

HARIKE (versi-0)	NH(d,m,y)
<u>DEFINISI DAN SPESIFIKASI TYPE</u> <u>type Hr</u> : <u>integer</u> [1...31] <i>{definisi ini hanyalah untuk memberi nama baru terhadap type integer dengan nilai tertentu supaya mewakili hari, sehingga jika dipunyai suatu nilai integere, kita dapat memeriksa apakah nilai integer tersebut mewakili Hari yang absah }</i> <u>type Bln</u> : <u>integer</u> [1...12] <i>{definisi ini hanyalah untuk memberi nama baru terhadap type integer dengan nilai tertentu supaya mewakili Bulan }</i> <u>type Thn</u> : <u>integer</u> > 0 <i>{definisi ini hanyalah untuk untuk memberi nama baru terhadap type integer dengan nilai tertentu supaya mewakili tahun }</i> <u>type date</u> <d: Hr, m:Bln,y:Thn> <i>{ <d,m,y> adalah tanggal d bulan m tahun y }</i> <u>type NHari</u> : <u>integer</u> [1..366] <i>{Jumlah hari dalam suatu tahun kalender }</i>	
<u>DEFINISI DAN SPESIFIKASI SELEKTOR DENGAN FUNGSI</u> Day : date → Hr <i>{Day (D) memberikan hari d dari D yang terdiri dari <d,m,y> }</i> Month : date → Bln <i>{Month (D) memberikan bulan m dari D yang terdiri dari <d,m,y> }</i> Year : date → Thn <i>{Year (D) memberikan tahun y dari D yang terdiri dari <d,m,y> }</i>	
<u>DEFINISI TABEL KOREKSI</u> Koreksi : <u>table of</u> Bln = [0,1,-1,0,0,1,1,2,3,3,4,4]	
<u>DEFINISI DAN SPESIFIKASI FUNGSI ANTARA</u> IsKabisat? : <u>integer</u> → <u>boolean</u> <i>{IsKabisat?(a) true jika tahun 1900+a adalah tahun kabisat: habis dibagi 4 tetapi tidak habis dibagi 100, atau habis dibagi 400 }</i>	
<u>DEFINISI DAN SPESIFIKASI FUNGSI NH</u> NH : date → NHari <i>{NH(<d,m,y> menghasilkan hari absolut pada tahun y; setiap bln bervariasi sesuai dengan jumlah hari dalam bulan ybs: Januari : 30 hari; Februari 28 atau 29 hari; maret : 31 hari,... Jadi realisasi versi-0 harus dikoreksi dengan tabel koreksi}</i>	
<u>REALISASI KOREKSI DENGAN TABEL</u> $\text{NH (D) : } 30 * (\text{Month (D)} - 1) + \text{Day (D)} + \text{Koreksi}_B +$ $\text{if } m > 2 \text{ and IsKabisat? Year (D) then } 1 \text{ else } 0$ IsKabisat?(a) : ((a mod 4 = 0) and (a mod 100 ≠ 0)) or (a div 400 = 0)	

Contoh 2:

Pada contoh berikut, realisasi koreksi akan dituliskan sebagai fungsi dan bukan sebagai tabel. Realisasinya menjadi suatu ekspresi kondisional yang pernah dibahas. Untuk kasus ini, karena dalam satu tahun “hanya” ada 12 bulan, dan untuk nilai koreksi hanya dibutuhkan satu nilai, maka hanya dibutuhkan sebuah nama dan sebuah analisa kasus terhadap 12 kemungkinan harga yang harus dihasilkan oleh fungsi.. **Solusi ini tidak memakai type berupa koleksi objek**, berbeda halnya dengan solusi dengan tabel.

