

**LAPORAN PRAKTIKUM TEKNIK PEMROGRAMAN  
PENANGANAN REKURSIF**

Laporan Praktikum:  
Disusun untuk memenuhi tugas matakuliah  
Teknik Pemrograman Kelas A



**UIN SUNAN AMPEL  
S U R A B A Y A**

Oleh:  
Anang Misbakhul Khoir (09040620047)

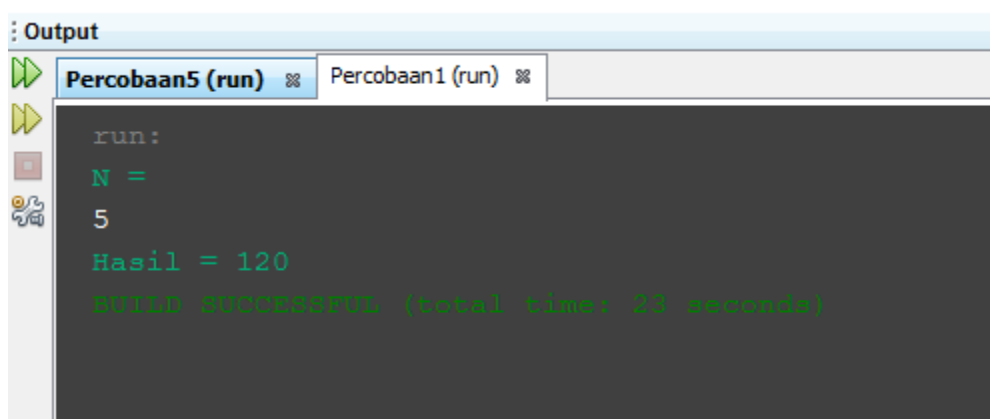
Dosen Pengampu:  
Dwi Rolliawati, MT

**PROGRAM STUDI SISTEM INFORMASI  
FAKULTAS SAINS DAN TEKNOLOGI  
UIN SUNAN AMPEL SURABAYA**

1. Percobaan nomer satu

```
6 package percobaan1;
7 import java.util.Scanner;
8
9 /**
10  * @author Anang Misbakhul Khoir
11  * nim 09040620047
12  */
13
14 public class Percobaan1 {
15     // Fungsi rekursif untuk menghitung nilai faktorial
16
17     public static int faktorial (int x){
18         if (x==1)
19             return x;
20         else
21             return x * faktorial (x-1);
22     }
23
24
25     public static void main(String[] args) {
26         Scanner input = new Scanner(System.in);
27         System.out.println("N = ");
28         int n = input.nextInt();
29         System.out.println("Hasil = " + faktorial (n) );
30     }
31
32 }
```

Pada syntax diatas kita memulai input secara scanner di line 26 dan kemudian diproses di line ke 18 – 21 yang mana jika inputan angka tersebut 5 maka akan langsung diproses di line 21 dikalikan dengan angka yang sama pada perkalian pertama dan kemudian di hasil nya dikalikan dengan angka perkalian pertama tapi dikurangi 1, dan proses tersebut angka yang dikurangi sampai dengan 1 dan kemudian hasilnya keluarkan.



```
Output
Percobaan5 (run) Percobaan1 (run)
run:
N =
5
Hasil = 120
BUILD SUCCESSFUL (total time: 23 seconds)
```

## 2. Percobaan nomor dua

```
6 package percobaan2;
7
8 /**
9  * @author Anang Misbakhul Khoir
10  * nim 09040620047
11  */
12
13 public class Percobaan2 {
14     // Fungsi rekursi untuk menampilkan deret fibonanci
15
16     public static int fibbon (int x){
17         if (x <= 0 || x <= 1)
18             return x;
19         else
20             return fibbon ( x - 2 ) + fibbon ( x - 1 );
21     }
22
23     public static void main(String[] args) {
24         int n = 10;
25         for (int i = 0; i < n; i++)
26             System.out.println( "f" + i + " = " + fibbon(i) );
27     }
28 }
29
```

Pada proses fibonaci

Yang mana fibinaci itu dimisalkan jika fibonaci4 = berisikan angka 5, berarti angka lima tersebut dihasilkan dari perjumlahan fibonaci 3 dan 2 yang berisikan angka 3 pada fibonaci 3 dan angka 2 pada fibonaci 2. Seperti itu kurang lebihnya untuk logikanya.

