**JOBSHEET 11**

**Fungsi Rekursif**

1. **Tujuan**

* Mahasiswa memahami konsep fungsi rekursif
* Mahasiswa mampu mengimplementasikan fungsi rekursif dalam kode program

1. **Teori**

Fungsi rekursif adalah fungsi yang memanggil dirinya sendiri. Hal ini bisa terjadi karena di dalam suatu fungsi rekursi, terdapat statement/baris program yang memanggil fungsi itu sendiri. Ketika suatu fungsi rekursif dipanggil/dijalankan dan kemudian proses eksekusi sudah sampai pada statement pemanggilan fungsi itu sendiri, maka fungsi tersebut akan dipanggil/dijalankan kembali. Dan LAGI, ketika proses eksekusi sudah sampai pada statement pemanggilan fungsi itu sendiri, maka fungsi tersebut akan dipanggil/dijalankan kembali, BEGITU SETERUSNYA hingga didapatkan suatu KONDISI AKHIR dimana proses pemanggilan fungsi tersebut tidak dilakukan lagi. Jika kondisi akhir tersebut tidak ditemukan/tidak ada, maka fungsi tersebut akan dipanggil terus menerus (*infinite loop*) dan ini tidak diperbolehkan.

Pada fungsi rekursif, terdapat dua komponen blok kode, yaitu:

* **Base Case**: kode program yang menunjukkan sebuah batas berhenti dari proses rekursif, sehingga apabila nilai batas ini terpenuhi maka proses rekursif diakhiri.
* **Recursion Call** atau **Reduction Step**: kode program untuk melakukan pemanggilan terhadap dirinya sendiri.

Pada umumnya, format fungsi rekursif mempunyai bentuk sebagai berikut:

**if (nilai batas)**

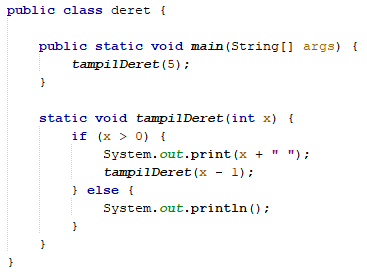
**//menyelesaikan masalah**

**else**

**//mendefinisikan kembali masalah menggunakan rekursi**

Cabang IF merupakan base case, sedangkan ELSE merupakan recursion call. Agar rekursi dapat berhenti, recursion call harus mendekati base case di setiap pemanggilan fungsi rekursif.

Berdasarkan penjelasan tersebut, sekilas terlihat seperti fungsi tersebut dijalankan secara berulang-ulang, dan fungsi rekursif memang biasanya digunakan pada kasus-kasus perulangan. Perhatikan fungsi **tampilDeret()** di bawah ini.



Memanggil fungsi **tampilDeret()** dengan nilai parameter 5

Memanggil fungsi **tampilDeret()** itu sendiri, dengan nilai parameter n-1

Pada contoh fungsi rekursif tersebut, fungsi tampilDeret pertema kali dijalankan (di dalam fungsi main) dengan mengirimkan nilai parameter 5 menggunakan perintah **tampilDeret(5)**. Dari proses pemanggilan fungsi tersebut akan ditampilkan nilai parameternya yaitu 5 dan diikuti dengan pemanggilan fungsi **tampilDeret(4).** Selanjutnya,dari proses pemanggilan fungsi tersebut akan ditampilkan nilai parameternya yaitu 4 dan diikuti dengan pemanggilan fungsi **tampilDeret(3).** Selanjutnya,dari proses pemanggilan fungsi tersebut akan ditampilkan nilai parameternya yaitu 3 dan diikuti dengan pemanggilan fungsi **tampilDeret(2).** Selanjutnya,dari proses pemanggilan fungsi tersebut akan ditampilkan nilai parameternya yaitu 2 dan diikuti dengan pemanggilan fungsi **tampilDeret(1).** Dari proses pemanggilan fungsi tersebut akan ditampilkan nilai parameternya yaitu 1 dan diikuti dengan pemanggilan fungsi **tampilDeret(0).** Dari proses pemanggilan fungsi tersebut maka akan ditampilkan ganti baris dan tidak ada pemanggilan fungsi tampilDeret selanjutnya (pemanggilan fungsi rekursif sudah selesai).

tampilDeret(5)

tampilDeret(4)

tampilDeret(3)

tampilDeret(2)

tampilDeret(1)

tampilDeret(0)

5

4

3

2

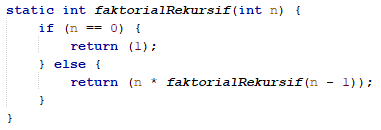
1

Tampilan yang muncul dari hasil pemanggilan fungsi tampiDeret adalah 5 4 3 2 1.

