

BAHAN AJAR
DASAR – DASAR PEMROGRAMAN



PENULIS

Nurhaeka Tou, S.Kom., M. Kom

JURUSAN TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS BANGKA BELITUNG

2022

LEMBAR PENGESIHAN

Judul

Dasar-Dasar Pemrograman

Dosen Pengampu

1. Nurhaeka Tou, S.Kom., M.Kom
2. Putri Mentari Endraswari, S.Tr.Kom., M.Kom
3. Iski Zaliman, S.Kom., M.Kom
4. Umar Faruq Vista, S.Kom., M.Kom
5. Rodiatul Adawiyah, S.SI., M.T.I., M.IM.
6. Fransiskus Panca Juniawan, S.Kom., M.Kom

Penyusun

Nurhaeka Tou, S.Kom., M.Kom

Jurusan

Teknologi Informasi

Fakultas

Teknik

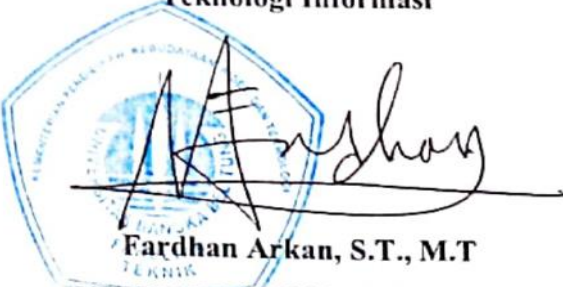
Instansi/PT

Universitas Bangka Belitung

Balunijuk, 02 Januari 2022

Plt. Ketua Jurusan

Teknologi Informasi



Fardhan Arkan, S.T., M.T

NI PPPK: 197409192021211003

KATA PENGANTAR

Segala puji bagi Allah SWT yang telah melimpahkan segala rahmat dan karunia-NYA, Solawat serta Salam Penulis haturkan pada junjungan kita Nabi Muhammad SAW yang telah membawa kita dari jalan kegelapan kepada jalan yang terang-menderang. Sehingga, penulis dapat menyelesaikan Bahan Ajar Dasar-Dasar Pemrograman tepat pada waktunya. Tujuan pembelajaran pemrograman dasar bagi Jurusan Teknologi Informasi untuk dijadikan sebagai pondasi awal bagi mahasiswa Teknologi Informasi, sehingga dapat dapat memahami dasar-dasar dan aturan dalam pemrograman.

Dasar-Dasar Pemrograman merupakan matakuliah wajib yang diajarkan kepada mahasiswa jurusan Teknologi Informasi. Hal ini penting bagi mereka yang ingin mempelajari tentang dasar dalam membuat program, struktur program yang tepat, dan tipe data yang harus ada dalam sebuah program. Bahan ajar ini diharapkan dapat membantu mahasiswa memahami bagaimana proses dasar pembuatan program.

Bahan ajar ini disusun berdasarkan kebutuhan dan penggunaan yang akan membuat program dasar, dalam hal ini mahasiswa. Sehingga, diharapkan dengan mempelajari Bahan Ajar ini seluruh pembaca mampu membuat program yang bermutu dan bernilai tinggi sehingga dapat bersaing merebut pangsa pasar yang semakin kompetitif.

Dalam menulis bahan ajar ini, penulis tidak luput dari berbagai kesalahan. Oleh karena itu, kritik dan saran sangat diharapkan demi kesempurnaan bahan ajar ini. Kami harapkan semoga buku ini dapat bermanfaat bagi Anda dan selamat belajar.

Balunijuk, 25 Agustus 2022

Nurhaeka Tou, S.Kom., M.Kom

DAFTAR ISI

KATA PENGANTAR	iii
DAFTAR ISI	iv
DAFTAR TEBEL	vii
PERTEMUAN 1	11
KONSEP DASAR PEMROGRAMAN	11
A. Tujuan Pembelajaran	11
B. Uraian Materi	11
1. Pengenalan Java	11
PERTEMUAN 2	14
TIPE DATA, VARIABEL DAN KONSTANTA	14
A. Tujuan Pembelajaran	14
B. Uraian Materi	14
1. Tipe Data	14
2. Variabel dan Konstanta	16
PERTEMUAN 3	17
OPERATOR DAN STRUKTUR PERCABANGAN	17
A. Tujuan Pembelajaran	17
B. Uraian Materi	17
1. Operator	17
2. Struktur Percabangan	19
PERTEMUAN 4	21
FUNGSI PERULANGAN	21
A. Tujuan Pembelajaran	21
B. Uraian Materi	21
1. Definisi Perulangan	21
2. Jenis-Jenis Perulangan	22
3. Jenis Perulangan dalam Perulangan	24
PERTEMUAN 5	25
INPUT DAN OUTPUT	25
A. Tujuan Pembelajaran	25
B. Uraian Materi	25