Output

```
Percobaan5 (run)  percobaan2 (run)

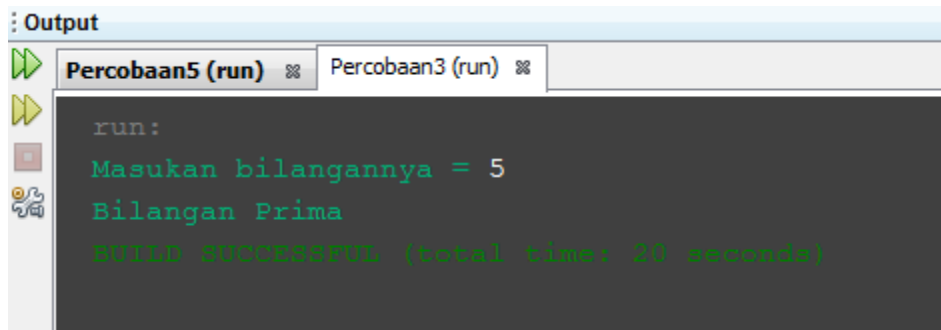
run:
f0 = 0
f1 = 1
f2 = 1
f3 = 2
f4 = 3
f5 = 5
f6 = 8
f7 = 13
f8 = 21
f9 = 34
BUILD SUCCESSFUL (total time: 0 seconds)
```

3. Percobaan nomor tiga

```
6 package percobaan3;
7 import java.util.Scanner;
8
9 /**
10  * @author Anang Misbakhul Khoir
11  * nim 09040620047
12  */
13
14 public class Percobaan3 {
15     //Fungsi rekursi untuk menentukan bilangan prima atau bukan prima
16
17     private static int ambilNilaiRekursif (int number, int index){
18         if (index == 1){
19             return 1;
20         }
21         else if (number % index == 0){
22             return 1 + ambilNilaiRekursif (number, -- index);
23         }
24         else {
25             return 0 + ambilNilaiRekursif (number, -- index);
26         }
27     }
28
29     public static boolean cekBilanganPrima (int num) {
30         if (num > 1){
31             return (ambilNilaiRekursif(num, num) == 2);
32         }
33         else
```

```
33         else
34             return false;
35     }
36
37     public static void main (String[] args){
38         Scanner input = new Scanner (System.in);
39         System.out.print("Masukan bilangannya = ");
40         int num = input.nextInt();
41
42         if(cekBilanganPrima(num)){
43             System.out.println("Bilangan Prima");
44         }
45         else{
46             System.out.println("Bukan Bilangan Prima");
47         }
48     }
49 }
50
```

