai Faktorial 4 menggunakan BruteForce: 24
Nilai Faktorial 4 menggunakan Divide n Conquer: 24
```

PERTANYAAN P1

1. Pada base line Algoritma Divide Conquer untuk melakukan pencarian nilai faktorial, jelaskan perbedaan bagian kode pada penggunaan if dan else!
   >
   Bagian IF, memeriksa jika angka N merupakan angka 1, jika iya, akan mengembalikan angka 1 yang menunjukkan indikasi bahwa rekursif telah berakhir. Jika tidak, masuk ke bagian ELSE, titik inilah dimana rekursif akan terus bekerja hingga menyentuh baseline IF yaitu N mencapai angka 1.

2. Apakah memungkinkan perulangan pada method faktorialBF() diubah selain menggunakan for? Buktikan!
   >
   Bisa menggunakan while, karena nilai N sudah diketahui berdasarkan input pengguna;

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

3. Jelaskan perbedaan antara fakto *= i; dan int fakto = n * faktorialDC(n-1); !
   >
   A = [fakto *= i] & B = [int fakto = n * faktorialDC(n-1)]
   A merupakan operator yang *langsung mengoperasikan* perkalian nilai fakto dengan nilai i dari perulangan, sedangkan ***B*** merupakan operator yang menambahkan fungsi rekursif terlebih dahulu hingga mencapai baseline nya yaitu n = 1. Setelah tercapai, barulah operasi perkalian dapat dijalankan menggunakan angka2 yang dihasilkan dari fungsi rekursif.

4. Buat Kesimpulan tentang perbedaan cara kerja method faktorialBF() dan faktorialDC()!
   >
   faktorialBF() akan langsung mengalikan, sedangkan faktorialDC() perlu memanggil fungsi rekursif dahulu hingga berakhir agar dapat mengoperasikan perkalian.

PERCOBAAN 2

Menghitung Hasil Pangkat dengan Algoritma Brute Force dan Divide and Conquer

1. Membuat class yang berisi atribut dan method, beserta konstruktor default dan berparameter di dalam package minggu5.

```
package minggu5;

public class classPangkat {
    int nilai, pangkat;
    public classPangkat () {}

    public classPangkat (int n, int p){
        nilai = n;
        pangkat = p;
    }

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

            }
        }
    }
}
```

2. Membuat class Main, command line untuk meminta input pengguna, deklarasi & instansiasi objek, beserta baris untuk menampilkan output dari prosesnya :

```
public class MainPangkat{
    Run | Debug
    public static void main(String[] args) {
        Scanner in = new Scanner (System.in);
        System.out.print(s:"Masukkan jumlah elemen: ");
        int elemen = in.nextInt();

        classPangkat[] png = new classPangkat[elemen];
        for(int i=0; i<elemen; i++){
            System.out.print("\nMasukkan nilai basis elemen ke-"+(i+1)+": ");
            int basis = in.nextInt();
            System.out.print("Masukkan nilai pangkat elemen ke-"+(i+1)+": ");
            int pangkat = in.nextInt();
            png[i] = new classPangkat(basis, pangkat);
        }
        System.out.println(x:"\nHASIL PANGKAT BRUTEFORCE:");
        for (classPangkat p : png) {
            System.out.println(p.nilai+"^"+p.pangkat+": "
            +p.pangkatBF(p.nilai, p.pangkat));
        }
        System.out.println(x:"\nHASIL PANGKAT DIVIDE N CONQUER:");
        for (classPangkat p : png) {
            System.out.println(p.nilai+"^"+p.pangkat+": "
            +p.pangkatDC(p.nilai, p.pangkat));
        }
        in.close();
    }
```

3. Verifikasi Kode:

```
Masukkan jumlah elemen: 2

Masukkan nilai basis elemen ke-1: 3
Masukkan nilai pangkat elemen ke-1: 4

Masukkan nilai basis elemen ke-2: 5
Masukkan nilai pangkat elemen ke-2: 6

HASIL PANGKAT BRUTEFORCE:
3^4: 81
5^6: 15625

HASIL PANGKAT DIVIDE N CONQUER:
3^4: 81
5^6: 15625
PS C:\Users\luvma\Downloads\KULIAH\Semeste
```

PERTANYAAN P2

1. Method pangkatBF() menginisialisasikan variabel lokal yang akan dijadikan sebagai nilai kembalian, menggunakan looping for untuk mengoperasikan perkalian pangkat dengan mengalikan nilai hasil dengan nilai dari input pengguna dan sebanyak pangkat (n).

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

   Method pangkatDC() membuat pemilihan yang bertujuan untuk menjalankan fungsi rekursif hingga berakhir di pemilihan (n==1) dan mengembalikan nilai awalnya (a). Menggunakan nilai pangkat sebagai penyeleksi, dengan nilai kembalian yaitu hasil dari perkalian antar dua fungsi rekursif yang telah dipecah.

