PROGRAM STUDI
TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

Mata Kuliah
**Dasar Pemrograman**

# Analisis 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

# Analisis Kasus Tunggal

# Analisis Kasus

- Memungkinkan untuk membuat teks algoritma yang sama tetapi menghasilkan eksekusi yang berbeda – beda
- Analisis kondisi / aliran kendali / pencabangan / branching
- Terdiri dari:
  - Kondisi: ekspresi yang bernilai benar / salah
  - Aksi: instruksi yang akan dilakukan jika kondisi yang dipasangkan dipenuhi, dapat berupa ekspresi atau perintah dasar lain

# Analisis Satu Kasus

- Pengecekan terhadap satu kondisi yang memenuhi syarat.
  - Jika benar lakukan aksi, jika salah abaikan
- Menggunakan keyword **if … then …** ditutup dengan **endif**
- Format Penuilsan **notasi algoritmik**:

```
if <kondisi> then
    <aksi>
    <aksi>
endif
```

# Kondisi di dalam if

- Merupakan ekspresi Boolean, sehingga akan menghasilkan nilai true atau false
- Untuk analisis satu kasus, **jika kondisi bernilai benar maka aksi akan dilakukan, tetapi jika kondisi bernilai salah maka tidak akan terjadi apa - apa (efek neto "kosong")**

# Contoh

- Misal: seseorang akan dinyatakan LULUS jika nilainya diatas 60

```
if nilai > 60 then
    Output “dinyatakan LULUS”
endif
```

# Ekspresi Boolean Majemuk

- Sebagai contoh, suatu bilangan bulat a akan di cek kondisinya dengan syarat, a adalah bilangan genap dan bilangan tersebut harus lebih dari 2
- Bagaimana kita menuliskan ekspresi booleannya?
  - Asumsi a terdefinisi dengan input akan dimasukkan oleh user
  - Pengecekan **a bilangan genap**? a mod 2 = 0 (dibaca: sisa bagi dari a dibagi 2 adalah 0)
  - Pengecekan **a lebih dari 2**? a > 2 (dibaca: a lebih dari 2)
  - Kedua ekspresi diatas perlu digabung dengan operator '**and**' sehingga ekspresi Boolean yang utuh adalah ((a%2 == 0) and a>2)

# Scope dalam Pemrograman

- Scope atau lingkup disini penting sekali
- Untuk notasi algoritmik ditandai dengan tab atau 4 spasi, tetapi untuk pemrograman Bahasa C++ ditandai dengan { dan }.
- Scope bermanfaat untuk dimana suatu variable itu dikenal dan kapan suatu variable tidak terpakai lagi atau dihancurkan
- Peran scope di penggunaan analisis kasus adalah untuk membatasi aksi. Dalam hal ini setelah kondisi terpenuhi maka aksi akan dilakukan.
- **Jika seandainya kita mendefinisikan suatu variable ke dalam lingkup aksi, maka diluar itu variable tersebut tidak dapat digunakan.**

# Program Ribuan Ratusan Puluhan Satuan

**JUDUL Program Ribuan Ratusan Puluhan Satuan**
**{Input bilangan bulat positif ribuan}**
**{Mencari masing-masing 1 angka ribuan, ratusan, puluhan, satuan dan tuliskan hasilnya}**

**KAMUS**

n : integer {angka yang dibaca, 0 sampai 9999}
rib : integer {ribuan, bilangan bulat positif}
rat : integer {ratusan, bilangan bulat positif}
pul : integer {puluhan, bilangan bulat positif}
sat : integer {satuan, bilangan bulat positif}

```
Masukkan angka ribuan antara 0 sampai 9999 = 5713
5 ribuan, 7 ratusan, 1 puluhan, dan 3 satuan.
```

```
Masukkan angka ribuan antara 0 sampai 9999 = 12345

Process returned 0 (0x0)   execution time : 3.266 s
Press any key to continue.
```