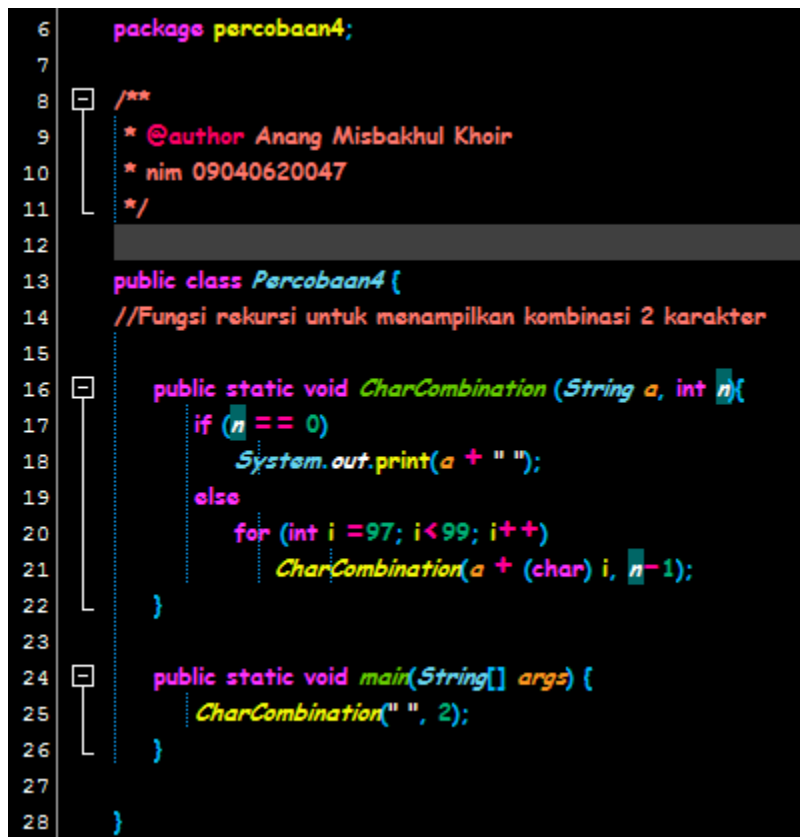
Dari percobaan ke 3 ini mengoutputkan hasil dari pembagian %2 jika angka tersebut bisa dibagi maka angka inputan tadi bukanlah angka bilangan prima dan jika angka prima tidak akan bisa dibagi dengan angka 2 dan akan habis jika anangkanya dibagi dengan dirinya sendiri dan angka 1.



```
Output
Percobaan5 (run)  Percobaan3 (run)

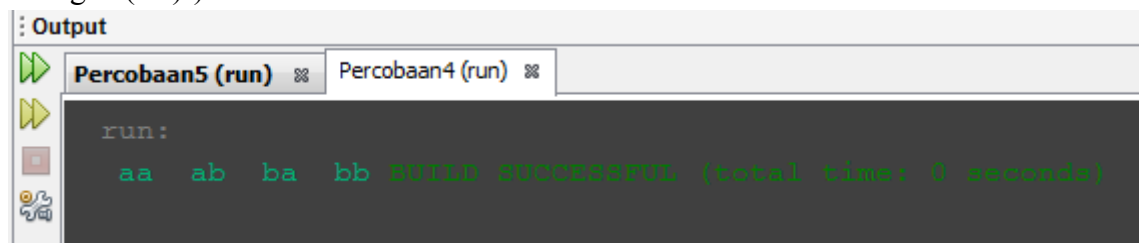
run:
Masukan bilangannya = 5
Bilangan Prima
BUILD SUCCESSFUL (total time: 20 seconds)
```

#### 4. Percobaan nomor empat



```
6 package percobaan4;
7
8 /**
9  * @author Anang Misbakhul Khoir
10  * nim 09040620047
11  */
12
13 public class Percobaan4 {
14     //Fungsi rekursi untuk menampilkan kombinasi 2 karakter
15
16     public static void CharCombination (String a, int n){
17         if (n == 0)
18             System.out.print(a + " ");
19         else
20             for (int i = 97; i < 99; i++)
21                 CharCombination(a + (char) i, n-1);
22     }
23
24     public static void main(String[] args) {
25         CharCombination("aa", 2);
26     }
27
28 }
```

Pada langkah ini kita membuat sebuah kombinasi dari 2 angka seperti dengan rumus  $(n! / (r!(n-r)!))$



```
Output
Percobaan5 (run)  Percobaan4 (run)

run:
aa ab ba bb BUILD SUCCESSFUL (total time: 0 seconds)
```

5. Percobaan nomor lima

```
6 package percobaan5;
7 import java.util.Scanner;
8
9 /**
10  * @author Anang Misbakhul Khoir
11  * nim 09040620047
12  */
13
14 public class Percobaan5 {
15     //Fungsi rekursi untuk menghitung pangkat
16
17     public static int pangkatRekursif(int x, int y){
18         if (y == 0)
19             return 1;
20         else return x * pangkatRekursif(x, y - 1);
21     }
22
23     public static void main(String[] args) {
24         System.out.println("x adalah Bilangan, y adalah pangkat");
25         Scanner input = new Scanner(System.in);
26         System.out.println("Bilangan x : ");
27         int x = input.nextInt();
28
29         System.out.println("Bilangan y : ");
30         int y = input.nextInt();
31
32         System.out.println(x + " dipangkatkan " + y + " = " + pangkatRekursif(x,y));
33     }
34 }
```

Pada awal kodingan ini yang diproses pertama kali yaitu kita menginputkan angka diline ke 26 dan meng inputkan pangkat yang diinginkan untuk si angka pada line 29, kemudian diproses pada line 18 yang mana jika pangkat pada angka berisikan 0 maka langsung di proses output 1, dan jika angka dengan input an pangkat dimisalkan 4 maka ada diproses dielse pada line 20 sampai angka hasil dengan perkalian sampai 1 dan kemudian dioutputkan hasil dari semua perkalian tadi.