1. Input/Output	25
2. Jenis-Jenis Input/Output.....	26
PERTEMUAN 6.....	28
ARRAY SATU DIMENSI DAN MULTI DIMENSI	28
A. Tujuan Pembelajaran.....	28
B. Uraian Materi.....	28
1. Pengenalan Array	28
2. Pendeklarasian Array.....	28
3. Pengaksesan Elemen pada Array	29
4. Array Satu Dimensi	29
5. Array Multidimensi	29
PERTEMUAN 7.....	31
PEMROGRAMAN MODULAR	31
A. Tujuan Pembelajaran.....	31
B. Uraian Materi.....	31
1. Konsep Pemrograman Modular	31
2. Fungsi dalam Pemrograman Modular.....	32
3. Prosedur dalam Pemrograman Modular.....	34
5. Perbedaan Fungsi dan Prosedur dalam Pemrograman Modular	35
PERTEMUAN 8.....	37
UJIAN TENGAH SEMESTER (UTS).....	37
PERTEMUAN 9-10	38
PEMROGRAMAN BERORIENTASI OBJEK	38
A. Tujuan Pembelajaran.....	38
B. Uraian Materi.....	38
1. Definsi Object Oriented Programming (OOP).....	38
2. Kelebihan Object Oriented Programming (OOP)	39
3. Kekurangan Object oriented Programming (OOP)	39
4. Struktur bagian dalam OOP.....	39
PERTEMUAN 11-12	40
PEMROGRAMAN BERORIENTASI OBJEK LANJUTAN	40
A. Tujuan Pembelajaran.....	40
B. Uraian Materi.....	40

1. Deifinisi Modifier	40
2. Bagian dari Modifier	40
REFERENSI.....	42

DAFTAR TEBEL

Tabel 1. Tipe Data Integer.....	14
Tabel 2. Tipe Data Real.....	15
Tabel 3. Tipe Data Boolean.....	15
Tabel 4. Jenis Operator dan Fungsinya	18
Tabel 5. Tabel Konversi Tipe Data pada BufferedReader	26

PERTEMUAN 1

KONSEP DASAR PEMROGRAMAN

A. Tujuan Pembelajaran

1. Mahasiswa dapat mengetahui dan memahami model komunikasi
2. Mahasiswa dapat memahami dan menjelaskan konsep dari komunikasi data

B. Uraian Materi

1. Pengenalan Java

Java adalah salah satu bahasa pemrograman yang bersifat open source. Bahasa pemrograman java merupakan bahasa yang multiplatform (Bisa berjalan di berbagai macam system operasi). Hal ini dikarenakan, bahasa pemrograman java memiliki Jre (Java Runtime Environment) atau dapat dimaknai sebagai mesin tersendiri untuk mengeksekusi code binary dari kompilasi program yang telah kita buat [1].

Dalam pemrograman, java merupakan bahasa pemrograman yang berorientasi pada *Object Oriented Programming* (OOP), dimana seluruh metode dalam pemrograman berorientasi pada objek. Tujuan dari java mengadopsi pemrograman berorientasi objek adalah, agar mempermudah pengembangan program dengan mengikuti model-model yang ada pada kehidupan sehari-hari. Contohnya: Motor adalah sebuah objek yang dirancang dari beberapa objek yang lebih kecil, seperti; mesin, roda, stang, dll. Semua komponen pembentuk mobil ini saling terhubung, berinteraksi, dan melakukan komunikasi satu sama lain. Sama halnya dengan sebuah program, dimana objek yang besar akan dibentuk dari objek-objek kecil yang kemudian saling terhubung [2].

