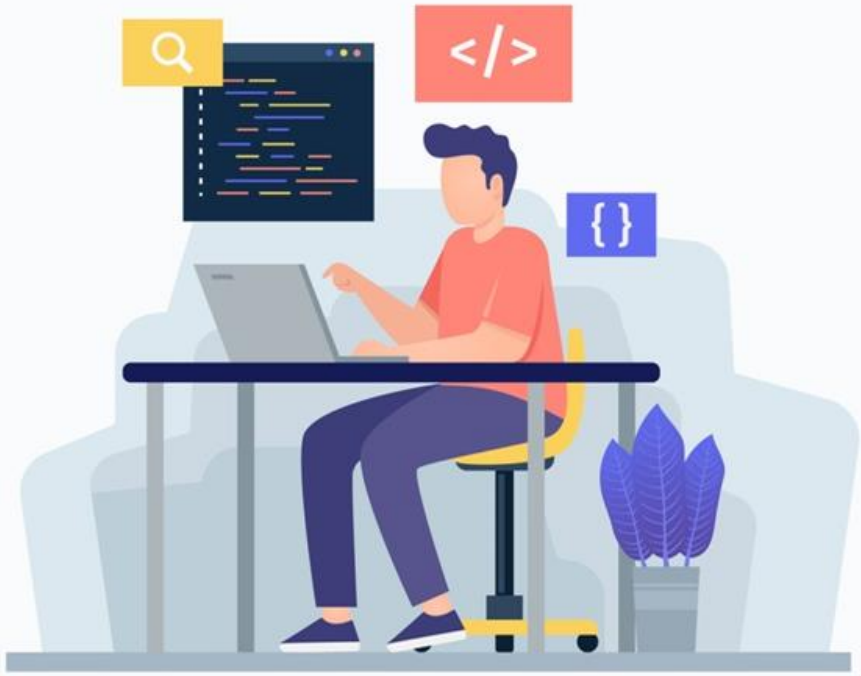
PROGRAM STUDI
TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

Mata Kuliah
Dasar Pemrograman



Analisis Kasus – Banyak Kasus

TIM DASAR PEMROGRAMAN
TEKNIK INFORMATIKA S1
UNIVERSITAS DIAN NUSWANTORO

Capaian Pembelajaran

- Setelah mengikuti kuliah ini mahasiswa dapat menjelaskan dan mempraktekkan analisis satu kasus dengan kondisi diterapkan pada aksi sekuensial permasalahan komputasional sederhana
- Setelah mengikuti kuliah ini mahasiswa dapat menjelaskan dan mempraktekkan analisis dua kasus komplemen dengan kondisi diterapkan pada aksi sekuensial permasalahan komputasional sederhana
- Setelah mengikuti kuliah ini mahasiswa dapat menjelaskan dan mempraktekkan analisis kasus bersarang diterapkan pada aksi sekuensial permasalahan komputasional sederhana

FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

IF – ELSE IF - ELSE

Analisis Banyak Kondisi

- Pengecekan terhadap banyak kondisi dengan memilih variasi aksi berdasarkan pasangan kondisi yang sesuai
- Berikut bentuk notasi umum:

depend on <list-nama>

<kondisi ke-1>: <aksi ke-1>

<kondisi ke-2>: <aksi ke-2>

…

<kondisi ke-N>: <aksi ke-N>

If dan Else if

- Baik if maupun else if boleh menggunakan ekspresi Boolean majemuk.
- Pada else jelas tidak ada ekspresi Boolean