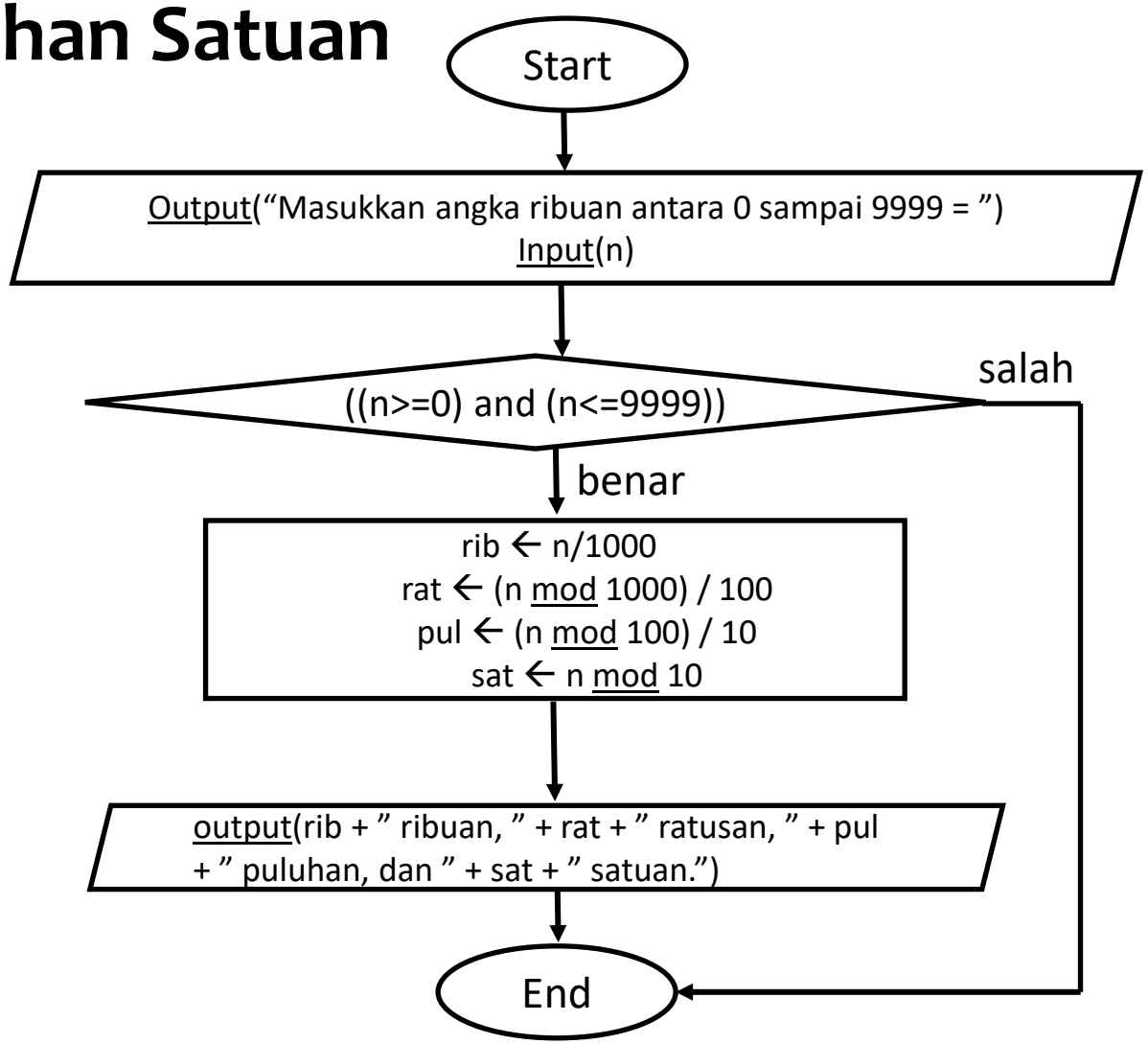
**ALGORITMA**

```
output("Masukkan angka ribuan antara 0 sampai 9999 = ")
input(n)
if ((n>=0) and (n<=9999)) then
    rib ← n/1000
    rat ← (n mod 1000) / 100
    pul ← (n mod 100) / 10
    sat ← n mod 10
    output(rib + " ribuan, " + rat + " ratusan, " + pul + " puluhan, dan " + sat + " satuan.")
endif
```