Bahasa pemrograman java di rilis secara resmi pada tahun 1996 oleh Sun Microsysteme, yang sampai saat ini masih terus berkembang. Pengembangan bahasa pemrograman java masih terus dilakukan hingga saat ini sudah sampai java versi 8. Berikut kelebihan dan kekurangan dari bahasa pemrograman java:

a. Kelebihan Bahasa Java

1) Mudah dipahami

Kelebihan dari java adalah dapat digunakan pada semua platform. Sehingga, pengguna-pengguna lain yang ingin belajar java, jika mengalami kesulitan dapat mengakses dari berbagai sumber atau berdiskusi di forum-forum java.

2) Merupakan bahasa OOP (*Object Oriented Programming*)

Kelebihan java yang menggunakan konsep OOP ini, dapat memudahkan bagi setiap pengguna dalam membuat aplikasinya.

- b. Kekurangan Java
 - 1) Aplikasi java yang dibuat masih rentan terhadap Compile (Melihat atau menggunakan source code), sehingga mudah dibajak.
 - 2) Java jika dijalankan pada Laptop/PC yang memiliki spesifikasi yang rendah, sedikit lama (lemot)

2. Konsep *Object Oriented Programming* (OOP)

Berikut beberapa Konsep Object Oriented Programming (OOP) yang digunakan pada bahasa Java:

- a. *Class* Merupakan sebuah blueprint yang digunakan untuk membuat suatu instant dari object. Selain itu, class juga didefinisikan sebagai group dari suatu object yang memiliki kemiripan atribut, behavior, dan relasi ke object lain. Contoh: Class Hewan, Class Mobil, Class Buah, dll.
- b. Object merupakan sebuah instance dari class yang secara umum digunakan untuk merepresentasikan (template) dari sebuah object dan suatu instance representasi nyata dari class itu sendiri. Contoh: Class buah, dalam class buah kita dapat membuat objek jeruk, apel, manggis, mangga, dll
- c. Attributes adalah nilai dari data yang ada pada suatu objek dalam class. Dalam pemrograman attribute merepresentasikan karakteristik dari object. Contoh: Dalam class Buah, terdapat warna, rasa. Kemudian, pada object apel: Atribut warna berisi merah, dan rasa berisi manis/asam.
- d. Method merupakan behavior yang digunakan untuk mengubah nilai attribute dari objek, menerima dan mengirim informasi dari/ke objek lain untuk melakukan sebuah instruksi. Dalam pemrograman java, method dibagi menjadi dua sebagai berikut:
 - 1) *Method Void* merupakan method yang tidak dapat mengembalikan nilai. Sedangkan,
 - 2) *Method Non-Void* merupakan method yang dapat memberikan nilai. Contoh: `Public Static Void main (String agrs [])`.
- e. *Encapsulation* merupakan suatu pembungkus yang digunakan melindungi dan menjaga proses dari suatu program, sehingga tidak dapat diintervensi oleh program yang lain.
- f. *Inheritance* merupakan sebuah pewarisan, dimana dalam sebuah class dalam program dapat mewarisi atribut dan method dari class-class yang lain.
- g. *Polymorism* merupakan sebuah aksi yang dapat memungkinkan seorang pemrogram dapat menyampaikan pesan keluar dari hirarki onjeknya. Kemudian, objek tersebut memberikan tanggapan terhadap pesan sesuai dengan sifat dari masing-masing objek.

3. *Modifer pada Java*

Modifer adalah sebuah ijin akses yang digunakan untuk penggunaan suatu atribut atau method pada package dan class. Dalam java terdapat 4 jenis modifer yaitu, public, private, protected, dan no modifer.

- a. *Public class* menunjukkan bahwa method ataupun attribute yang memiliki akses modifer public dapat diakses dari manapun dan kapanpun.
- b. *Protected* menunjukkan bahwa atribut dan method yang memiliki akses modifer protected dapat diakses oleh class, package dan class turunan yang sama (subclass).
- c. *Private* menunjukkan bahwa atribut dan method memiliki akses default, dimana atribut dan method tersebut dapat diakses oleh class yang sama.
- d. *No modifer* menunjukkan bahwa atribut dan method memiliki akses yang default, dimana atribut dan method dapat diakses dari kelas yang sama dan package yang sama.