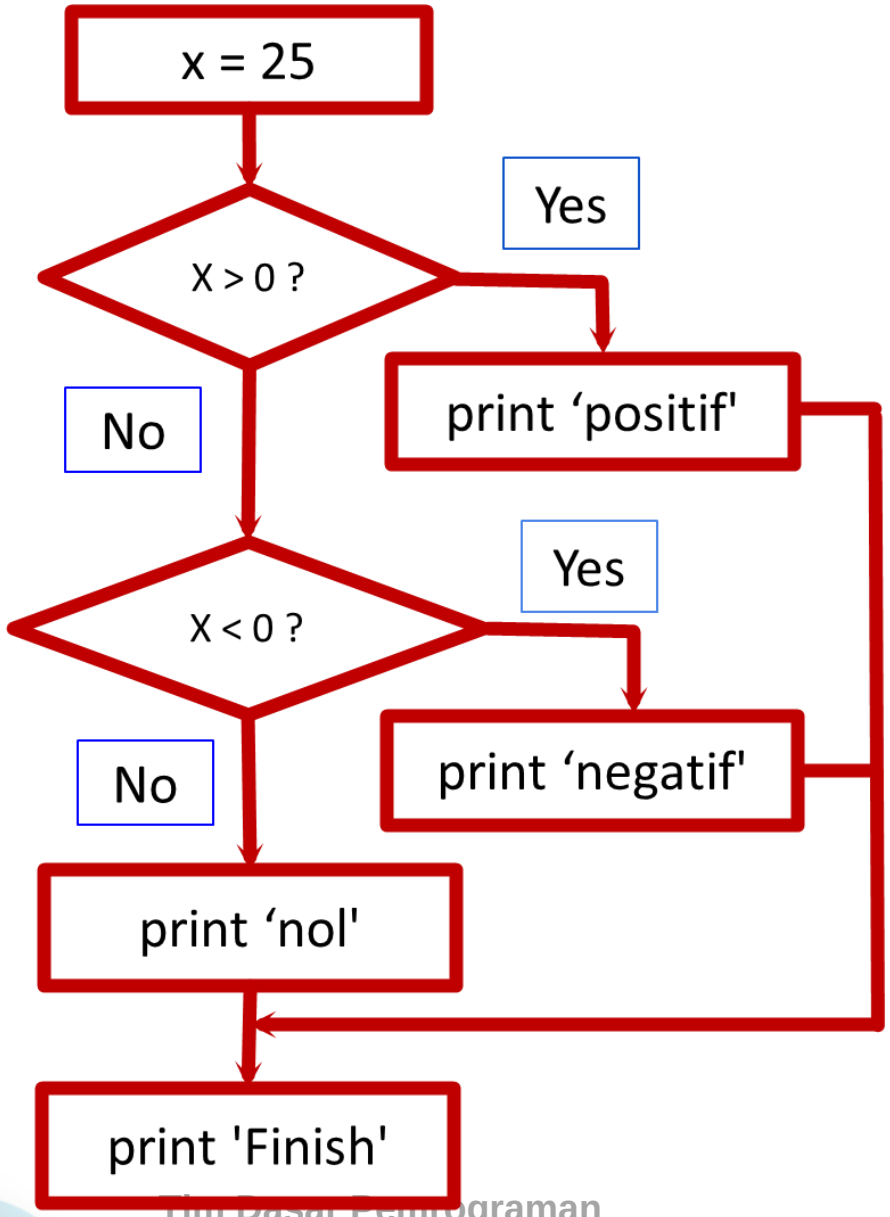
```
if x < y and x < z then
     output (“x adalah yang terkecil”)
else if y < z then
     output (“y adalah yang terkecil”)
else
     output(“z adalah yang terkecil”)
```

Penulisan lain untuk Analisis Banyak Kasus

Notasi Algoritmik	Notasi yang lain

depend on <list-nama>
  <kondisike-1>: <aksike-1>
  <kondisike-2>: <aksike-2>
  .....
  <kondisike-n>: <aksike-n>
	if <kondisike-1> then
	  <aksike_1>
	else if <kondisike-2> then
	  <aksike-2>
	.....
	else if <kondisike-n> then
	  <aksike-n>
	endif

depend on <list-nama>
  <kondisike-1>: <aksike-1>
  <kondisike-2>: <aksike-2>
  .....
  <kondisike-n>: <aksike-n>
  else: <aksi-else>
	if <kondisike-1> then
	  <aksike_1>
	else if <kondisike-2> then
	  <aksike-2>
	.....
	else if <kondisike-n> then
	  <aksike-n>
	else
	  <aksi-else>
	endif



Contoh Kasus

- Skenario input

X = 0

Pengecekan pertama x > 0 ?

