



**PROGRAM STUDI**  
**TEKNIK INFORMATIKA**  
**FAKULTAS ILMU KOMPUTER**  
**UNIVERSITAS DIAN NUSWANTORO**

Mata Kuliah  
**Dasar Pemrograman**



# Perulangan (Looping)

TIM DASAR PEMROGRAMAN  
TEKNIK INFORMATIKA S1  
UNIVERSITAS DIAN NUSWANTORO

## Capaian Pembelajaran

- Setelah mengikuti kuliah ini mahasiswa dapat menjelaskan dan mempraktekkan jenis pengulangan berdasarkan jumlah pengulangan dan berdasarkan kondisi kondisi, dua aksi, dan pencacah dalam pemrograman
- Setelah mengikuti kuliah ini mahasiswa dapat menjelaskan dan mempraktekkan jenis pengulangan bersarang

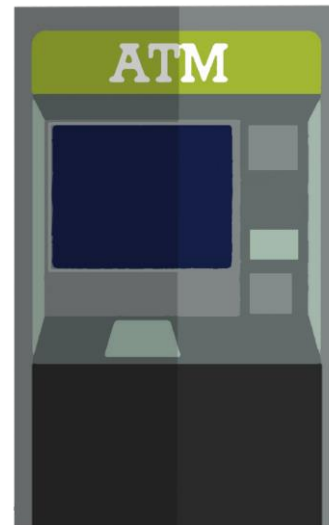
## Latar Belakang Pentingnya Perulangan [1]

- Melakukan suatu instruksi atau aksi secara berulang – ulang; menulis hal yang sama dengan banyak dan berulang
- Komputer: performansi sama
- Manusia: cenderung melakukan kesalahan (letih/bosan)



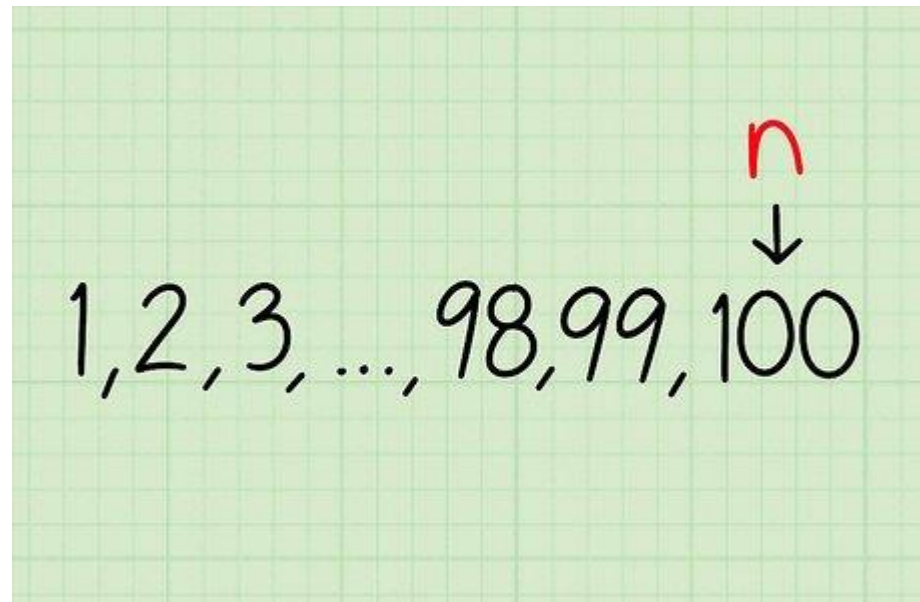
## Latar Belakang Pentingnya Perulangan [2]

- Lampu lalu lintas bergantian menyala sesuai durasi waktu
- Mesin ATM menampilkan menu yang sama tiap kali nasabah memasukkan kartu ke dalam mesin



## Latar Belakang Pentingnya Perulangan [3]

- Menuliskan deret bilangan hingga  $n$  dengan pola tertentu
- Apakah akan diketikkan semua? Sampai ber baris – baris ?



Handwritten text on a green grid background showing a sequence of numbers:  $1, 2, 3, \dots, 98, 99, 100$ . Above the number 100, there is a red letter  $n$  with a black arrow pointing down to it, indicating the end of the sequence.