# Flowchart Ribuan Ratusan Puluhan Satuan

**ALGORITMA**

```
output("Masukkan angka ribuan antara 0 sampai 9999 = ")
input(n)
if ((n>=0) and (n<=9999)) then
    rib ← n/1000
    rat ← (n mod 1000) / 100
    pul ← (n mod 100) / 10
    sat ← n mod 10
    output(rib + " ribuan, " + rat + " ratusan, " + pul + "
puluhan, dan " + sat + " satuan.")
endif
```



FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

# Analisis 2 Kasus

# Analisis Dua Kasus Komplementer

- Pengecekan terhadap satu kondisi yang memenuhi syarat
  - Jika benar lakukan aksi, jika salah lakukan aksi lainnya
- Menggunakan keyword **if... then else ...**

```
if <kondisi> then
    <aksi>
else
    <aksi>
endif
```

# Contoh Analisa Dua Kondisi

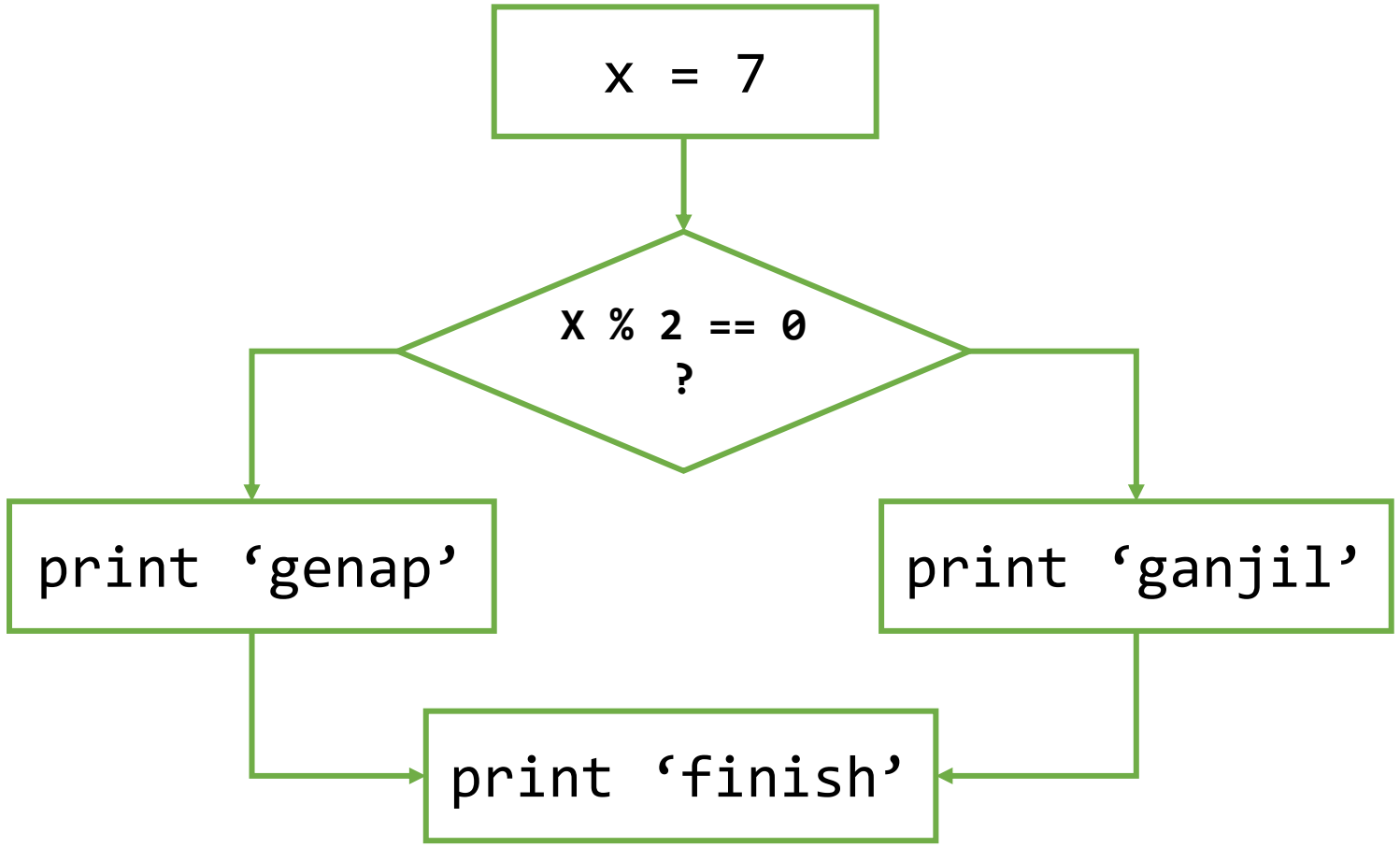
- Algoritma untuk mengecek bilangan genap atau ganjil dari masukan x

```
PROGRAM GenapGanjil
{program untuk mengecek apakah a bilangan genap atau ganjil}
```

```
KAMUS
x : integer
```

```
ALGORITMA
input(x)
if x mod 2 == 0 then
   output(“genap”)
else
   output(“ganjil”)
```