```
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));

        }
    }
}
```

2. Tahap combine sudah ada dalam baris;

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

3. Method tanpaParameter():

```
int tanpaParameter(){
    int hasil = 1;
    for (int i=0; i<pangkat; i++){
        hasil *= nilai;
    } return hasil;
}
```

Method ini bisa dibuat karena instansiasi objek awal telah menggunakan variabel dari class MainPangkat yang dipanggil menggunakan konstruktor berparameter di class classPangkat untuk digunakan sebagai atribut dalam classPangkat.

4. Cara Kerja:
   Sampel: Nilai = 2 (a), Pangkat = 3 (n)
   A. pangkatBF();
      hasil = 1;
      iterasi dilakukan sebanyak 3 kali (i=0, i<3);
      iterasi pertama (i=0) || hasil = 1 * 2, hasil = 2;
      iterasi kedua (i=1) || hasil = 2 * 2, hasil = 4;
      iterasi ketiga (i=2) || hasil = 4 * 2, hasil = 8;
      return = 8;

   B. pangkatDC();
      || 1st Rekursif || pangkatDC (2,3)
      if pertama = false;
      else, if (n%2==1) = true;
      return (pangkatDC (2, 3/2) * pangkatDC (2, 3/2) * 2);

      || 2nd Rekursif || pangkatDC (2, 3/2)
      if kedua = true;
      return 2;

      return:
      = (pangkatDC (2, 3/2) * pangkatDC (2, 3/2) * 2);
      = 2 * 2 * 2;
      = 8;

PERCOBAAN 3

Menghitung Sum Array dengan Algoritma Brute Force dan Divide and Conquer

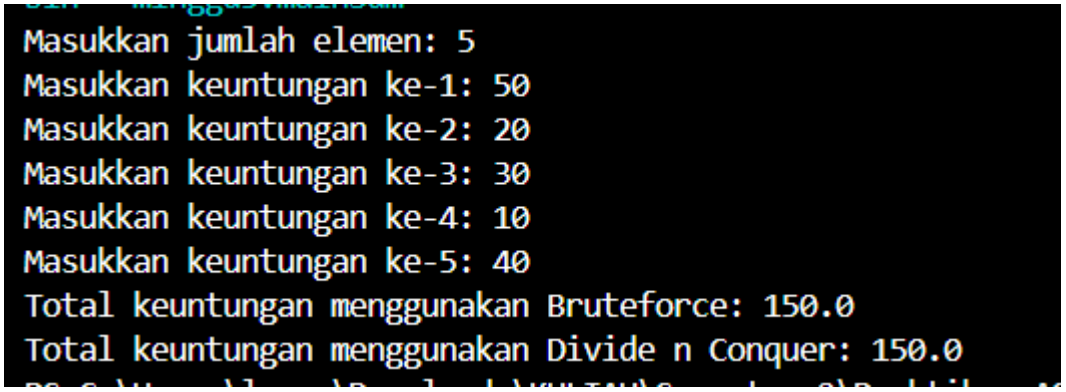
1. Membuat class untuk diberi atribut, method, dan konstruktor berparameter:

```java
public class classSum {
    double keuntungan[];

    public classSum(int el){
        keuntungan = new double[el];
    }

    double totalBF() {
        double total = 0;
        for(int i=0; i<keuntungan.length;i++){
            total = total + keuntungan[i];
        } return total;
    }

    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;
    }
}
```

2. Membuat class Main yang meminta input pengguna yang digunakan untuk inisialisasi panjang array, menginisialisasi nilai array of object menggunakan perulangan, dan menampilkannya sesuai method:

```java
import java.util.Scanner;
public class mainSum {
    Run | Debug
    public static void main(String[] args) {
        Scanner in = new Scanner (System.in);
        System.out.print(s:"Masukkan jumlah elemen: ");
        int elemen = in.nextInt();

        classSum sm = new classSum(elemen);
        for (int i=0;i<elemen;i++){
            System.out.print("Masukkan keuntungan ke-"+(i+1)+": ");
            sm.keuntungan[i] = in.nextDouble();
        }
        System.out.println("Total keuntungan menggunakan Bruteforce: "+sm.totalBF());
        System.out.println("Total keuntungan menggunakan Divide n Conquer: "
        +sm.totalDC(sm.keuntungan,l:0,elemen-1));
        in.close();
    }
}
```

3. Verifikasi Kode:



PERTANYAAN P3

1. Variabel Mid pada method totalDC() diperlukan untuk memecah array menjadi sub array yang lebih kecil, sesuai dengan konsep Divide yaitu memecah masalah menjadi sub masalah.

2. Pada baris ini, 2 variabel (lsum & rsum) bertujuan untuk menyimpan nilai dari eksekusi method secara rekursif untuk menyelesaikan sub array yang telah dipecah sebelumnya.

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

3. Setelah total masing-masing nilai telah ditemukan oleh setiap method yang dipanggil, kedua isi array tersebut dijumlahkan yang menjadi hasil akhir dari total array.

4. Base case dari method totalDC():

```
if (l==r){
    return arr[l];
}
```

5. totalDC(): membagi elemen array yang diklasifikasikan menjadi 2 sisi array, yang akan berakhir menjadi sub array yang hanya memiliki 1 elemen masing-masing. Akhirnya, kedua sisi dijumlahkan menjadi hasil akhir dari operasi.