**FAKULTAS ILMU KOMPUTER  
UNIVERSITAS DIAN NUSWANTORO**

# Perulangan

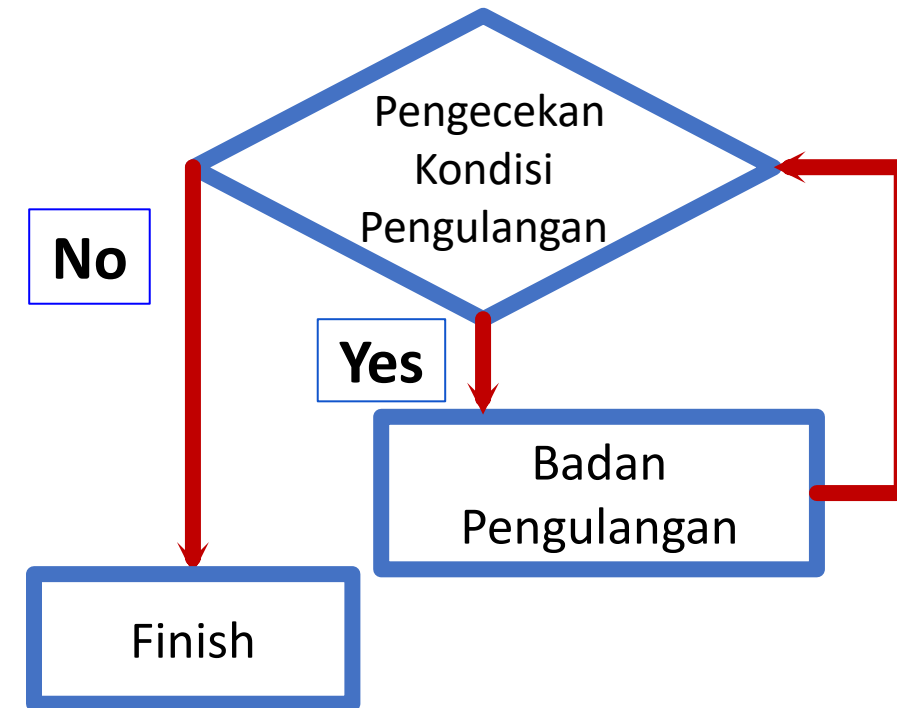
# Perulangan

- Proses untuk melakukan instruksi berulang – ulang sesuai batas yang ditentukan
- Istilah lain looping atau iterasi
- Fungsi perulangan:
  - Memudahkan dalam menuliskan kode yang berulang – ulang
  - Mempersingkat kode program

# Elemen Perulangan

Elemen perulangan:

- **Kondisi** perulangan berhenti
  - Mengevaluasi ekspresi logika yang menyebabkan perulangan berhenti
- **Badan perulangan**
  - aksi yang diulang



# Notasi Perulangan

- Jenis notasi perulangan:
  1. Berdasarkan jumlah perulangan
  2. Berdasarkan kondisi berhenti
  3. Berdasarkan kondisi perulangan
  4. Berdasarkan dua aksi
  5. Berdasarkan pencacah

## Berdasarkan jumlah Perulangan

- Aksi akan diulang sebanyak  $n$  kali
- Perulangan pasti akan berhenti suatu saat setelah mengulang sebanyak  $n$

Notasi Algoritmik	Non-Perulangan	Notasi
<u>repeat</u> $n$ <u>times</u> <ekspresi>	output("hello world") output("hello world") output("hello world") output("hello world") output("hello world")	<u>repeat</u> 5 <u>times</u> Output("hello world")

# Program Cetak “Mahasiswa”

**Program CetakMahasiswa**  
{program untuk mencetak kata mahasiswa berkali – kali dengan Batasan n input oleh user}

## KAMUS

n: integer

## ALGORITMA

input(n)

repeat n times

    output(“mahasiswa”)

output(“finish”)

## Contoh Persoalan Komputasional

- Tulis semua bilangan positif lebih dari 0, dari 1, 2, 3, ... sampai N.
- Perhatikan bahwa N merupakan bilangan bulat positif yang diinputkan oleh user