PERTEMUAN 2

TIPE DATA, VARIABEL DAN KONSTANTA

A. Tujuan Pembelajaran

1. Mahasiswa mampu memahami dan menjelaskan konsep dasar media transmisi data
2. Mahasiswa mampu memahami, mengidentifikasi, dan menjelaskan jenis-jenis media transmisi data

B. Uraian Materi

1. Tipe Data

Tipe data dalam sebuah pemrograman berfungsi untuk menentukan nilai yang akan dimasukkan ke dalam sebuah variabel. Selain itu, tipe data juga digunakan sebagai penyimpanan, karena untuk setiap tipe data memiliki besaran memori yang berbeda-beda. Sehingga, tipe data merupakan hal yang sangat krusial dalam sebuah pemrograman, yang akan memengaruhi proses berjalannya kode program yang telah dibuat. Tipe data dibagi menjadi dua jenis sebagai berikut [3]:

a. Tipe Data Sederhana (Simple Type)

Tipe data sederhana atau tipe data primitif merupakan sebuah tipe data dasar yang digunakan untuk menyimpan satu nilai pada satu variabel. Tipe data sederhana dibagi menjadi 3 kelompok sebagai berikut:

1) Kelompok *Numeric*

Kelompok data numeric merupakan tipe data yang berhubungan dengan angka. Kelompok numeric ini dibagi menjadi dua tipe sebagai berikut:

✓ Tipe Data Integer

Tipe data integer merupakan tipe data yang digunakan untuk mendeklarasikan atau menampung nilai bilangan bulat. Tipe data ini memiliki beberapa jenis tipe data yang digunakan untuk memberikan batasan pada penggunaan ruang penyimpanan agar lebih efisien. Tipe data integer dapat dilihat pada Tabel 1.

Tabel 1. Tipe Data Integer

Tipe Data	Ukuran Tempat	Rentang Nilai
Byte	1 Byte	0 s / d +255
Shortint	1Byte	-28 s / d +127
Integer	2 Bytes	-32768 s / d 32767
Word	2 Bytes	0 s / d 65535
Longint	4 Bytes	2147483648 s / d 2147483647

✓ Tipe Data Real

Tipe data real adalah sebuah tipe data yang digunakan untuk mendeklarasikan atau menyimpan bilangan – bilangan pecahan. Tipe data real dibagi menjadi beberapa jenis sebagai berikut:

Tabel 2. Tipe Data Real

Tipe Data	Ukuran Tempat	Rentang Nilai
Real	6 Byte	2.9×10^{-39} s / d 1.7×10^{38}
Single	4 Byte	1.5×10^{45} s / d 3.4×10^{38}
Double	8 Bytes	5.0×10^{-324} s / d 1.7×10^{308}
Extended	10 Bytes	3.4×10^{-4932} s / d 10^{4932}
Comp	8 Bytes	-9.2×10^{18} s / d 9.2×10^{18}

2) Kelompok *Char* (Karakter)

Tipe data *char* merupakan tipe data yang digunakan untuk menyimpan satu karakter atau menggunakan 1 bytes (1 bytes = 8 bit). Aturan dalam memberikan nilai pada tipe data *char* adalah dengan memberikan tanda petik pada karakternya seperti berikut: 'a', 'b', 'c', '1' dan seterusnya. Tipe data *char* dapat digunakan untuk menyimpan huruf (a, b, c, d, dll), angka(1,2,3,4, dll), tanda baca (.,?!, dll), dan dapat menyimpan karakter khusus (@#\$%^&*) [4].

3) Kelompok Boolean

Tipe data boolean merupakan tipe data yang digunakan untuk menentukan tipe data logika yang hanya dapat menyimpan keterangan benar (*true*) dan salah (*false*). Dalam pemrograman tipe data boolean disimbolkan dengan 0 (*false*) dan 1 (*true*). Tipe data boolean dibagi menjadi 3 jenis sebagai berikut:

Tabel 3. Tipe Data Boolean

Tipe Data	Ukuran Tempat
Boolean	1 Byte
Wordbool	2 Byte
Longbool	3 Byte

b. Tipe data Terstruktur

Tipe data terstruktur merupakan kumpulan dari tipe data array, string, set, record, dan file. Salah satu tipe data terstruktur adalah string, String merupakan tipe data yang digunakan untuk menyimpan teks yang memiliki kesamaan dengan tipe data *char*. Namun, pada tipe

data string memiliki panjang karakter, sehingga mampu menampung jenis teks. Aturan yang digunakan untuk penulisan tipe data string yaitu dengan memberikan petik pada teks yang akan digunakan dalam sebuah pemrograman.