Ya, maka output adalah positif

Kemudian diluar dari percabangan ada output lagi

finish

- Skenario Output

Positif

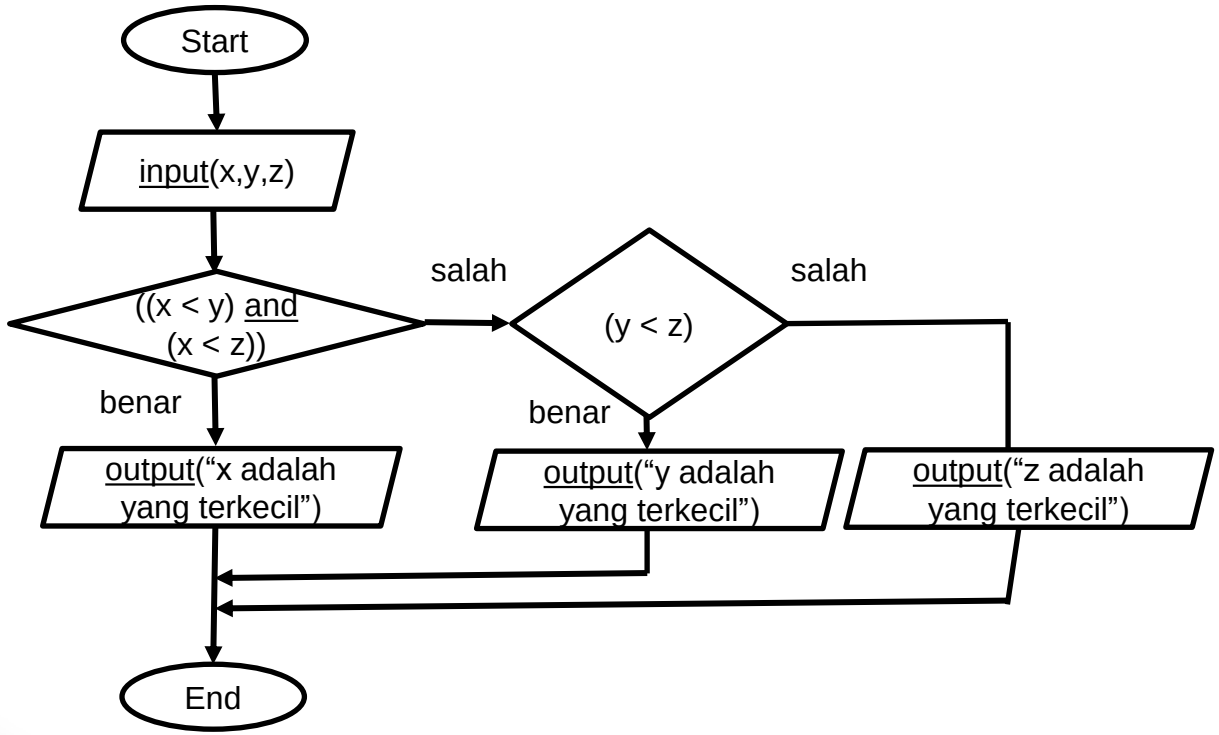
Finish

If, Else If, Else

- Program Min3Angka
- Program untuk mencetak keterangan nilai minimal dari tiga bilangan bulat
- Contoh:
- Input angka 5,7,2
- Output: 2 adalah angka yang terkecil

If, Else If, Else

- Baik if ataupun else if boleh menggunakan ekspresi boolean majemuk,
- Pada else jelas tidak ada ekspresi boolean



Program Min3Angka
{Program untuk mencetak keterangan nilai minimal tiga bilangan bulat}

KAMUS
x , y , z : integer

ALGORITMA
```
input(x, y, z)
if ((x < y) and (x < z)) then
    output("x adalah yang terkecil")
else if (y < z) then
    output("y adalah yang terkecil")
else
    output("z adalah yang terkecil")
endif
```

Program Min3Angka
{Program untuk mencetak keterangan nilai minimal tiga bilangan bulat}

KAMUS
x , y , z : integer

ALGORITMA
```
input(x, y, z)
depend on (x,y,z)
    ((x < y) and (x < z)) : output("x adalah yang terkecil")
    (y < z)               : output("y adalah yang terkecil")
    else                  : output("z adalah yang terkecil")
```

Kondisi dengan suatu ekspresi konstan

- Jika kondisike-1, kondisike-2…, kondisi ke-n dapat dinyatakan dalam bentuk: nama = **const-exp** (const-exp adalah suatu ekspresi konstan), maka dapat digunakan switch.