**Program CetakBilanganPositif**

{program untuk mencetak bilangan positif lebih dari 0}

### KAMUS

n, i: integer (I merupakan bilangan yang akan ditampilkan dalam loop)

### ALGORITMA

input(n)

i ← 1

repeat n times

    output(i)

    i = i+1;

## Berdasarkan kondisi Berhenti

- Suatu aksi akan dihentikan jika kondisi berhenti terpenuhi (bernilai True), jika tidak, ekspresi atau aksi tersebut akan diulang terus – menerus.
- Tes kondisi-berhenti dilakukan setelah ekspresi atau aksi dilakukan, sehingga setidaknya ada 1 ekspresi atau aksi yang akan dilakukan lebih dulu
- Perulangan berpotensi mengalami ‘kebocoran’, jika ada kemungkinan bahwa seharusnya ekspresi atau aksi tidak pernah boleh dilakukan untuk kasus tertentu

### Notasi Algoritmik

**repeat**

<ekspresi>

**until** kondisi-berhenti

# Perulangan berdasarkan Kondisi Berhenti

- Non-Perulangan

{ program sederhana, tanya “apa mau belok kiri” }

If belokkiri != ‘y’ then

    Output(“apa mau belok kiri [Y/T]”)

If belokkiri != ‘y’ then

    Output(“apa mau belok kiri [Y/T]”)

If belokkiri != ‘y’ then

    Output(“apa mau belok kiri [Y/T]”)

- Perulangan

{ program sederhana, tanya “apa mau belok kiri” }

repeat

    output(“ apa mau belok kiri[Y/T] ”)

until belokkiri = ‘y’

# Program BelokKiri

## Program BelokKiri

{program untuk tanya 'apa mau belok kiri[Y/N] diberikan variable belokkiri yang merupakan input user bertipe karakter}

### KAMUS

belokkiri: char

### ALGORITMA

repeat

    output('apa mau belok kiri[Y/N]')

    Input(belokkiri)

until belokkiri = 'y'

## Contoh Persoalan Komputasional yang Lain

- Tulis semua bilangan positif lebih dari 0 dari 1, 2, 3, ... sampai N.  
perhatikan bahwa N merupakan bilangan bulat positif yang diinputkan oleh user.

Program CetakBilanganPositif2

{program untuk mencetak bilangan positif lebih dari 0}

### KAMUS

n, i: integer (I merupakan bilangan yang akan ditampilkan dalam loop)

### ALGORITMA

input(n)

i ← 1

repeat

    output(i)

    i ← i + 1

until i = n {perhatikan I = n juga bisa diganti i >= n}

# Notasi Perulangan

- Jenis notasi perulangan:
  1. Berdasarkan jumlah perulangan
  2. Berdasarkan kondisi berhenti
  3. **Berdasarkan kondisi perulangan**
  4. **Berdasarkan dua aksi**
  5. **Berdasarkan pencacah**

## Berdasarkan Kondisi Perulangan

- Aksi akan dilakukan selama kondisi perulangan masih dipenuhi (bernilai True)
- Tes terhadap kondisi perulangan dilakukan setiap kali sebelum aksi dilaksanakan
- Perulangan ini berpotensi untuk menimbulkan aksi 'kosong' (tidak pernah melakukan apa-apa) karena pada tes yang pertama, kondisi perulangan tidak dipenuhi (bernilai false)

### Notasi Algoritmik

**while** (kondisi-pengulangan) **do**  
    <ekspresi>

# Pengulangan berdasarkan Kondisi Perulangan

- Non – Perulangan

{program sederhana tanya ‘apa mau belok kiri[Y/N]’}

if belokkiri != ‘Y’ then

    output(‘apa mau belok kiri[Y/N]’)

if belokkiri != ‘Y’ then

    output(‘apa mau belok kiri[Y/N]’)

if belokkiri != ‘Y’ then

    output(‘apa mau belok kiri[Y/N]’)

- Perulangan

{program sederhana tanya ‘apa mau belok kiri[Y/N]’}

output(‘apa mau belok kiri[Y/N]’)

input(belokkiri)

while belokkiri != ‘Y’ do

    output(‘apa mau belok kiri[Y/N]’)

input(belokkiri)