Output - Percobaan5 (run)

```
run:
x adalah Bilangan, y adalah pangkat
Bilangan x :
5
Bilangan y :
3
5 dipangkatkan 3 = 125
BUILD SUCCESSFUL (total time: 8 seconds)
```

1. Latihan pertama (

```
6 package latihan1;
7 import java.util.*;
8
9 /**
10  * @author Anang Misbakhul Khoir
11  * nim 09040620047
12  */
13
14 public class Latihan1 {
15     //rekursif untuk menghitung segitiga Pascal
16
17     static long faktorial(int n){
18         long z = 1;
19         int i = 1;
20         while (i <= n){
21             z = z * i;
22             i++;
23         }
24         return z;
25     }
26
27     public static void main(String[] args) {
28         int a,i,j;
29
30         Scanner scan = new Scanner(System.in);
31         System.out.print("Masukan nilai : ");
32         a = scan.nextInt();
33     }
```

```
33
34         for (i = 0; i < a; i++){
35             for (j = 0; j < a-i-1; j++){
36                 System.out.print(" ");
37             }
38             for (j = 0; j <= i; j++){
39                 System.out.print(faktorial(i)/(faktorial(j)*faktorial(i-j)) + " ");
40             }
41             System.out.println();
42         }
43     }
44 }
```

Disini segitiga pascal hampir mirip dengan proses fibonaci yang mana hasil dibawah adalah dari proses penjumlahan angka diatasnya tetapi secara terus menerus dari kedua hasil yang sama ditambahkan 2 angka satu disamping kanan dan kiri.

```
Output - Latihan1 (run)

run:
Masukan nilai : 6
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
BUILD SUCCESSFUL (total time: 6 seconds)
```

## 2. Latihan kedua (

```
6 package latihan2;
7 import java.util.Scanner;
8
9 /**
10  * @author Anang Misbakhul Khoir
11  * nim 09040620047
12  */
13
14 public class Latihan2 {
15     //Fungsi rekursi untuk menampilkan kombinasi 3 karakter
16
17     public static void Kombinasi (String x, int b, int c){
18         if(b==0)
19             System.out.print(x + " ");
20         else
21             for(int i = 97; i < 97 + c; i++)
22                 Kombinasi (x + (char) i , b-1, c);
23     }
24
25     public static void main(String[] args) {
26         System.out.print("Masukkan Jumlah Karakter : ");
27         Scanner masuk = new Scanner(System.in);
28         int mix = masuk.nextInt();
29         Kombinasi(" ", mix, mix);
30         System.out.println(" ");
31     }
32 }
```

Pada proses kali ini sebenarnya sama dengan proses percobaan yang ada di percobaan 4 dengan rumus  $(n!) / (r!(n-r)!)$



Output - Latihan2 (run)

```
run:
Masukkan Jumlah Karakter : 3
aaa aab aac aba abb abc aca acb acc baa bab bac bba bbb
BUILD SUCCESSFUL (total time: 3 seconds)
```

```
b bc bbc bca bcb bcc caa cab cac cba cbb cbc cca ccb ccc
```