Notasi Algoritmik	C/C++/C#/Java

depend on nama
nama=const-exp-1 : ekspresike-1
nama=const-exp-2 : ekspresike-2
...
else : ekspresi-else
	switch (nama) {
	case const-exp-1: ekspresike_1;
	[break;]
	case const-exp-2: ekspresike_2;
	[break;]
	...
	default : ekspresi_else;
	[break;]
	}

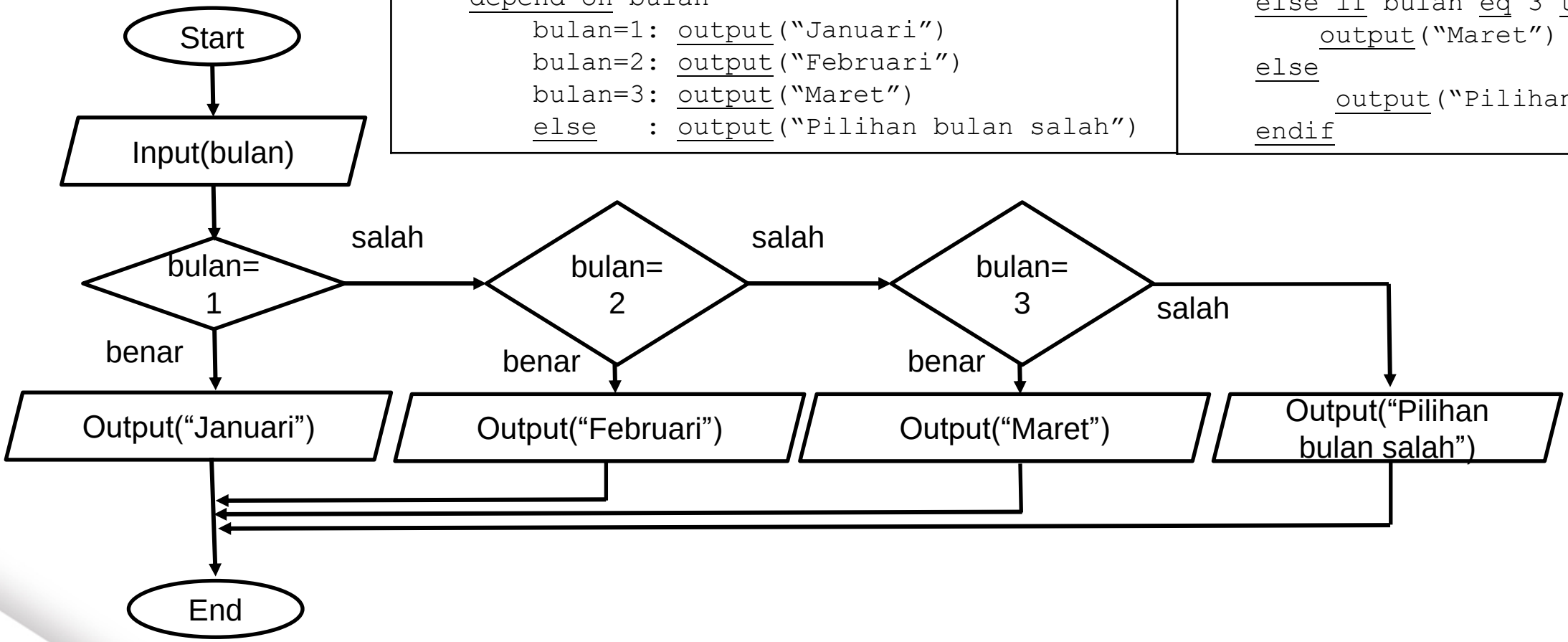
Contoh Ekspresi Konstan

Algoritma

```
input(bulan)
depend on bulan
    bulan=1: output("Januari")
    bulan=2: output("Februari")
    bulan=3: output("Maret")
    else   : output("Pilihan bulan salah")
```

