Fungsi 2

Tim Ajar Matakuliah Dasar Pemrograman 2023

Kompetensi

- Mahasiswa mampu memahami konsep fungsi rekursif
- Mahasiswa mampu menerapkan fungsi rekursif untuk berbagai permasalahan

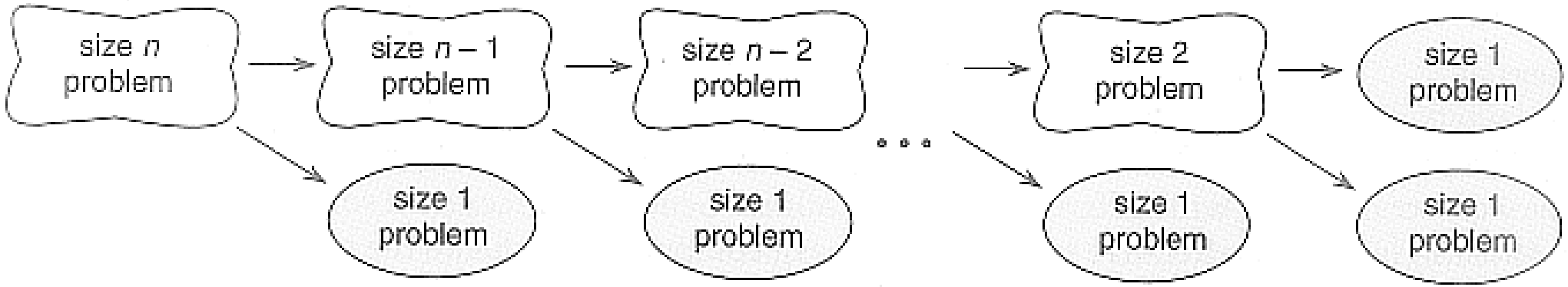
Fungsi Rekursif

- Biasanya sebuah fungsi akan dipanggil (di-CALL) oleh fungsi lain
- Pada fungsi rekursif, di dalam sebuah fungsi terdapat perintah untuk memanggil fungsi itu sendiri (dirinya sendiri). Dengan demikian, proses pemanggilan fungsi akan terjadi secara berulang-ulang
- Bentuk umum:

```
tipe_data_kembalian nama_fungsi (parameter){
    ...
    nama_fungsi(...)
    ...
}
```

Fungsi Rekursif

- Strategi penyelesaian masalah pada kasus rekursif disebut *decrease and conquer*
- Idenya adalah mengurangi ukuran permasalahan sampai menjadi kasus sederhana yang mempunyai penyelesaian jelas



- Fungsi rekursif akan memanggil dirinya sendiri, tetapi nilai parameter yang digunakan pada setiap pemanggilan berbeda

Komponen Fungsi Rekursif

- **Base Case**

  Rekursi berakhir jika base case (nilai batas) terpenuhi
- **Recursion call / Reduction step**

  Fungsi rekursif konvergen (mendekat) ke arah nilai batas

  Biasanya mempunyai keyword **return** untuk mengembalikan nilai ke fungsi yang memanggilnya

Format Fungsi Rekursif

- Pada umumnya format fungsi rekursif mempunyai bentuk sebagai berikut

```
if (nilai batas)
    //menyelesaikan masalah
else
    //mendefinisikan kembali masalah menggunakan rekursi
```

- Cabang IF merupakan **base case**, sedangkan ELSE merupakan **recursion call**
- Bagian recursion call menyediakan pengulangan yang dibutuhkan untuk menyederhanakan permasalahan dan base case menyediakan pengehentian
- Agar rekursi dapat berhenti, **recursion call harus mendekati base case di setiap pemanggilan** fungsi rekursif

Trace Fungsi Rekursif

Eksekusi fungsi rekursif berlangsung dalam dua tahap, yaitu:

- **Fase ekspansi**: pemanggilan fungsi rekursif yang semakin mendekati base case
- **Fase substitusi**: solusi dihitung secara terbalik mulai dari base case

Contoh 1

Fungsi faktorial

- Base case: n = 0
- Recursion call: f(n) = n * f(n-1)

```
public class faktorial {

    public static void main(String[] args) {
        System.out.println(faktorialRekursif(5));
    }

    static int faktorialRekursif(int n) {
        if (n == 0) {
            return (1);
        } else {
            return (n * faktorialRekursif(n - 1));
        }
    }
}
```

Base case

Recursion call

Contoh 1 - Trace

faktorialRekursif(5) = 5 * faktorialRekursif(4)

n * faktorialRekursif(n-1)