HARIKE (versi-Fungsi)	NH(d,m,y)
<u>DEFINISI DAN SPESIFIKASI TYPE</u> type Hr : <u>integer</u> [1...31] <i>{definisi ini hanyalah untuk “menamakan kembali type integer dengan nilai tertentu supaya mewakili hari, sehingga jika dipunyai suatu nilai integer, kita dapat memeriksa apakah nilai integer tersebut mewakili Hari yang absah }</i> type Bln : <u>integer</u> [1...12] <i>{definisi ini hanyalah untuk “menamakan kembali type integer dengan nilai tertentu supaya mewakili Bulan }</i> type Thn : <u>integer</u> > 0 <i>{definisi ini hanyalah untuk “menamakan kembali type integer dengan nilai tertentu supaya mewakili tahun }</i> type date <d: Hr, m:Bln,y:Thn> <i>{ <d,m,y> adalah tanggal d bulan m tahun y }</i> type NHari : <u>integer</u> [1..366] <i>{Jumlah hari dalam suatu tahun kalender }</i>	
<u>DEFINISI DAN SPESIFIKASI SELEKTOR DENGAN FUNGSI</u> Day : date → Hr <i>{Day (D) memberikan hari d dari D yang terdiri dari <d,m,y> }</i> Month : date → Bln <i>{Month (D) memberikan bulan m dari D yang terdiri dari <d,m,y> }</i> Year : date → Thn <i>{Year (D) memberikan tahun y dari D yang terdiri dari <d,m,y> }</i>	
<u>DEFINISI DAN SPESIFIKASI FUNGSI NH</u> NH : date → NHari <i>{NH(<d,m,y> menghasilkan hari absolut pada tahun y; setiap bln bervariasi sesuai dengan jumlah hari dalam bulan ybs: Januari : 30 hari; Februari 28 atau 29 hari; maret : 31 hari,... }</i>	

DEFINISI DAN SPESIFIKASI FUNGSI ANTARA

IsKabisat? : integer → boolean

{IsKabisat?(a) true jika tahun 1900+a adalah tahun kabisat: habis dibagi 4 tetapi tidak habis dibagi 100, atau habis dibagi 400 }

Koreksi(B) : Bln → integer

{Koreksi (B) adalah banyaknya koreksi terhadap perhitungan hari s/d tanggal 1 bulan B, }

REALISASI

NH(D) : $30 * (\text{Month (D)} - 1) + \text{Day (D)} + \text{Koreksi}(\text{Month (D)}) +$
 $\text{if } m > 2 \text{ and IsKabisat? Year (D) then } 1 \text{ else } 0$

IsKabisat?(a) : $((a \bmod 4 = 0) \text{ and } (a \bmod 100 \neq 0)) \text{ or } (a \text{ div } 400 = 0)$

REALISASI KOREKSI DENGAN FUNGSI

{ Pada solusi ini, fungsi Koreksi(B) menggantikan tabel Koreksi pada solusi sebelumnya }

Koreksi(B) : depend on B

B = 1 : 0

B = 2 : 1

B = 3 : -1

B = 4 : 0

B = 5 : 0

B = 6 : 1

B = 7 : 1

B = 8 : 1

B = 9 : 2

B = 10 : 3

B = 11 : 3

B = 12 : 4

EKSPRESI REKURSIF

Definisi entitas (type, fungsi) disebut rekursif jika definisi tersebut mengandung terminologi dirinya sendiri.

Ekspresi rekursif dalam pemrograman fungsional didasari oleh Analisa rekurens, yaitu penalaran berdasarkan definisi **fungsi rekursif**, yang biasanya juga berdasarkan “type” yang juga terdefinisi secara rekursif-induktif.

Bandingkanlah cara induksi ini dengan pendekatan eksperimental klasik, yang mencoba-coba menurunkan kebenaran dari observasi, atau dapat juga dengan cara berusaha menemukan *counter-example* (contoh yang salah, yang menunjukkan kebalikannya).

Contoh cara pembuktian kebenaran dengan menemukan contoh yang salah :

$$f(n) = n^2 - n + 41$$

$$f(0) = 41 \quad f(1) = 41 \quad f(2) = 43$$

$$f(3) = 47 \quad f(4) = 53$$

$f(n)$ adalah fungsi untuk menghasilkan bilangan prima : benar untuk $n = 40$, tetapi untuk $n = 41$ salah, karena $41^2 - 41 + 41 = 1681$ bukan bilangan prima.