3. Latihan ketiga (

```
6 package latihan3;
7
8 /**
9  * @author Anang Misbakhul Khoir
10  * nim 09040620047
11  */
12
13 public class Latihan3 {
14     //program BinarySearch dengan Rekursif
15
16     public static void main(String[] args) {
17
18         int angka [] = {1,2,5,7,8,9,12}; // deret angka yng sudah terurut
19
20         int kunci = 7; //key atau kunci yang akan dicari
21         int index = angka.length / 2;
22         boolean ketemu = false;
23
24         int tengah = index;
25         while (index >=0 && index < angka.length && ketemu == false){
26
```

```

26
27     if ( kunci == angka[index]){
28         System.out.println(" data ditemukan pada index ke- " + index);
29         ketemu = true;
30     }
31
32     else{
33         if(kunci < angka[tengah]){
34             index--;
35         }else{
36             index++;
37         }
38     }
39 }
40
41 if (ketemu == false){
42     System.out.println("Data tidak ditemukan");
43 }
44 }
45 }

```

Output - Latihan3 (run)

```

run:
    data ditemukan pada index ke- 3
BUILD SUCCESSFUL (total time: 0 seconds)

```

#### 4. Latihan keempat (

```
6 package latihan4;
7 import java.util.Scanner;
8
9 /**
10  * @author Anang Misbakhul Khoir
11  * nim 09040620047
12  */
13
14 public class Latihan4 {
15     //program rekursif untuk memecahkan permasalahan Menara Hanoi
16     static int move = 1;
17
18     public static void main(String[] args) {
19         Scanner input = new Scanner(System.in);
20         System.out.print("Masukkan Jumlah Cakram / Disc : ");
21         int Cakram = input.nextInt();
22         hanoi(Cakram, 'A', 'B', 'C');
23     }
24     static void hanoi(int Cakram, char awal, char bantu, char tujuan){
25         if (Cakram >= 1){
26             hanoi(Cakram-1, awal, tujuan, bantu);
27             move(Cakram, awal, tujuan);
28             hanoi(Cakram-1, bantu, awal, tujuan);
29         }
30     }
31     static void move(int step, char awal, char tujuan){
32         System.out.println("Langkah " + move);
33         move++;
34         System.out.print("Pindahkan Cakram " + step);
35         System.out.print("dari " + awal);
36         System.out.println("ke " + tujuan);
37     }
38 }
```

Maksud dari coding diatas ialah memindahkan disc yang ada di A ke C dan B adalah sebagai jembatan perantara pemindah dari A ke C.

```
Output - Latihan4 (run)

run:
Masukkan Jumlah Cakram / Disc : 3
Langkah 1
Pindahkan Cakram 1dari Ake C
Langkah 2
Pindahkan Cakram 2dari Ake B
Langkah 3
Pindahkan Cakram 1dari Cke B
Langkah 4
Pindahkan Cakram 3dari Ake C
Langkah 5
Pindahkan Cakram 1dari Bke A
Langkah 6
Pindahkan Cakram 2dari Bke C
Langkah 7
Pindahkan Cakram 1dari Ake C
BUILD SUCCESSFUL (total time: 3 seconds)
```

#### 5. Latihan kelima

```
public static void decToBin(int num)
{
    if (num > 0)
    {
        decToBin(num / 2);
        System.out.print(num % 2);
    }
}
```

Pada program Rekursif diatas menjelaskan sebuah proses dengan menginisialkan tipe data integer dengan variabel num. Selanjutnya dilakukan proses percabangan, apabila variabel num nilainya > 0 maka harus dibagi 2. Setelah itu menampilkan nilai sisa bagi 2 variabel num.

#### 6. Latihan keenam

```
test("01101", 4)!
int test(String s, int last) { if (last < 0) {
    if (last < 0) {
        return 0;
    }
    if (s.charAt(last) == "0") {
        return 2 * test(s, last-1);
    }
    return 1 + 2 * test(s, last-1);
}
```

,Menggunakan public class dengan nilai untuk tipe data yang akan dideklarasikan  
 ,Memanggil class test dengan mendeklarsikan tipe data yang sudah dibuat oleh class test  
 ,Cek kondisi dengan perintah if() apakah variabel dari last berjumlah lebih dari 0  
 Jika benar maka akan mengembalikan nilai 0  
 ,Cek kondisi dengan perintah if() apakah nilai yang sudah dikembalikan dari nilai char ke index last berjumlah 0, jika benar maka program akan mengembalikan hasil dari 2 dikali dengan hasil dari string s dan int last dikurangi 1  
 ,Kemudian mengembalikan hasil dari 1 ditambah 2 dikali dengan hasil dari string s dan int last dikurangi 1'