= 5 * (4 * faktorialRekursif(3))
= 5 * (4 * (3 * faktorialRekursif(2)))
= 5 * (4 * (3 * (2 * faktorialRekursif(1))))
= 5 * (4 * (3 * (2 * (1 * faktorialRekursif(0)))))

Fase Ekspansi

= 5 * (4 * (3 * (2 * (1 * 1))))
= 5 * (4 * (3 * (2 * 1)))
= 5 * (4 * (3 * 2))
= 5 * (4 * 6)
= 5 * 24
= 120

Fase Substitusi

Contoh 2

- Misalnya kita ingin membuat fungsi rekursif untuk mengalikan integer m dan integer n menggunakan penjumlahan
- Kita perlu mengidentifikasi base case dan recursion call
  - **Base case**: jika n bernilai 1, jawabannya adalah m
  - **Recursion call**: m * n = m + m(n-1)

$$m * n \begin{cases} m, & n = 1 \\ m + m\,(n-1), & n>1 \end{cases}$$

Contoh 2 - Trace

```
public class perkalian {

    public static void main(String[] args) {
        int nilai1 = 5, nilai2 = 4;
        System.out.println(kali(nilai1, nilai2));
    }

    static int kali(int m, int n) {
        if (n == 1) {
            return m;
        } else {
            return m + kali(m, n - 1);
        }
    }
}
```

Fase Ekspansi

kali(5, 4) = 5 + kali(5, 3)
= 5 + (5 + kali(5, 2))
= 5 + (5 + (5+ kali(5, 1)))

Fase Substitusi

= 5 + (5 + (5 + 5))
= 5 + (5 + 10)
= 5 + 15
= 20

Fungsi Rekursif vs Fungsi Iteratif

Fungsi Rekursif VS Fungsi Iteratif

- Pengulangan dengan struktur seleksi (IF-ELSE) dan pemanggilan fungsi dirinya sendiri
- Pengulangan berhenti saat base case terpenuhi
- Pengulangan tanpa henti jika base case tidak pernah terpenuhi
- Membutuhkan lebih banyak memori dan kerja prosesor lebih tinggi karena memanggil banyak fungsi
- Terbaca lebih jelas, pemodelan lebih dekat dengan masalah, contoh: faktorial

- Pengulangan dengan struktur repetisi (FOR/WHILE)
- Pengulangan berhenti saat kondisi pengulangan bernilai FALSE
- Pengulangan tanpa henti jika kondisi pengulangan selalu benar
- Membutuhkan memori lebih kecil dan kerja prosesor lebih rendah karena proses pengulangan berada dalam satu fungsi
- Terbaca kurang jelas, model kurang dekat dengan masalah

Fungsi Rekursif VS Fungsi Iteratif

```
static int faktorialRekursif(int n) {
    if (n == 0) {
        return (1);
    } else {
        return (n * faktorialRekursif(n - 1));
    }
}
```

```
static int faktorialIteratif(int n) {
    int faktor = 1;
    for (int i = n; i >= 1; i--) {
        faktor = faktor * i;
    }
    return faktor;
}
```

Fungsi main

```
public static void main(String[] args) {
    System.out.println(faktorialRekursif(5));
    System.out.println(faktorialIteratif(5));
}
```

Kapan Menggunakan Rekursif?

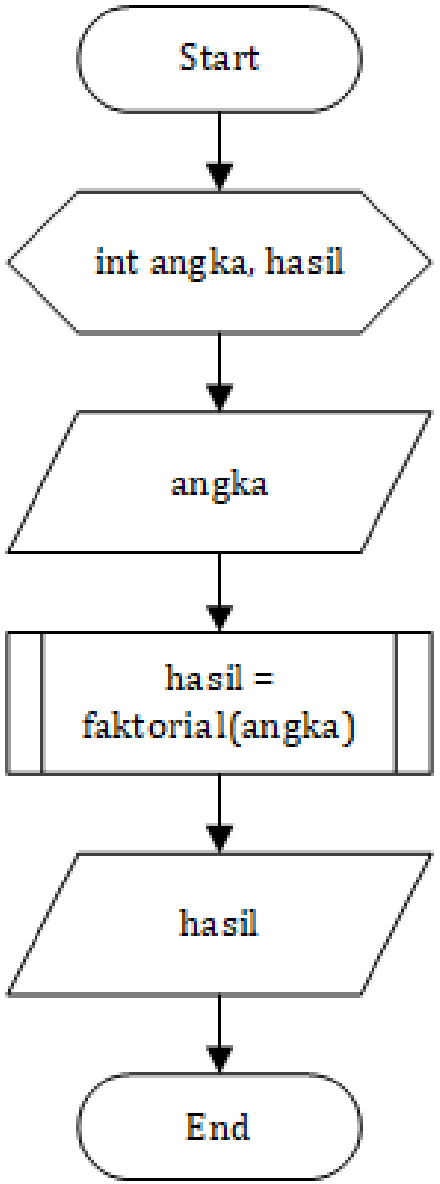
Ketika:

- Penyelesaian masalah sulit dikerjakan secara iteratif
- Tidak mempertimbangkan faktor penghematan memori dan kecepatan eksekusi program

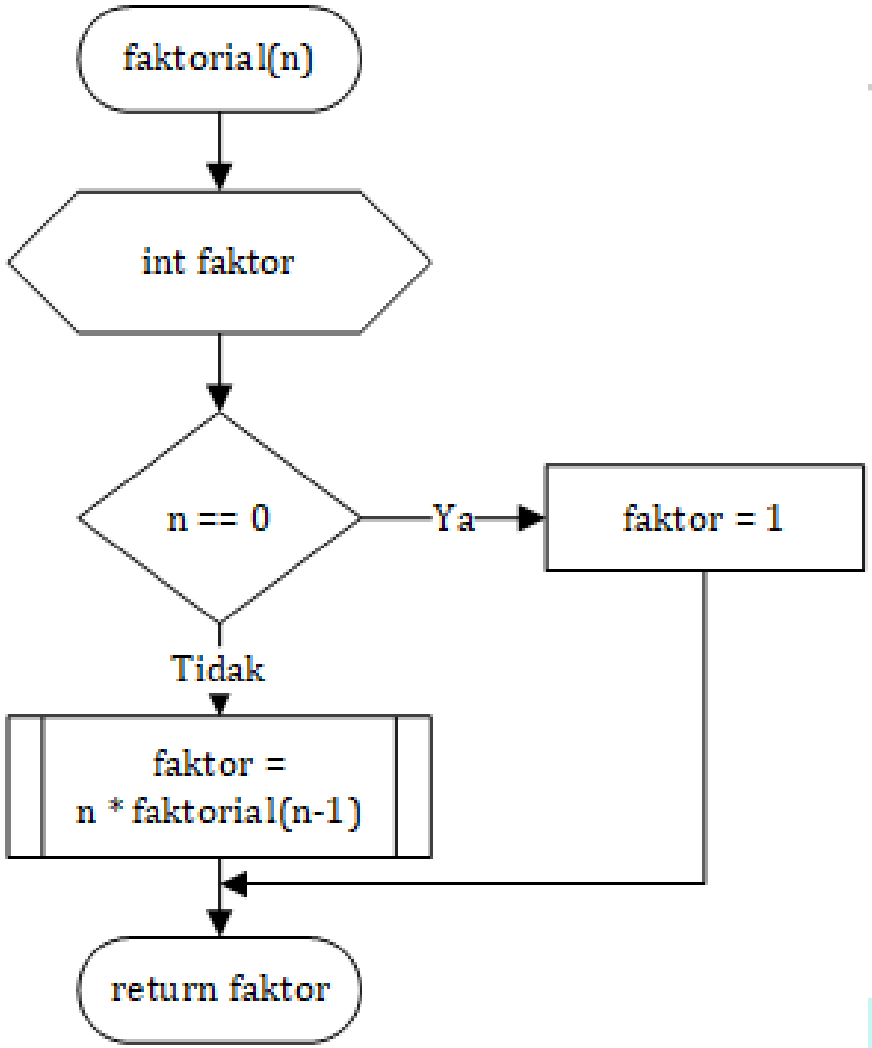
Contoh 1 - Jawaban

➢ BUATLAH FLOWCHART UNTUK MENGHITUNG NILAI FAKTORIAL DARI SEBUAH BILANGAN DENGAN MENGGUNAKAN FUNGSI REKURSIF!