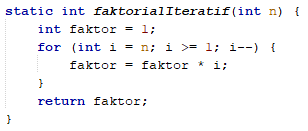
1. **Praktikum**
   1. **Percobaan 1**

Pada percobaan ini akan dilakukan pembuatan program untuk menghitung nilai faktorial dari suatu bilangan dengan menggunakan fungsi **rekursif**. Selain itu, akan dibuat juga fungsi untuk menghitung nilai faktorial dengan menggunakan algoritma **iteratif** sebagai pembandingnya.

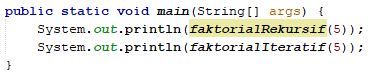
1. Jalankan NetBeans
2. Buat project baru bernama **Rekursif**, dan buat file Java dengan nama **Percobaan1**
3. Buat fungsi static dengan nama **faktorialRekursif()**, dengan tipe data kembalian fungsi **int** dan memiliki 1 parameter dengan tipe data **int** berupa bilangan yang akan dihitung nilai faktorialnya.



1. Buat lagi fungsi static dengan nama **faktorialIteratif()**, dengan tipe data kembalian fungsi **int** dan memiliki 1 parameter dengan tipe data **int** berupa bilangan yang akan dihitung nilai faktorialnya.



1. Buatlah fungsi main dan lakukan pemanggilan terhadap kedua fungsi yang telah dibuat sebelumnya, dan tampilkan hasil yang didapatkan.



1. Jalankan program tersebut. Amati apa yang terjadi!
2. Jika ditelusuri, pada saat pemanggilan fungsi **faktorialRekursif(5)**,maka proses yang terjadi dapat diilustrasikan sebagai berikut:

5 \* hitungFaktorialRekursif(4)

4 \* hitungFaktorialRekursif(3)

3 \* hitungFaktorialRekursif(2)

2 \* hitungFaktorialRekursif(1)

return 1

return 2 \* 1 = 2

return 3 \* 2 = 6

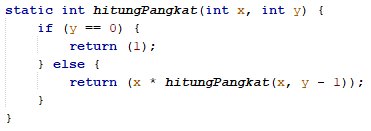
return 4 \* 6 = 24

return 5 \* 24 = 120

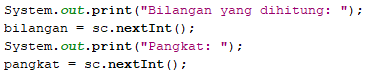
* 1. **Percobaan 2**

Pada percobaan ini akan dilakukan pembuatan program untuk menghitung pangkat sebuah bilangan dengan menggunakan fungsi rekursif.

1. Jalankan NetBeans
2. Pada project **Rekursif**, dan buat file Java dengan nama **Percobaan2**
3. Buat fungsi static dengan nama **hitungPangkat()**, dengan tipe data kembalian fungsi **int** dan memiliki 2 parameter dengan tipe data **int** berupa bilangan yang akan dihitung pangkatnya dan bilangan pangkatnya.



1. Buatlah fungsi main dan deklarasikan Scanner dengan nama **sc**
2. Buatlah dua buah variabel bertipa int dengan nama **bilangan** dan **pangkat**
3. Tambahkan kode berikut ini untuk menerima input dari keyboard



1. Lakukan pemanggilan fungsi hitungPangkat yang telah dibuat sebelumnya dengan mengirimkan dua nilai parameter.

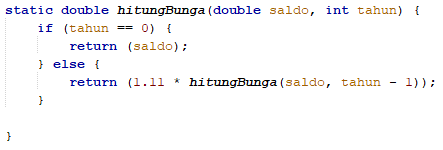


1. Jalankan program tersebut. Amati apa yang terjadi!
   1. **Percobaan 3**

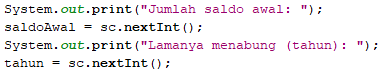
Pada percobaan ini akan dilakukan pembuatan program untuk menghitung jumlah uang nasabah yang disimpan di Bank setelah mendapatkan bunga selama beberapa tahun dengan menggunakan fungsi rekursif.

1. Jalankan NetBeans
2. Pada project **Rekursif**, dan buat file Java dengan nama **Percobaan3**
3. Buat fungsi static dengan nama **hitungBunga()**, dengan tipe data kembalian fungsi **double** dan memiliki 2 parameter dengan tipe data **int** berupa saldo nasabah dan lamanya menabung.

Pada kasus ini dianggap bunga yang ditentukan oleh bank adalah 11% per tahun. Karena perhitungan bunga adalah **bunga \* saldo**, sehingga untuk menghitung besarnya uang setelah ditambah bunga adalah **saldo + bunga \* saldo**. Dalam hal ini, besarnya bunga adalah 0.11 \* saldo, dan saldo dianggap 1 \* saldo, sehingga 1 \* saldo + 0.11 \* saldo dapat diringkas menjadi **1.11 \* saldo** untuk perhitungan saldo setelah ditambah bunga (dalam setahun).