Algoritma

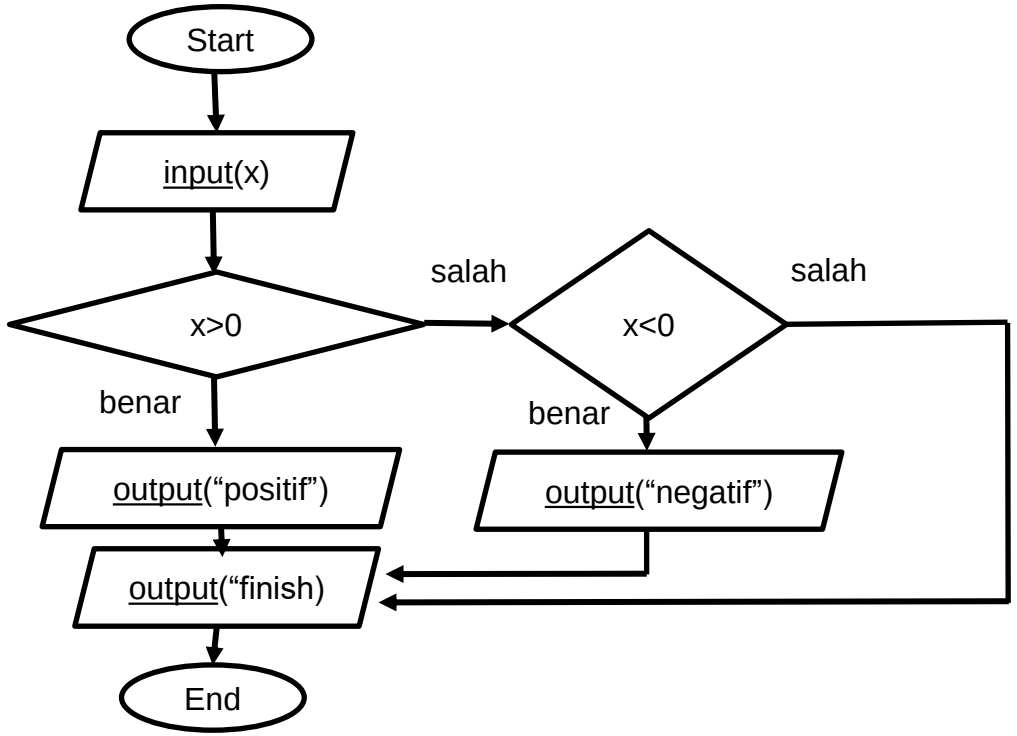
```
input(bulan)
if bulan eq 1 then
    output("Januari")
else if bulan eq 2 then
    output("Februari")
else if bulan eq 3 then
    output("Maret")
else
    output("Pilihan bulan salah")
endif
```



Notasi Algoritmik (Cek Bilangan Bulat)

```
Program CekBilanganBulat
{Program dengan notasi algoritmik untuk mengecek bilangan bulat positif/negatif/NOL}
KAMUS
    x : integer
ALGORITMA
    input(x)
    if x > 0 then
        output("positif")
    else if x < 0 then
        output("negatif")
    else
        output("nol")
    output("finish")
```

```
Program CekBilanganBulat
{Program dengan notasi algoritmik untuk mengecek bilangan bulat positif/negatif/NOL}
KAMUS
    x : integer
ALGORITMA
    input(x)
    depend on (x)
        x>0: output("positif")
        x<0: output("negatif")
    else   : output("nol")
    output("finish")
```



Program Wujud Air

- Spesifikasi:

Input: s bertipe integer

Proses: menuliskan wujud air sesuai dengan nilai s, bisa diasumsikan sebagai suhu

Jika s ≤ 0 maka wujud air beku

Jika 0 < s ≤ 100 maka wujud cair

jika s > 100 maka wujud uap

Output: “beku” atau “cair” atau “uap”

Program Wujud Air

```
PROGRAM wujudAir
{menuliskan wujud air sesuai dengan nilai s, bisa diasumsikan sebagai suhu}
{s dalam hal ini representasikan dengan integer}
```

```
KAMUS
s: integer
```

```
ALGORITMA
Input(s)
Depend on(s)
   s ≤ 0: output(“beku”)
   0 < s ≤ 100: output(“cair”)
   s > 100: output(“uap”)
```

Kasus Komputasional

- Deskripsi: mawar merupakan mahasiswa Teknik informatika, dia ingin mendapatkan beasiswa, syarat untuk mendapatkan beasiswa dilihat berdasarkan IPK. Hanya mahasiswa yang memiliki IPK tinggi yang dianggap lulus mendapat beasiswa. Jika IPK sedang maka dia dipertimbangkan. Jika IPK rendah tidak lulus. Buat system untuk permasalahan tersebut sehingga mawar bisa mengecek apakah dia lulus dapat beasiswa atau tidak