# Contoh Kasus Analisa Dua Kondisi




Skenario

Input:
X = 7

Output:
Ganjil
Finish

FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

# Notasi Algoritmik untuk analisis dua kasus komplemen

# Program MaxAB

Spesifikasi:

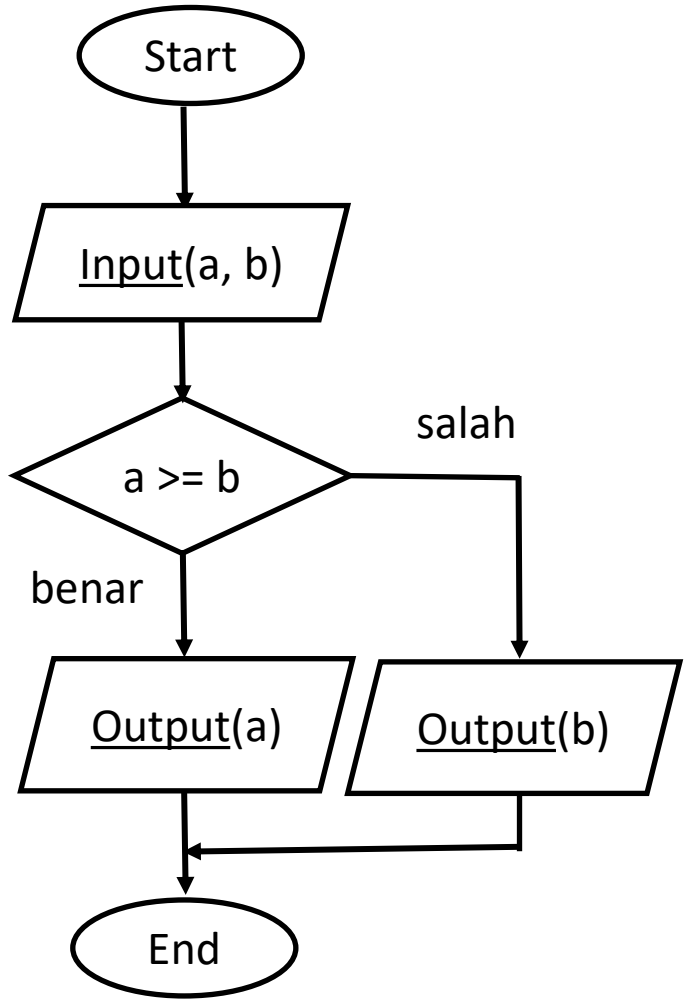
- Input: a dan b bilangan bulat
- proses: menuliskan nilai yang lebih besar dari kedua bilangan tersebut, cetak a jika a >= b, selain itu cetak b (jika a < b)
- Output: a atau b