Flowchart: main()



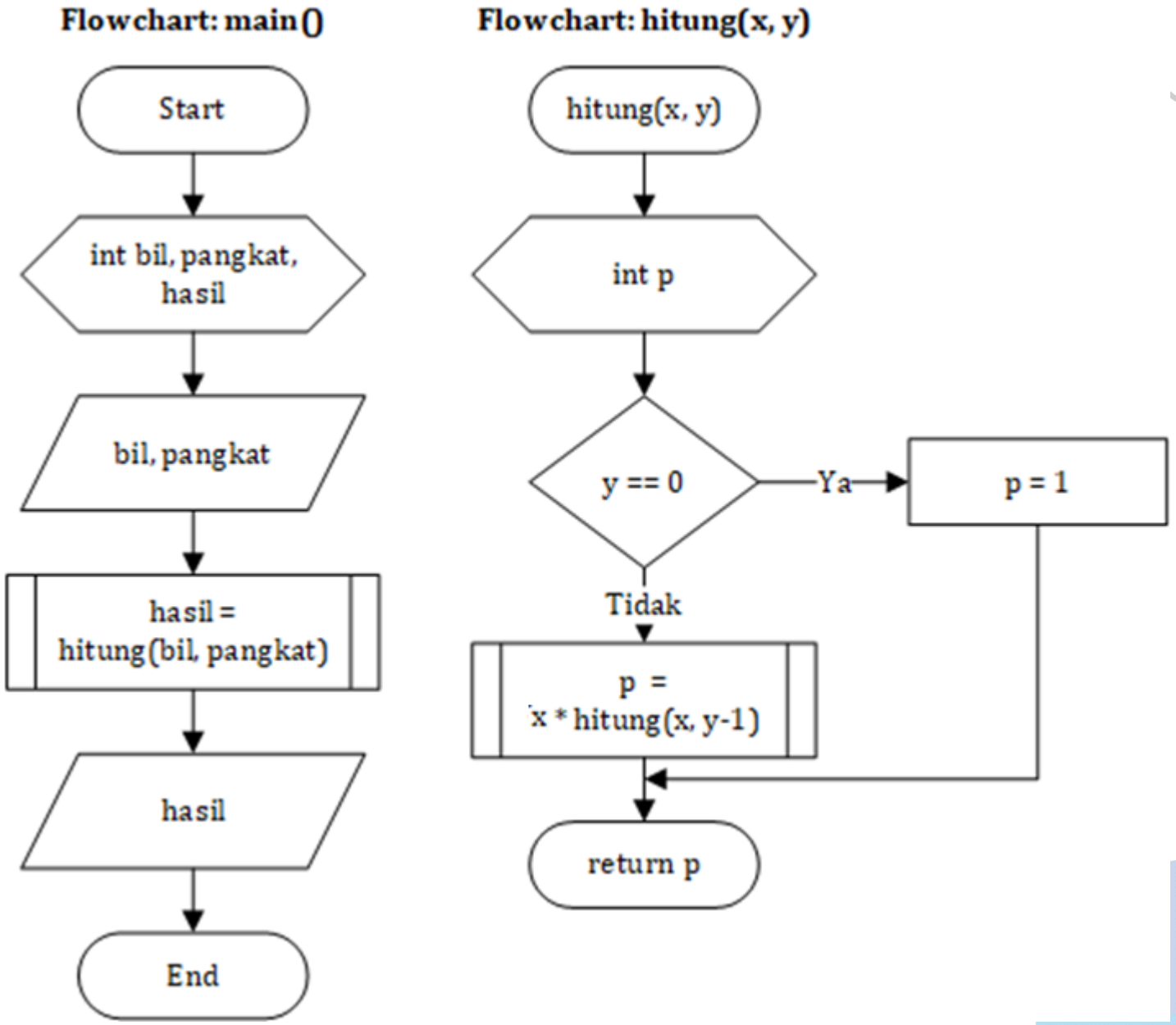
Flowchart: faktorial(n)



Contoh 2

- Terdapat sebuah program untuk menghitung nilai dari x pangkat y.
  Seperti yang kita ketahui, nilai dari X pangkat Y dihitung dengan cara X dikali X sebanyak (Y- 1) kali, tetapi jika Y adalah 0 (X pangkat 0) maka nilai X adalah 1.
- Sehingga untuk menghitung nilai X pangkat Y, program harus memberikan batasan bahwa jika Y = 0 maka nilai X menjadi 1.
- Buatlah flowchartnya!

Contoh 2 - Jawaban



Tugas Individu

1. Buatlah flowchart untuk menghitung dan mencetak total dengan input N

   1 + 2 + 3 + 4 + 5 + … + … + N

   dengan fungsi

   a) Secara iteratif

   b) Secara rekursif

2. Buatlah flowchart untuk menghitung pola deret Fibonanci

   Pola fibonanci : 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ….

   **menjumlahkan dua angka awal untuk mendapatkan angka-angka setelahnya*

3. Hitunglah hasil investasi seseorang pada pembelian emas batang.

   Keuntungan investasi emas yang didapatkan di setiap tahun nya 11,7%. Buatlah flowchart untuk menentukan banyaknya uang setelah beberapa (N) tahun, misalnya 10 tahun!