## 7. Latihan ketujuh

```
boolean search(int[] x, int size, int n) {
    if (size > 0) {
        if (x[size-1] == n) {
            return true;
        } else {
            return search(x, size-1, n);
        }
    }
}
```

Pada program Rekursif diatas menjelaskan sebuah proses dengan menginisialkan tipe data integer dengan variabel x, size dan n. Apabila nilai dari variabel size > 0 maka selanjutnya dilakukan proses percabangan dan apabila x dengan size – 1 sama dengan nilai n maka perlu melakukan proses boolean dan jawabannya benar (TRUE). Jika tidak maka melakukan proses dengan menampilkan nilai variabel x, size dan n. Kemudian jika tidak termasuk percabangan pertama maka proses selanjutnya yaitu dengan proses boolean dengan jawaban yang salah (FALSE).

## 8. Latihan kedelapan

```
boolean binarySearch(int[] x, int start, int end, int n) {
    if (end < start) return false;
    int mid = (start+end) / 2;
    if (x[mid] == n) {
        return true;
    } else {
        if (x[mid] < n) {
            return search(x, mid+1, end, n);
        } else {
            return search(x, start, mid-1, n);
        }
    }
}
```

Menggunakan public class boolean binarySearch dengan tipe data integer

,Cek kondisi dengan perintah if() apakah nilai dari variabel end lebih kecil dari nilai variabel start jika benar akan mengembalikan program ke tahap cek kondisi.

.Deklarasi variabel mid yang merupakan hasil dari operasi penambahan variabel start dan end dibagi 2

,Cek kondisi dengan perintah if() apakah nilai dari x mid sama dengan hasil dari variabel n, jika benar maka program melanjutkan ke tahap

.jika salah maka cek kondisi lagi menggunakan perintah else if () apakah nilai dari variabel x mid lebih kecil dari n, jika benar maka program akan mencari dari variabel x, hasil dari mid +1, variabel end, dan n

,jika salah maka program akan mencari lagi nilai dari variabel x, start, hasil dari mid -1, dan n

#### 9. Latihan kesembilan

```
public static int mystery(int a, int b) {  
    if (b == 0) return 0;  
    if (b % 2 == 0) return mystery(a+a, b/2);  
    return mystery(a+a, b/2) + a;  
}
```

Pada program Rekursif diatas menjelaskan sebuah proses dengan menginisialkan tipe data integer dengan variabel mystery yang nantinya tipe publicnya dapat dipanggil, serta variabel a, b. Melakukan percabangan apabila nilai variabel b = 0, sebaliknya apabila variabel b sisa bagi 2 = 0 maka akan memanggil variabel mystery diikuti dengan penjumlahan pada variabel a + a dan variabel b di bagi 2. Jika tidak mau dilakukan uji coba pada percabangan maka langsung lakukan proses dengan memanggil variabel mystery diikuti dengan penjumlahan variabel a + a dan variabel b dibagi 2 kemudian dijumlahkan dengan variabel a.

#### 10. Latihan kesepuluh

```
public static boolean gcdlike(int p, int q) {  
    if (q == 0) return (p == 1);  
    return gcdlike(q, p % q);  
}
```

Pada program Rekursif diatas menjelaskan sebuah proses dengan menginisialkan tipe data boolean dengan variabel gcdlike yang nantinya tipe publicnya dapat dipanggil, serta variabel p dan q. Proses selanjutnya dengan melakukan percabangan apabila nilai variabel q = 0 maka p nilainya 1. Jika tidak perlu melakukan percabangan maka menggunakan proses boolean variabel gcdlike, serta variabel p dan q, dan nilai sisa bagi p dan q.

#### 11. Latihan kesebelas

```
public static String ex234(int n) {  
    if (n <= 0) return "";  
    return ex234(n-3) + n + ex234(n-2) + n;  
}
```

Pada program Rekursif diatas menjelaskan sebuah proses dengan menginisialkan tipe data string dengan variabel ex234, serta variabel n dengan tipe data integer. Melakukan proses percabangan apabila nilai variabel n kurang dari atau sama dengan 0 maka proses akan selesai atau diakhiri. Jika tidak maka akan melakukan proses pemanggilan variabel ex234 kemudian dengan menambahkan rumus sesuai kode program tersebut.