2. Variabel dan Konstanta

Variabel merupakan fungsi yang digunakan untuk menyimpan data dalam sebuah program. Selain itu, variabel didefinisikan sebagai simbol berupa kata, huruf, atau kombinasi teks dan angka yang digunakan untuk menampung data sementara. Dalam variabel, disaat program berjalan nilai dalam suatu variabel dapat diubah sesuai dengan kebutuhan. Dalam pendeklarasian variabel dalam sebuah program diawali dengan simbol \$ [5]. Variabel dibagi menjadi 3 jenis sebagai berikut:

a. Variabel Lokal

Variabel lokal merupakan sebuah variabel yang dideklarasikan dalam sebuah fungsi, yang hanya dapat digunakan dalam fungsi tersebut.

b. Variabel Global

Variabel Global merupakan sebuah variabel yang dideklarasikan atau ditentukan di luar fungsi, jika ingin menggunkan ke dalam fungsi maka kita dapat mendeklarasikannya ke dalam fungsi dengan ketentuan penambahan kata global.

c. Variabel *Static*

Variabel static merupakan sebuah variabel yang digunakan untuk menampung atau menyimpan hasil.

Sedangkan, konstanta merupakan sebuah variabel yang nilai datanya bersifat final, dan tidak dapat dirubah selama program dijalankan. Aturan dalam pendeklarasian nilai konstanta tidak perlu menggunakan simbol \$. Sebagai contoh pendeklarasian variabel dan konstanta sebagai berikut:

- ✓ $A = 5$, A merupakan variabel, dan 5 merupakan nilai data yang disimpan pada variabel A
- ✓ $B = 3$, B merupakan variabel, dan 3 merupakan nilai data yang disimpan pada variabel B
- ✓ $C = A+B$, C merupakan variabel, sedangkan A dan B merupakan operand variabel, tanda + merupakan operator yang digunakan untuk menjumlahkan variabel A dan Variabel B.

PERTEMUAN 3

OPERATOR DAN STRUKTUR PERCABANGAN

A. Tujuan Pembelajaran

1. Mahasiswa dapat memahami dan menjelaskan pengertian encoding menurut para ahli
2. Mahasiswa dapat memahami dan menjelaskan Tujuan dari Encoding
3. Mahasiswa dapat memahami dan mendefinisikan karakter encoding
4. Mahasiswa mampu memahami dan menjelaskan prinsip transmisi data

B. Uraian Materi

1. Operator

Operator merupakan sebuah ekspresi yang digunakan untuk mengolah / mengubah nilai dari suatu variabel. Dalam pemrograman java terdapat beberapa operator yang digunakan yaitu, operator aritmatika, bitwise, assignment, kondisi, instanceof, bit shift, operator logika, dan operator relasi. Berikut pembahasan beberapa operator:

a. Operator Aritmatika

Operator aritmatika merupakan operator yang digunakan melakukan operasi penambahan, pengurangan, perkalian dan pembagian. Dalam pemrograman java juga mengenal operator modulus atau bisa disebut dengan pembagi biasa. Sebagai contoh, $10 \% 5$ sisa 0 atau $10 \% 3$ sisa 1. Dalam operator aritmatika, simbol + tidak hanya digunakan sebagai penjumlahan, namun dapat digunakan sebagai penggabungan string (String concatenation) [6].

b. Operator Kondisi

Operator kondisi merupakan sebuah ternary operator. Ternary operator merupakan sebuah operator yang mempunyai tiga operand. Dalam pemrograman java, operator kondisi digunakan untuk mengevaluasi suatu kondisi yang memiliki nilai benar (true) atau salah (false), selanjutnya menginputkan suatu nilai ke dalam suatu variabel. Operator kondisi ini hampir mirip dengan if, akan tetapi operator kondisi digunakan untuk meng-assign nilai ke dalam sebuah variabel berdasarkan suatu kondisi. Dalam program, operator kondisi menggunakan simbol tanda tanya (?) dan titik dua (:).

c. Operator Logika

Operator logika merupakan sebuah operator yang digunakan untuk mendefinisikan logika dari program yang dibuat. Dalam sebuah pemrograman operator logika merupakan hal yang sangat penting, hal ini dikarenakan pembuatan rangkaian logika yang rapi dan mudah dipahami akan memudahkan dalam memahami alur dari sebuah program. Dalam operator logika terdapat 6 operator yang dapat digunakan sebagai berikut:

- 1) Operator dan (&) merupakan sebuah operator yang digunakan untuk operator bitwise, jika operand yang digunakan memiliki tipe angka.
 - 2) Operator Or (|) merupakan sebuah operator yang digunakan untuk operator bitwise, jika operand yang digunakan memiliki tipe angka
 - 3) Operator xor (^) merupakan sebuah operator yang digunakan untuk operator bitwise jika operand yang digunakan memiliki tipe angka.
 - 4) Operator logika negasi (!)
 - 5) Operator singkat (&&)
 - 6) Operator singkat or (||)
- d. Operator Relasi

Operator relasi merupakan sebuah operator yang digunakan untuk menghasilkan nilai boolean (true atau false). Dalam program, operator ini sering digunakan untuk mengecek kondisi dan ditempatkan pada fungsi percabangan (if). Operator relasi dibagi menjadi enam jenis sebagai berikut:

- 1) Lebih kecil (<)
- 2) Lebih besar (>)
- 3) Lebih kecil sama dengan (<=)
- 4) Lebih besar sama dengan (>=)
- 5) Perbandingan (==)
- 6) Tidak sebanding (!=)

Tabel 4. Jenis Operator dan Fungsinya

Operator	Lambang	Keterangan
Aritmatika	+	Penjumlahan
	-	Pengurangan
	*	Perkalian
	/	Pembagian
	%	% = modulus (sisa bagi); 5 % 2 = 1
Assignment	=	X+=Y; sama artinya :X=X+Y
Increment	++i	I=I+1; (naikkan I sebelum operasi)
	i++	I=I+1; (naikkan I setelah operasi)
Decrement	--i	I=I-1; (turunkan I sebelum operasi)
	i--	I=I-1; (turunkan I setelah operasi)
Relasional	>	<i>Great the</i> (lebih besar)
	<	<i>Less then</i> (lebih kecil)

	>=	<i>Great or equal</i> (lebih besar atau sama dengan)
	<=	<i>Less or equal</i> (lebih kecil atau sama dengan)
	= =	<i>Equal</i> (sama dengan)
	! =	<i>Not equal</i> (tidak sama dengan)
Logika	&&	Logika AND
		Logika OR
	!	Logika NOT
	&	Boolean Logika AND
		Boolean Logika inclusive OR
	^	Boolean Logika exclusive OR
Kondisi	?:	Operator ternary, yang membawa 3 argumen.

2. Struktur Percabangan

Percabangan merupakan suatu perintah dalam sebuah program yang memungkinkan suatu perintah (Pernyataan) akan dieksekusi jika sebuah kondisi terpenuhi atau tidak terpenuhi. Dalam percabangan jika syarat kondisi terpenuhi, maka program akan dijalankan. Namun, jika sebuah kondisi tidak terpenuhi, maka akan beralih pada perintah program yang lainnya. Fungsi percabangan dalam pemrograman digunakan untuk menentukan langkah kerja intruksi. Berikut jenis-jenis percabangan:

a. Percabangan If

Fungsi percabangan ini digunakan untuk mengantisipasi sebuah kondisi yang terjadi saat program berjalan dan untuk menentukan tindakan apa yang dilakukan jika kondisi terpenuhi. Selain itu, percabangan if juga didefinisikan sebagai sebuah pernyataan yang berfungsi untuk mengambil keputusan dari dua kemungkinan. Dalam pemrograman java, if dapat berdiri sendiri atau juga dapat digunakan berdampingan dengan else. Adapun jenis-jenis dari fungsi If sebagai berikut:

1) If Tunggal

Adapun struktur dari percabangan if tunggal sebagai berikut:

```
if(kondisi) {
    ` pernyataan yang dijalankan, bila kondisi benar `
}
```

2) IF Else IF

Percabangan If else If merupakan suatu percabangan yang berfungsi untuk mencari kondisi benar dan mencari kondisi salah dalam suatu program. Adapun struktur fungsi pernyataan IF sebagai berikut:


```
if(kondisi){  
    pernyataan yang dijalankan, bila kondisi benar  
} else{  
    blok pernyataan yang dijalankan, bila kondisi salah}
```

3) Fungsi Switch Case

Fungsi switch case merupakan fungsi percabangan yang digunakan untuk mengevaluasi nilai dari sebuah variabel yang memiliki banyak kemungkinan penyelesaian masalah. Fungsi percabangan iswitch case hampir memiliki fungsi yang sama dengan if-else-if.