# Notasi Algoritmik – Program MaxAB

```
PROGRAM MaxAB
{program untuk mencetak bilangan dengan nilai
maksimal dua bilangan bulat a dan b}
```

```
KAMUS
a, b: integer
```

```
ALGORITMA
input(a)
input(b)
if a>= b then
   output(a)
else
   output(b)
Endif
```

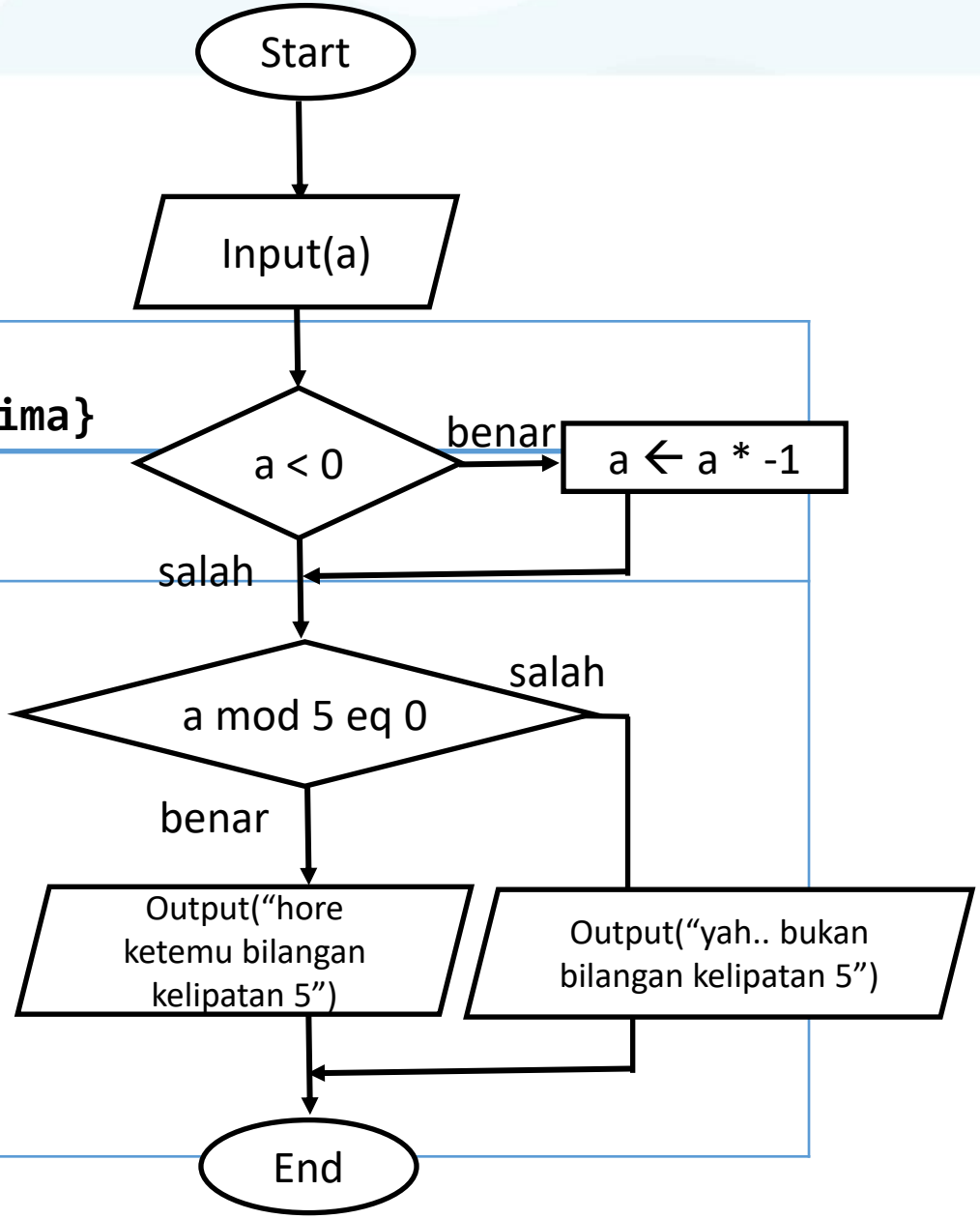


# Kasus Komputasional

- Deskripsi: Mawar suka sekali dengan permainan bilangan. Dia ingin memastikan bilangan bulat yang dia dapat adalah bilangan yang absolut (tidak boleh bernilai negative) kemudian dia juga ingin memastikan bahwa bilangannya itu merupakan bilangan kelipatan 5. jika bilangan tadi kelipatan 5 maka dia ingin mencetak “hore ketemu bilangan kelipatan 5”, jika tidak, dia ingin mencetak “yah, bukan bilangan kelipatan 5”. Buatkan program untuk mawar