Metoda pembuktian rekurens : metoda dimana sifat (*property*) dibuktikan benar untuk satu elemen (sebagai basis), kemudian benar untuk semua elemen

Bukti secara rekurens

Bukti rekurens adalah Cara untuk membuktikan bahwa *property* (suatu sifat) adalah benar untuk semua elemen:

- dengan basis benar jika $n=0$;
- kemudian untuk n dianggap benar, kita membuktikan untuk $n+1$ benar sehingga dapat menyimpulkan bahwa untuk semua n benar.

a. **Bukti rekurens terhadap bilangan integer.**

- Basis : property untuk $n = 0$

- Rekurens : benar untuk n
benar untuk $n+1$

Contoh -pembuktian-1

Buktikan bahwa $10^n - 1$ habis dibagi 9.

Bukti : A (bilangan natural) habis dibagi 9 jika ada n sehingga $A=n*9$

Basis : untuk $n= 0$ benar, sebab $10^n - 1 = 0$ habis dibagi 9

Rekurens : untuk n sembarang, dianggap bahwa $10^n - 1$ habis dibagi 9.

$$10^{n+1} - 1 = 10 * (10^n - 1) + 9$$

$$10 (10^n) - 1 = 9 * (10 * 9)$$

$$10 * (10^n - 1) + 10 - 1$$

Contoh-pembuktian -2

Buktikan bahwa semua bilangan bulat ≥ 2 dapat diuraikan dalam faktor primer, yaitu dapat diekspresikan sebagai proses hasil kali dari faktor primer.

Basis : benar untuk $n=2$ dengan analisa kasus :

Rekurens:

- jika n bilangan prima, maka n dapat diuraikan dalam faktor primer (benar) sebab semua bilangan prima dapat difaktorisasi menjadi bilangan prima
- jika n bukan bilangan prima :
Misalnya k dapat ditulis dalam faktor bilangan prima.
 $k+1$ juga dapat ditulis dalam faktor bilangan prima, $k+1 = d$
ada $2 \leq c, d \leq k$. Karena $2, 3, \dots k$ mempunyai pembagi prima,
maka c dan d dapat ditulis dengan faktor bilangan prima

Contoh- Pembuktian -3

Buktikan bahwa banyaknya himpunan bagian dari suatu himpunan dengan kardinalitas n adalah 2^n

Basis : untuk himpunan kosong : $2^0 = 1$

Sebuah himpunan adalah merupakan himpunan bagian dari diri sendiri.

Himpunan bagian dari suatu himpunan dengan kardinalitas n adalah 2^n .

E adalah himpunan dengan kardinalitas $n+1$

Isolasi sebuah elemen a dari E , dan himpunan selain a disebut F .

Maka

$E - P$ himpunan. bagian (hanya mengandung a)

P^1 himpunan bagian (tanpa a)

P dan P^1 adalah *disjoint*

Misalnya kardinalitas P adalah n

$P^1 : 2^n$

Kardinalitas P sama dengan P^1 dengan menambahkan elemen a menjadi elemen P^1 adalah $2 * 2^n = 2^{n+1}$.

Type Rekursif

- Jika teks yang mendefinisikan type mengandung referensi terhadap diri sendiri, maka type disebut type rekursif.
- Type dibentuk dengan komponen yang merupakan type itu sendiri

Perhatikanlah beberapa contoh definisi type sebagai berikut:

a. Bilangan integer ganjil

Basis : 1 adalah bilangan integer ganjil


Rekurens : if x adalah bilangan integer ganjil

then $x + 2$ adalah bilangan integer ganjil

b. Luas bujur sangkar

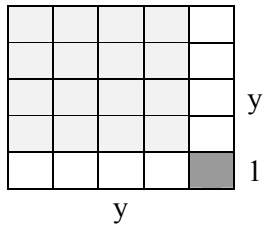
Basis : luas bujur sangkar dengan sisi 1 = 1

Rekurens :

Jika C adalah bujur sangkar dengan sisi y, 

bujur sangkar dengan sisi y+1 didapat dengan menambahkan 1 pada setiap sisinya
= y+1. Luas bujur sangkar menjadi :

$$(y+1)^2 = y^2 + 2Y + 1$$

**c. List/sequence :**

Basis : list Kosong : [] list tanpa elemen adalah sebuah list

Rekurens : list tidak kosong dibentuk dengan konstruktor

List **dibentuk** dengan menambahkan sebuah elemen menjadi anggota list

Konso(S,e) : S o e tambah sebuah element di 'kanan'

Konso•(e,S) : e•S tambah sebuah element di 'kiri'

d. Bilangan natural :

Basis : 0 adalah bilangan natural

Rekurens :

Jika x adalah bilangan natural,

Bilangan natural yang lain dibentuk dengan menambahkan 1 pada x, succ(x)

atau dengan mengalikan suatu bilangan natural dengan 2: $x \rightarrow 2x$,

$d(x,0) \rightarrow$ untuk integer genap ($2n$)

$d(x,1) \rightarrow$ untuk integer ganjil ($2n+1$)

Fungsi Rekursif

Dalam ekspresi fungsional: realisasi fungsi rekursif. Ada dua kategori fungsi rekursif, yaitu rekursif langsung atau tidak langsung.