# Program BelokKiri

## Program BelokKiri

{program belokkiri, tanya “apa mau belok kiri? (y/t)” dengan diberikan variabel belokkiri yang merupakan input user yang bertipe karakter}

## KAMUS

belokkiri : char

## ALGORITMA

output(“apa mau belok kiri? (y/t)”)  
input(belokkiri)

while belokkiri ≠ ‘y’ do  
    output(“apa mau belok kiri? (y/t)”)  
    input(belokkiri)

# Program HalloNTimes

Program HalloNTimes  
{program mencetak kata “Hallo” sebanyak n kali}

## KAMUS

n : int

## ALGORITMA

input(n)

while  $n \geq 0$  and  $n < 10$  do  
     $n \leftarrow n + 1$   
    output(“Hallo”)

iterate

&lt;aksi-1&gt;

stop (kondisi berhenti)

&lt;aksi-2&gt;

## Berdasarkan dua Ekspresi

- Perulangan ini seolah – olah gabungan antara bentuk perulangan kedua dan ketiga
- Mekanisme yang dilakukan oleh perulangan ini adalah dengan melakukan secara otomatis aksi-1 pada eksekusi yang pertama kemudian dilakukan tes terhadap kondisi berhenti
- Tergantung kepada kondisi berhenti yang di uji:
  - Aksi-2 akan diaktifkan dan kemudian aksi-1 yang berikutnya diulang, atau
  - Perulangan dihentikan karena efek neto dari aksi-1 menghasilkan kondisi berhenti
- Perulangan ini berguna untuk kasus – kasus dimana aksi-2 merupakan hal yang harus dilakukan tergantung dari hasil aksi-1

# Program HalloNTimes

```
Program HalloNTimes  
{program mencetak kata "Hallo" sebanyak n kali}
```

## KAMUS

```
n,i : int
```

## ALGORITMA

```
input(n)  
i ← 0  
iterate  
    i ← i+1  
    output("Hallo")  
stop i ≥ n:  
    print("stop")  
    break
```

## Berdasarkan Pencacah

- Aksi akan dilakukan dengan memperhitungkan nilai dari nama-pencacah yang di-'jelajahi'
- Dengan memakai perulangan ini, pemrogram tidak perlu melakukan operasi terhadap sukses/predesor karena setiap kali selesai melakukan Aksi, otomatis mesin akan melakukan operasi untuk mendapatkan suksesor dari harga yang berlaku saat itu nama pencacah.
- Range bisa dari kecil ke besar atau sebaliknya
- Setelah pelaksanaan perulangan selesai, harga yang tersimpan pada nama-pencacah tidak terdefinisi: jika hendak dipakai, harus didefinisikan kembali

Notasi Algoritmik	Non- Pengulangan	Notasi
Nama-pencacah <u>traversal</u> [range] <ekspresi>	output("makan bang") output("makan bang") output("makan bang") output("makan bang") output("makan bang")	i <u>traversal</u> [1..5] output("makan bang")

# Program CetakMahasiswa

Program CetakMahasiswa

{Program untuk mencetak mahasiswa berkali-kali dengan batasan n input oleh user}

KAMUS

n, i : integer

ALGORITMA

input(n)

i traversal [1 ... n]

output("mahasiswa")

output("finish")

# Contoh Persoalan Komputasional

## Program CetakBilanganPositifRange

{Tulis semua bilangan positif lebih dari 0 mulai dari x sampai y. Perhatikan bahwa x dan y merupakan bilangan bulat positif yang di inputkan oleh user.}

### KAMUS

x, y , i : integer {i merupakan bilangan yang akan di outputkan nantinya didalam pengulangan}

### ALGORITMA

```
input(x,y)
i traversal [x ... y]
    output(i)
output("finish")
```

# Contoh Persoalan Komputasional

## Program CetakBilanganGenapPositifRange

{Tulis semua bilangan genap positif lebih dari 0 mulai dari 0 sampai n. Perhatikan bahwa n merupakan bilangan bulat positif yang di inputkan oleh user.}

### KAMUS

n, i : integer {i merupakan bilangan yang akan di outputkan nantinya didalam pengulangan}

### ALGORITMA

input(n)

i traversal [1 ... n]

i ← i+1

{variabel i akan digunakan sebagai jeda}

output(i)

output("finish")

# 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?