- Input: a merupakan bilangan bulat boleh positif atau negative
- Output: tulisan “hore ketemu bilangan kelipatan 5” atau “yah, bukan bilangan kelipatan 5”
- Input-Output: input: 10, output: hore ketemu bilangan kelipatan 5

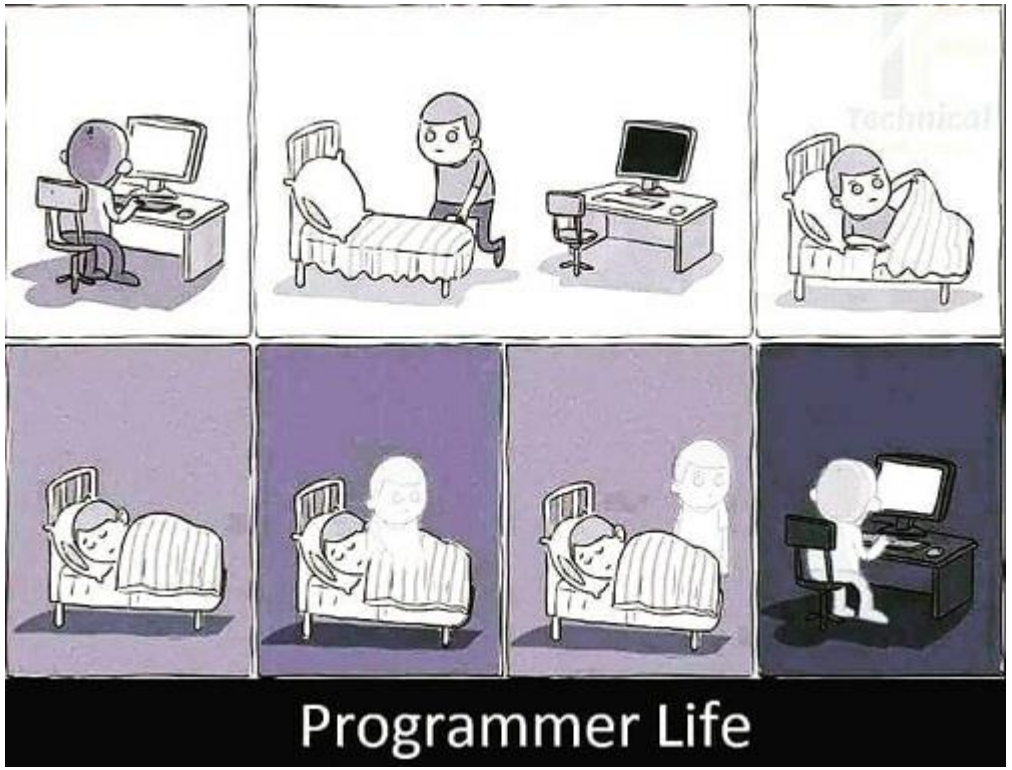
# Program Kelipatan 5

```
PROGRAM Mawar
{program untuk penyelesaian kasus kelipatan lima}

KAMUS
a: integer

ALGORITMA
input(a)
if a < 0 then
   a ← a * (-1)
endif
if a mod 5 == 0 then
   output(“hore ketemu bilangan kelipatan 5”)
else
   output(“yah, bukan bilangan kelipatan 5”)
endif
```



# Cara cepat jadi programmer?

- BERLATIH BERLATIH BERLATIH !
- RESEP: NGODING 3X SEHARI SETELAH MAKAN



# 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?