Fungsi Rekursif langsung

Fungsi didefinisikan rekursif, jika **ekspresi** yang merealisasi fungsi tersebut mengandung **aplikasi** terhadap fungsi tersebut. :

REALISASI

```
F (<list-param>) :  
  depend on  
    <kondisi-basis >      : <ekspresi-1 >  
    <kondisi-rekurens > : F (<ekspresi-2 >)
```

Dengan catatan, bahwa ekspresi-2 biasanya dinyatakan dengan domain yang sama dengan <list-param>, namun “mendekati” kondisi basis sehingga suatu saat akan terjadi kondisi basis yang menyebabkan aplikasi berhenti.

Fungsi rekursif tidak langsung

Realisasi fungsi mungkin *cross-recursive*, yaitu jika realisasi fungsi F mengandung aplikasi fungsi G yang realisasinya adalah aplikasi terhadap F.

REALISASI

```
G (<list-param>): F (<ekspresi-1 >)  
  
F (<list-param>):  
  depend on  
    <kondisi-basis > : <ekspresi-1 >  
    <kondisi-rekurens > : G (<ekspresi-2 >)
```

Dalam menuliskan suatu fungsi rekursif, pemrogram harus membaca ulang teksnya, dan dapat “membuktikan” bahwa suatu saat program akan “berhenti”, karena mempunyai basis. Pembuktian secara matematis diluar cakupan diktat kuliah ini

Berikut ini akan diberikan contoh fungsi rekursif sederhana dan aplikasinya. Disebut “sederhana”, karena program rekursif benar-benar dibangun dari definisi rekursif dari persoalan yang akan dikomputasi, seperti definisi Faktorial dan Fibonacci.

Contoh-1 Ekspresi rekursif : FACTORIAL

Persoalan :

Tuliskanlah sebuah fungsi yang menghitung factorial dari n sesuai dengan **definisi rekursif** faktorial.

Faktorial	fac(n)
<u>DEFINISI DAN SPESIFIKASI</u>	
$\text{fac} : \text{integer} \geq 0 \rightarrow \text{integer} > 0$ $\{ \text{fac}(n) = n! \text{ sesuai dengan definisi rekursif factorial} \}$	
<u>REALISASI (VERSI-1)</u>	
$\{ \text{Realisasi dengan definisi factorial sebagai berikut jika fac(n) adalah n!} :$ $n = 0 : n! = 1$ $n \geq 1 : n! = (n-1)! * n \}$ $\text{fac}(n) : \text{if } n = 1 \text{ then } \{ \text{Basis 1} \}$ $\quad 1$ $\quad \text{else } \{ \text{Rekurens : definisi faktorial} \}$ $\quad \text{fac}(n-1) * n$	
<u>REALISASI (VERSI-2)</u>	
$\{ \text{Realisasi dengan definisi factorial sebagai berikut jika fac(n) adalah n!} :$ $n = 0 : n! = 1$ $n \geq 1 : n! = (n-1)! * n \}$ $\text{fac}(n) : \text{if } n = 0 \text{ then } \{ \text{Basis 0} \}$ $\quad 1$ $\quad \text{else } \{ \text{Rekurens : definisi faktorial} \}$ $\quad n * \text{fac}(n-1)$	
<u>REALISASI (VERSI-3): RUMUS BENAR, PROGRAM YANG SALAH !</u>	
$\{ \text{Realisasi dengan definisi factorial sebagai berikut jika fac(n) adalah n!} :$ $n = 1 : n! = 1$ $n > 1 : n! = (n+1)! / n \}$ $\{ \text{Realisasi berikut yang didasari definisi di atas tidak benar sebab evaluasi untuk fac(n), } n > 1 \text{ menimbulkan pemanggilan yang tidak pernah berhenti} \}$ $\text{fac}(n) : \text{if } (n = 1) \text{ then } \{ \text{Basis 1} \}$ $\quad 1$ $\quad \text{else } \{ \text{Rekurens : definisi faktorial} \}$ $\quad \text{fac}(n+1) / (n)$	