- Keterangan:
  - IPK ≥ 3.5 dianggap tinggi (berarti dia akan lulus)
  - IPK minimal 2.5 sampai 3.49 dianggap sedang (dipertimbangkan)
  - IPK dibawah 2.5 dianggap rendah (berarti tidak lulus)
  - Selain itu, missal -1 atau 5, cetak Inputan Salah

- Input: IPK merupakan bilangan real

- Output: tulisan “lulus” atau “dipertimbangkan” atau “tidak lulus” atau “inputan salah”

- Contoh input-output:
  - Input: 3.6
  - Output: Lulus

Program Beasiswa

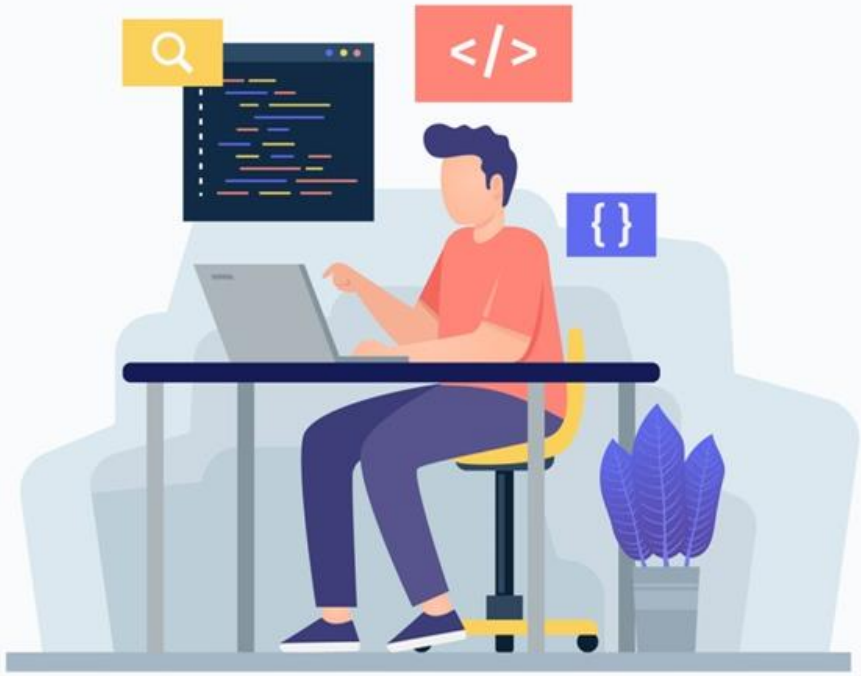
```
PROGRAM Beasiswa
{Program untuk mengecek kelulusan beasiswa berdasarkan IPK}

KAMUS
ipk: real

ALGORITMA
input(ipk)
depend on(ipk)
   ipk ≥ 3.5: output(“lulus”)
   2.5 ≤ ipk < 3.5: output(“dipertimbangkan”)
   ipk < 2.5: output(“tidak lulus”)
```

PROGRAM STUDI
TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

MATA KULIAH
Dasar Pemrograman



Analisis Kasus – Analisis Bersarang

TIM DASAR PEMROGRAMAN
TEKNIK INFORMATIKA S1
UNIVERSITAS DIAN NUSWANTORO

Variasi Analisis Banyak Kasus

- Pada kehidupan nyata permasalahan komputasional yang berkaitan dengan analisis kasus juga terkadang terdapat system prasyarat.
- Sistem prasyarat ini misalnya:
  - Untuk mendapatkan beasiswa mahasiswa harus mendapatkan persetujuan dahulu dari dosen wali sebagai syarat utama, baru setelah itu dia boleh menggunakan ipknya untuk kemudian dinilai apakah layak mendapatkan beasiswa
  - Dalam kasus tersebut syarat utama = persetujuan dimana syarat utama ini merupakan prasyarat untuk syarat berikutnya yaitu pengecekan kelayakan berdasarkan IPK
- Dari kasus diatas muncul ide untuk membuat blok if didalam if atau istilahnya analisis kasus bersarang atau nested if

Analisis Kasus Bersarang

depend on <list-nama_ke-1>

<kondisi_ke-1>: depend on <list-nama_ke-1.1>

<kondisi_ke-1.1>: depend on <list-nama_ke-1.1.1>

<kondisi_ke-1.1.1>:

.....