1. Buatlah fungsi main dan deklarasikan Scanner dengan nama **sc**
2. Buatlah sebuah variabel bertipa double dengan nama **saldoAwal** dan sebuah variabel bertipe int bernama **tahun**
3. Tambahkan kode berikut ini untuk menerima input dari keyboard



1. Lakukan pemanggilan fungsi hitungBunga yang telah dibuat sebelumnya dengan mengirimkan dua nilai parameter.



1. Jalankan program tersebut. Amati apa yang terjadi!
2. **Pertanyaan**
3. Apa yang dimaksud dengan fungsi rekursif?
4. Pada kasus-kasus seperti apa fungsi rekursif digunakan?
5. Pada **Percobaan1**, apakah hasil yang diberikan fungsi **faktorialRekursif()** dan fungsi **faktorialIteratif()** sama? Jelaskan perbedaan alur jalannya program pada penggunaan fungsi rekursif dan fungsi iteratif!
6. Pada **Percobaan2**, terdapat pemanggilan fungsi rekursif **hitungPangkat(bilangan, pangkat)** pada fungsi main, kemudian dilakukan pemanggilan fungsi **hitungPangkat()** secara berulangkali.Jelaskan sampai kapan proses pemanggilan fungsi tersebut akan dijalankan!
7. Pada **Percobaan2**, jika pada saat pemanggilan fungsi **faktorialRekursif** nilai parameter yang dilewatkan diganti dari **n-1** menjadi **n** saja, sehingga sintaks program menjadi:

return n \* faktorialRekursif(n);

Apa yang terjadi? Mengapa demikian?

1. Pada **Percobaan3**, sebutkan blok kode program manakah yang merupakan “base case” dan “recursion call”!
2. Pada **Percobaan3**, fungsi hitungBunga, terdapat kondisi IF (tahun == 0), ganti kode tersebut menjadi IF (tahun < 1). Apa yang terjadi? Jelaskan mengapa demikian!
3. **Tugas**
4. (**DeretDescendingRekursif**). Buatlah program untuk menampilkan bilangan n sampai 0 dengan menggunakan fungsi rekursif dan fungsi iteratif.
5. (**PenjumlahanRekursif**). Buatlah program yang di dalamnya terdapat fungsi rekursif untuk menghitung hasil penjumlahan dari n bilangan. Misalniya n = 8, maka akan dihasilkan 1+2+3+4+5+6+7+8 = **36**
6. (**CekPrimaRekursif**). Buat program yang di dalamnya terdapat fungsi rekursif untuk mengecek apakah suatu bilangan n merupakan bilangan prima atau bukan. n dikatakan bukan bilangan prima jika ia habis dibagi dengan bilangan kurang dari n.
7. (**FPBEuclidRekursif**). Buat fungsi rekursif untuk mencari nilai FPB (Faktor Persekutuan Terbesar) dari dua bilangan, dengan menggunakan algoritma Euclid. Contoh mencari nilai FPB dengan Euclid adalah sebagai berikut:

Misal: cari FPB dari bilangan 45 dan 20.

m=45, n=20 🡪 sisa = m%n = 45%20 = 5

m=20, n=5 🡪 sisa = m%n = 20%5 = 0

Karena sisa sudah bernilai 0, maka bilangan FPB = n = 5

Cari FPB dari bilangan 24 dan 30

m=24, n=30 🡪 sisa = m%n = 24%30 = 24

m=30, n=24 🡪 sisa = m%n = 30%24 = 6

m=24, n=6 🡪 sisa = m%n = 24%6 = 0

Karena sisa sudah bernilai 0, maka bilangan FPB = n = 6

1. (**Fibonacci**). Sepasang kelinci yang baru lahir (jantan dan betina) ditempatkan pada suatu pembiakan. Setelah dua bulan pasangan kelinci tersebut melahirkan sepasang kelinci kembar (jantan dan betina). Setiap pasangan kelinci yang lahir juga akan melahirkan sepasang kelinci juga setiap 2 bulan. Berapa pasangan kelinci yang ada pada akhir bulan ke-12? Buatlah programnya menggunakan fungsi rekursif!

Berikut ini adalah ilustrasinya dalam bentuk tabel.

|  |  |  |  |
| --- | --- | --- | --- |
| **Bulan ke-** | **Jumlah Pasangan** | | **Total Pasangan** |
| **Produktif** | **Belum Produktif** |
| 1 | 0 | 1 | 1 |
| 2 | 0 | 1 | 1 |
| 3 | 1 | 1 | 2 |
| 4 | 1 | 2 | 3 |
| 5 | 2 | 3 | 5 |
| 6 | 3 | 5 | 8 |
| 7 | 5 | 8 | 13 |
| 8 | 8 | 13 | 21 |
| 9 | 13 | 21 | 34 |
| 10 | 21 | 34 | 55 |
| 11 | 34 | 55 | 89 |
| 12 | 55 | 89 | 144 |