Contoh-3 Ekspresi rekursif : FIBONACCI

Persoalan :

Tuliskanlah sebuah fungsi yang menghitung Fibonacci dari n , dengan **definisi rekursif** fungsi Fibonacci.:

Fibonacci	Fib(n)
<u>DEFINISI DAN SPESIFIKASI</u>	
Fib : $\text{integer} \geq 0 \rightarrow \text{integer} \geq 0$ { Definisi rekursif fungsi fibonacci : } { Fib (n) = sesuai dengan definisi deret fibonacci : $n = 0 : \text{Fib}(0) = 0$ $n = 1 : \text{Fib}(1) = 1$ $n > 1 : \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2)$ }	
<u>REALISASI</u>	
Fib (n) : <u>depend on</u> (n) $n = 0 : 0$ {Basis 0} $n = 1 : 1$ {Basis 1} $n > 1 : \text{Fib}(n-1) + \text{Fib}(n-2)$ {Rekurens}	

Perhatikanlah bahwa “basis” dari fungsi ini sesuai dengan dua buah aturan, yaitu untuk $n=0$ dan $n=1$, karena rekurens dimulai dari $n=2$

Ekspresi Rekursif terhadap Bilangan Bulat

Pada bagian ini akan diberikan contoh, bagaimana menuliskan ekspresi rekursif yang dibangun berdasarkan analisa rekurens, terhadap bilangan integer, untuk merealisasi penjumlahan, perkalian dan pemangkatan bilangan integer.

Contoh yang dibahas dalam sub bab ini hanyalah memberikan pola berpikir rekursif terhadap type sederhana (integer). Dalam kenyataannya, memang pemrogram tidak menuliskan lagi fungsi rekursif untuk penjumlahan, pengurangan, pembagian karena sudah dianggap operator dasar.

Untuk menulis ekspresi rekursif dengan fungsi rekursif untuk bilangan integer, bilangan integer harus didefinisikan secara rekursif sebagai berikut:

- Basis : Nol adalah bilangan integer
- Rekurens :
 - Jika n adalah bilangan integer, maka suksesor dari n adalah bilangan integer
 - Jika n adalah bilangan integer, maka predesesor dari n adalah bilangan integer

TYPE INTEGER

DEFINISI KONSTRUKTOR

{ Konstruktor integer > 0 }

succ : integer $\geq 0 \rightarrow$ integer > 0

{ Basis: $n = 0 \rightarrow 0$

Rekurens : $n \rightarrow n + 1$

succ(n) membentuk deret bilangan integer positif}

{ Konstruktor integer < 0 }

prec : integer \rightarrow integer

{ Basis: $n =$ representasi maksimal atau minimal mesin . Untuk n bilangan positif, basisnya seringkali diambil 0}

Rekurens : $n \rightarrow n - 1$

prec(n) membentuk deret bilangan integer negatif}

REALISASI

succ (n) : $n + 1$

prec (n) : $n - 1$

Berikut ini akan diberikan beberapa contoh fungsi rekursif berdasarkan definisi rekursif bilangan integer tersebut, dengan “memperkecil” menuju basisnya

Contoh-1 Penjumlahan bilangan bulat dengan ekspresi rekursif

Persoalan :

Tuliskanlah sebuah fungsi yang menjumlahkan dua buah integer, dan menghasilkan sebuah integer, dengan membuat definisi rekursif dari penjumlahan