<kondisi_ke-2>: <aksi_ke-2>

.....

<kondisi_ke-n>: <aksi_ke-n>

Bentuk yang lebih mudah dibaca

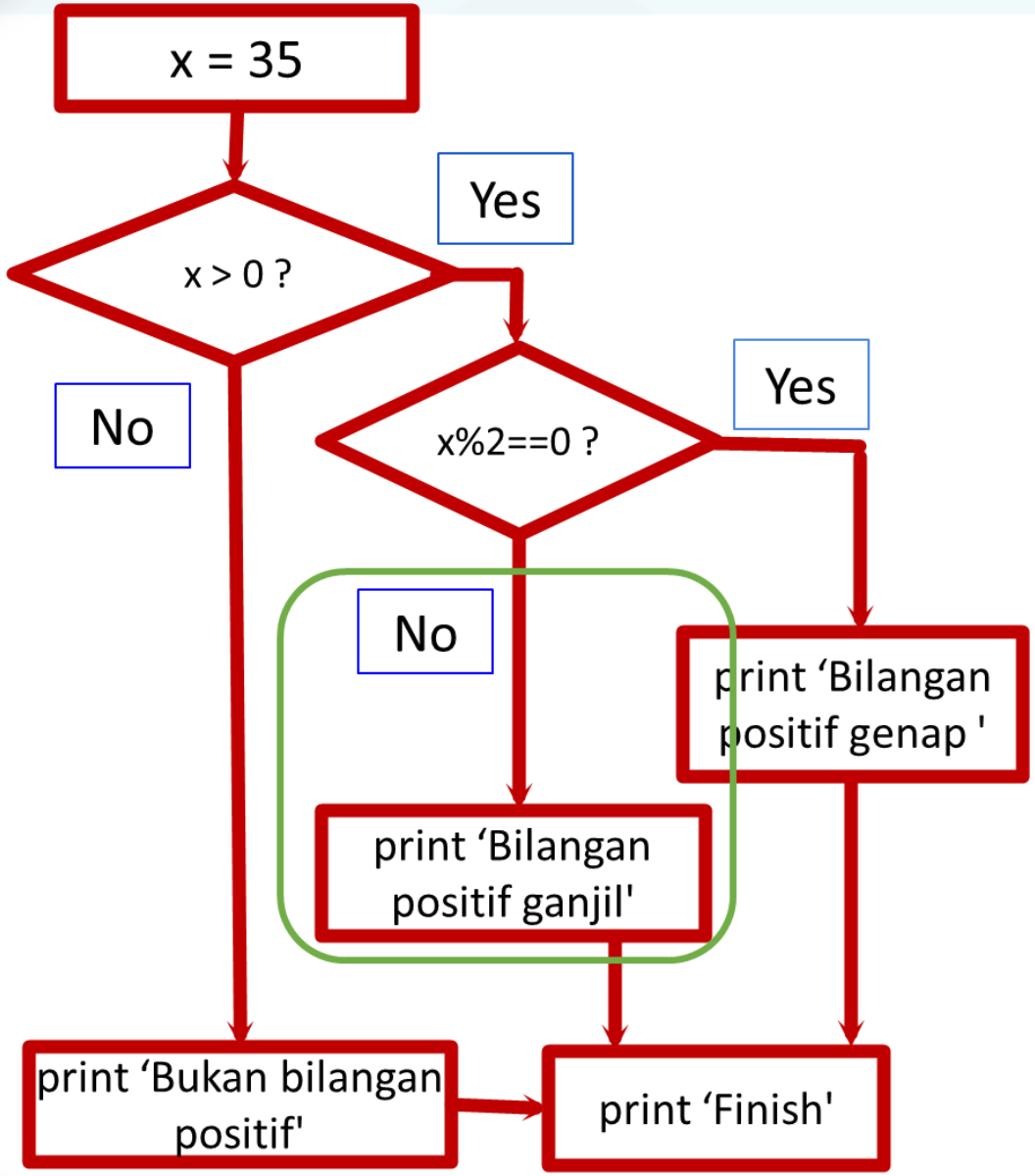
Notasi Algoritmik (if dan depend on)	Notasi yang lain