Dalam pemrograman java, percabangan yang terlalu banyak seringkali membingungkan pembaca source code. Sehingga, java menyediakan sebuah instruksi switch case untuk memudahkan pembacaar terhadap alur source code program. Fungsi percabangan switch case didesain untuk menggantikan fungsi if-else-if, namun fungsi ini memiliki beberpa batasan sebagai berikut:

- ✓ Data harus bertipe integer (int) atau karakter (char) atau string dapat digunakan.
- ✓ Range data yang diperiksa bernilai 0 s/d 255.

Berikut struktur dari fungsi switch case :

```
switch(ekspresi){  
    case nilaiSatu: Pernyataan 1  
    break; case nilaiDua:  
    Pernyataan2  
    break;  
    ...  
    default: PernyataanN;  
}
```

PERTEMUAN 4

FUNGSI PERULANGAN

A. Tujuan Pembelajaran

1. Mahasiswa dapat memahami dan menjelaskan konsep dasar Perulangan
2. Mahasiswa dapat menjelaskan dan menerapkan berbagai jenis perulangan.

B. Uraian Materi

1. Definisi Perulangan

Perulangan atau *looping* merupakan suatu perintah program yang digunakan untuk mengulang beberapa baris kode program atau perintah sampai pada batas yang telah ditentukan.

Dalam perulangan terdapat tiga komponen yang harus ada, sebagai berikut:

- ✓ Kondisi pada saat awal perulangan
- ✓ Kode program atau perintah yang akan diulang
- ✓ Kapan perulangan tersebut akan berhenti

a. Fungsi Perulangan (*Looping*)

Dalam pemrograman, perulangan memiliki fungsi sebagai berikut:

- 1) Memberikan efisiensi dalam eksekusi program
- 2) Memperpendek kode program
- 3) Contoh Kasus:

- ✓ Proses yang dilakukan secara berulang, seperti menuliskan huruf A – Z atau angka 1 -500, dapat dituliskan dengan beberapa baris koding saja, tanpa menulis satu perstatu.
- ✓ Misalnya kita ingin menuliskan “Hallo Pemrograman Java” sebanyak 10 kali, dengan konsep perulangan kita tidak perlu menuliskan program sebanyak 10 kali, tapi cukup satu program untuk menampilkan 10 kali “Hallo Pemrograman Java”.

b. Dua Jenis Proses Perulangan

Secara umum proses looping atau perulangan dibedakan menjadi dua jenis, yaitu:

- 1) Looping yang jumlah kali perulangannya sudah diketahui terlebih dahulu.
- 2) Looping yang jumlah kali perulangannya belum diketahui kapan perulangan tersebut akan berhenti.
- 3) Contoh studi kasus:
 - ✓ Perulangan yang sudah diketahui jumlah perulangannya: kita ingin menampilkan “Hallo Word” Sebanyak 10 Kali.

- ✓ Perulangan yang belum diketahui jumlah perulangannya: Kita makan nasi, prosesnya kita lakukan secara berulang-ulang, proses berhentinya kapan? akan berhenti jika muncul suatu kondisi kita “Kenyang”.

c. Struktur Utama pada Perulangan

Adapun struktur utama pada perulangan sebagai berikut:

- 1) Kondisi perulangan, merupakan suatu kondisi yang harus dipenuhi agar perulangan dapat dilakukan, biasanya menggunakan ekspresi boolean.
- 2) Bagian perintah coding yang akan dilakukan secara berulang.

2. Jenis-Jenis Perulangan

1) Perulangan For

Perulangan for merupakan sebuah perulangan yang sudah diketahui berapa kali kita akan melakukan perulangan pada suatu program. Dalam perulangan for dibutuhkan tiga struktur kondisi yang sebagai berikut:

- ✓ Inisialisasi merupakan satu atau lebih statement yang hanya dieksekusi satu kali pada awal perulangan, setelah itu tidak akan dieksekusi lagi. Contoh: $I = 1$ (Inisialisasi variabel)
- ✓ Kondisi merupakan sebuah situasi yang menentukan apakah bagian *statement (s)* akan dikerjakan atau tidak. Jika hasil kondisinya bernilai True, maka *statement(s)* akan dieksekusi atau ditampilkan. Namun, jika kondisi bernilai false, maka *statement(s)* tidak akan dieksekusi atau ditampilkan.
- ✓ Post-Increment merupakan sebuah ekspresi yang dilakukan diakhir setiap statement. Contoh simbol: $i++$ ($I+ = 1$), $++I$ ($i+ = 1$, $I = 1 + i$).