PENJUMLAHAN dua bilangan integer	Plus(x,y)
<u>DEFINISI DAN SPESIFIKASI</u>	
Plus : 2 <u>integer</u> > 0 → <u>integer</u> > 0 <i>{ Plus(x,y) menghasilkan x + y }</i>	
<u>REALISASI VERSI-1 : REKURENS TERHADAP Y</u>	
$\{ \text{Plus}(x,y) = x + y = x + 1 + 1 + \dots + 1$ $= x + 1 + (y-1) = 1 + x + (y-1) \}$ $\{ \text{Basis} : y = 0 \rightarrow x$ $\text{Rekurens} : y > 0 \rightarrow 1 + \text{Plus}(x, \text{prec}(y)) \}$ Plus (x,y) : <u>if</u> y = 0 <u>then</u> {Basis 0} x <u>else</u> {Rekurens terhadap y } 1 + Plus(x,prec(y))	
<u>REALISASI VERSI-2 : REKURENS TERHADAP X</u>	
$\{ \text{Plus}(x,y) = x + y = 1 + 1 + \dots + 1 + y$ $= 1 + (x-1) + y \}$ $\{ \text{Basis} : x = 0 \rightarrow y$ $\text{Rekurens} : x > 0 \rightarrow 1 + \text{Plus}(\text{prec}(x), y) \}$ Plus (x,y) : <u>if</u> x = 0 <u>then</u> {Basis 0} y <u>else</u> {Rekurens terhadap x} Plus(prec(x), y)	

Contoh-2 Perkalian bilangan bulat dengan ekspresi rekursif

Persoalan :

Tuliskanlah definisi, spesifikasi dan realisasi sebuah fungsi yang mengalikan dua buah integer, dan menghasilkan sebuah integer, dengan membuat definisi rekursif dari perkalian.

PERKALIAN dua bilangan integer	Mul(x,y)
<u>DEFINISI DAN SPESIFIKASI</u>	
Mul : 2 <u>integer</u> $>0 \rightarrow$ <u>integer</u> >0 $\{ \text{Mul}(x,y) \text{ menghasilkan } x * y \}$	
<u>REALISASI VERSI-1 : REKURENS TERHADAP Y</u>	
$\{ \text{Mul}(x,y) = x * y$ $= x + x + x + x \dots\dots + x$ $= x + x * (y-1) \}$ $\{ \text{Basis} : y = 0 \rightarrow 0$ $\text{Rekurens} : y > 0 \rightarrow x + \text{Mul}(x, \text{prec}(y)) \}$ $\text{Mul}(x,y) : \begin{array}{l} \text{if } y = 0 \text{ then } \{ \text{Basis } 0 \} \\ 0 \\ \text{else } \{ \text{Rekurens terhadap } y \} \\ x + \text{Mul}(x, \text{prec}(y)) \end{array}$	
<u>REALISASI VERSI-1 : REKURENS TERHADAP X</u>	
$\{ \text{Mul}(x,y) = x * y$ $= y + y + y + y \dots\dots + y$ $= y + y * (x-1) \}$ $\{ \text{Basis} : x = 0 \rightarrow 0$ $\text{Rekurens} : x > 0 \rightarrow y + \text{Mul}(\text{prec}(x), y) \}$ $\text{Mul}(x,y) : \begin{array}{l} \text{if } x = 0 \text{ then } \{ \text{Basis } 0 \} \\ 0 \\ \text{else } \{ \text{Rekurens terhadap } x \} \\ y + \text{Mul}(\text{prec}(x), y) \end{array}$	

Contoh-2 Pemangkatan bilangan bulat dengan ekspresi rekursif

Persoalan :

Tuliskanlah definisi, spesifikasi dan realisasi sebuah fungsi yang memangkatkan sebuah integer dengan sebuah integer, dan menghasilkan sebuah integer, dengan membuat definisi rekursif dari pemangkatan.

PEMANGKATAN dua bilangan integer	Exp(x,y)
<u>DEFINISI DAN SPESIFIKASI</u>	
Exp : 2 <u>integer</u> > 0 → <u>integer</u> >0 <i>{ Exp (x,y) menghasilkan x^y, $x \neq 0$ }</i>	
<u>REALISASI</u> <i>{ Exp (x,y) = x^y = $x * x * x * x * \dots * x$ = $x * x^{(y-1)}$ }</i> <i>{ Basis : $y = 0 \rightarrow 1$ Rekurens : $y > 0 \rightarrow x * \text{Exp}(x, \text{prec}(y))$ }</i> Exp (x,y) : <u>if</u> y = 0 <u>then</u> {Basis 0} 1 <u>else</u> {Rekurens terhadap y} x * Exp (x,prec (y))	