if <kondisike-1> then
  <aksike_1>
  if <kondisike-1.1> then
  <aksike-1.1>
  depend on <list-nama>
  <kondisike-1>: <aksike-1>
  <kondisike-2>: <aksike-2>
  .....
  <kondisike-n>: <aksike-n>
else if <kondisike-2> then
  <aksike-2>
  if <kondisike-2.1> then
  <aksike-2.1>
.....
else if <kondisike-n> then
  <aksike-n>
else
  <aksike-else>
endif
	if <kondisike-1> then
	  <aksike_1>
	  if <kondisike-1.1> then
	  <aksike-1.1>
	  if <kondisike-1.1.1> then
	  <aksike-1.1.1>
	else if <kondisike-2> then
	  <aksike-2>
	  if <kondisike-2.1> then
	  <aksike-2.1>
	.....
	else if <kondisike-n> then
	  <aksike-n>
	else
	  <aksike-else>
	Endif

Adakah Variasi Lain

- Algoritma mengecek suatu bilangan positif atau bukan, cek pula bilangan positif tersebut ganjil atau genap?
- Dalam kode dibawah ini, ada kondisi bersarang pada pernyataan if di tingkat atas

```
if x > 0 then
        if x mod 2 == 0 then
                output(“bilangan positif genap”)
        else
                output(:bilangan positif ganjil”)
        endif
else
        output(“bukan bilangan positif”)
endif
output(“finish”)
```



Contoh Kasus

- Skenario input x = 35

1. Dicek apakah x>0 jawabannya Ya maka dilanjutkan ke

1.1 apakah x mod 2 sama dengan 0, tidak, maka dia akan loncat ke blok else

1.2 else, outputkan “bilangan positif ganjil”

Diluar dari percabangan yang bersarang tersebut terdapat cetak output “finish”

Output:
Bilangan positif ganjil
Finish

Program PositifGanjilGenap

```
Program PositifGanjilGenap
{Program untuk  mengecek suatu bilangan positif atau bukan, cek pula bilangan positif tersebut ganjil atau genap?}

KAMUS
    x : integer

ALGORITMA
    input(s)
    if x > 0 then
        if x mod 2 eq 0 then
            output("Bilangan positif genap")
        else
            output("Bilangan positif ganjil")
    else
        output("Bukan bilangan positif")
    endif
    output("finish")
```

Kasus Komputasional

- Deskripsi: Mawar membutuhkan program pendataan uang saku mahasiswa. Syarat utamanya adalah seseorang yang menggunakan system tersebut adalah mahasiswa. Kemudian dia akan ditanya berapa uang sakunya. Jika uang sakunya lebih dari 1 juta maka dianggap mahasiswa mampu. Jika kurang dari itu dianggap kurang mampu. Lalu jika seseorang yang menggunakan system tersebut bukan mahasiswa, maka akan mencetak tulisan bukan mahasiswa
- Input: karakter ‘y’ atau ‘t’ untuk verifikasi mahasiswa atau tidak dan uang saku dalam tipe integer

Program Uang Saku

```
PROGRAM UangSaku
{program verifikasi mahasiswa dan pengecekan mahasiswa mampu atau kurang mampu}
```

```
KAMUS
c: char
U: integer
```

```
ALGORITMA
Output("apakah anda mahasiswa? (y/t)")
input(c)
if (c == 'y' or c == 'Y') then
   output ("masukkan uang saku per-bulan?")
   input(u)
   if u > 1000000 then
       output("mahasiswa mampu")
   else
       output("mahasiswa kurang mampu")
else
   output("bukan mahasiswa")
```

Referensi

Utama:

1. Bjarne Stroustrup, 2014, Programming: Principles and Practice Using C++ (Second Edition), Addison-Wesley Professional

Pendukung:

1. Introduction to Computer Science and Programming in Python, MIT https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-0001-introduction-to-computer-science-and-programming-in-python-fall-2016

2. Introduction to Computer Science and Programming, MIT https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-00sc-introduction-to-computer-science-and-programming-spring-2011/index.htm

TERIMA KASIH

ANY QUESTIONS?