#### 12. Latihan keduabelas

```
public static void collatz(int n) {  
    System.out.print(n + " ");  
    if (n == 1) return;  
    if (n % 2 == 0) collatz(n / 2);  
    else collatz(3*n + 1);  
}
```

Pada program Rekursif diatas menjelaskan sebuah proses dengan menginisialkan tipe data integer dengan variabel n yang nantinya tipe public static void bernama collatz digunakan pada saat pemanggilannya. Melakukan proses percabangan untuk menampilkan variabel n, apabila nilainya 1 maka program tidak bisa merespon prosesnya. Jika sisa bagi 2 nilai variabel n = 0 maka akan dilakukan pemanggilan pada tipe public static void dan melakukan proses menghitung nilai variabel n dibagi 2. Jika tidak maka sama aja akan memanggilnya dan melakukan proses dengan dikalikan 3 pada variabel n kemudian di tambah 1.

#### 13. Latihan ketigabelas

```
public static int mystery(int a, int b) {  
    if (a == b) StdOut.println(a);  
    else {  
  
        int m1 = (a + b) / 2;  
        int m2 = (a + b + 1) / 2;  
        mystery(a, m1);  
        mystery(m2, b);  
    }  
}
```

Pada program Rekursif diatas menjelaskan sebuah proses dengan menginisialkan tipe data integer dengan variabel mystery, serta variabel a dan b. Melakukan proses percabangan apabila variabel a = b maka akan menampilkan nilai a. Jika tidak maka akan melakukan proses dengan inisialisasi m1 dan m2 b dan masing-masing memiliki rumus yang berbeda, kemudian memanggil variabel mystery dan menampilkan variabel a, variabel m1 dan variabel m2, variabel b.

#### 14. Latihan keempatbelas

```
public static int f(int n) {  
    if (n == 0) return 0;  
    if (n == 1) return 1;  
    if (n == 2) return 1;  
    return 2*f(n-2) + f(n-3);  
}
```

Pada program Rekursif diatas menjelaskan sebuah proses dengan menginisialkan tipe data integer dengan variabel f dan variabel n dengan tipe data n. Melakukan proses percabangan apabila nilai variabel  $n = 0$  maka langsung selesai. Jika variabel  $n = 1$  atau 2 bahkan lebih maka melakukan proses return 1 dan jika tidak maka menghitung dengan rumus yang ada.

#### 15. Latihan kelimabelas

```
public static int square(int n) {  
    if (n == 0) return 0;  
    return square(n-1) + 2*n - 1;  
}  
  
public static int cube(int n) {  
    if (n == 0) return 0;  
    return cube(n-1) + 3*(square(n)) - 3*n + 1;  
}
```

Pada program Rekursif diatas menjelaskan sebuah proses dengan menginisialkan tipe data integer dengan variabel square yang nantinya tipe publicnya dapat dipanggil, serta variabel n. Melakukan percabangan apabila nilai variabel  $n = 0$  maka langsung selesai. Jika tidak maka melakukan proses pemanggilan variabel square. Selanjutnya menginisialisasikan variabel cube serta variabel n juga. Melakukan percabangan apabila nilai variabel  $n = 0$  maka langsung selesai. Jika tidak maka melakukan proses pemanggilan pada variabel cube.

### KESIMPULAN

Fungsi rekursif adalah fungsi yang memanggil dirinya sendiri artinya disebut di dalam tubuh fungsi itu sendiri. Fungsi ini akan terus berjalan hingga kondisi lengkap terpenuhi. Oleh karena itu, dalam fungsi rekursif terdapat dua blok penting, yaitu blok yang berfungsi sebagai titik henti dari proses rekursif dan blok yang memanggil dirinya sendiri. Ada banyak parameter yang dilewatkan saat fungsi dipanggil. Ada beberapa aplikasi rekursif yaitu faktorial, Fibonacci, Hanoi, dll. Untuk beberapa aplikasi, pemrograman rekursif itu mudah, tetapi membutuhkan banyak memori.