Berikut contoh kasus perulangan for:

- ✓ Kita ingin mencetak nama kita sebanyak 5 – 10 kali
- ✓ Kita ingin menampilkan nilai 7 sebanyak 5 kali
- ✓ Kita ingin menampilkan nilai < 50 .

Berikut sintaks perulangan for:

```
for (inisialisasi; kondisi; penaikan_penurunan)
{
    pernyataan;
}
```

2) Perulangan *While*

Perulangan *while* merupakan jenis perulangan yang digunakan untuk mengulang suatu pernyataan, namun tidak diketahui atau tidak dapat dipastikan berapa banyak perulangan yang akan dilakukan. Berikut alur program dari perulangan *while*:

- ✓ Perulangan *while* dimulai dengan mengevaluasi kondisi apakah benar atau tidak. Jika benar maka statement akan dijalankan.
- ✓ Pada perulangan *while*, suatu kondisi dicek atau dieksekusi terlebih dahulu sebelum masuk ke blok perulangan.
- ✓ Jika kondisi awal nilainya *false*, maka pernyataan di dalam *while* tidak akan dieksekusi atau dijalankan.
- ✓ Contoh kasus: Kita akan membuat program untuk mengacak buah-buahan, kita tidak akan tahu berapa kali nanti user akan mencoba, apakah 2, 3, 4 atau 10 kali percobaan sampai hasil yang diinginkan ditemukan.
- ✓ Adapun struktur sintaks dari perulangan *while* sebagai berikut:

```
while (kondisi)
{
    pernyataan
}
```

- Dalam perulangan *while* kondisi digunakan sebagai syarat untuk mengentikan proses perulangan
- Pernyataan (statement) digunakan untuk menampilkan instruksi program yang akan diulang saat kondisi bernilai benar.

3) Perulangan *do While*

Perulangan *do while* merupakan perulangan yang hampir sama dengan perulangan *while*. Perbedaannya hanya terdapat pada saat pengecekan kondisi perulangan yang dilakukan. Berikut perbedaan dari perulangan *while* dan *do while*:

- ✓ Pada perulangan *while* kondisi perulangan diperiksa di awal sehingga apa bila kondisi salah maka perulangan tidak akan dilakukan sama sekali
- ✓ Sedangkan, pada perulangan *do while* lokasi pengecekan perulangan dilakukan di akhir. Maka perintah looping akan melakukan satu kali perintah sebelum mengecek perulangan di belakang.
- ✓ Jadi, dalam perulangan *do while* walaupun kondisinya salah atau tidak terpenuhi, maka perulangan tetap akan dilakukan sekali.
- ✓ Berikut struktur dari perulangan *do while*:

```
do {  
    pernyataan  
} while (kondisi);
```

3. Jenis Perulangan dalam Perulangan

1) *Nested Loop*

Nested loop merupakan suatu perulangan yang terjadi ketika terdapat *loop* di dalam *loop*. Sehingga, jumlah perulangan yang terjadi lebih banyak. Karena, terjadi perkalian *loop* yang didalam dan *loop* yang di luar.

2) Interupsi Break

Interupsi Break merupakan suatu kondisi yang digunakan untuk mengentikan proses perulangan. Jika terjadi *Break*, maka perulangan akan berhenti dan alur program keluar dari loop yang sedang dijalankan.

3) *Interupsi Continue*

Interupsi continue merupakan suatu kondisi yang digunakan dalam perulangan, akan menyebabkan program dihentikan, namun tidak keluar dari perulangan. *Continue* dapat menyebabkan program dilanjutkan kembali dengan iterasi berikutnya.

PERTEMUAN 5

INPUT DAN OUTPUT

A. Tujuan Pembelajaran

1. Mahasiswa dapat memahami dan menjelaskan konsep dasar dari input dan output
2. Mahasiswa dapat menjelaskan dan menerapkan input Scanner, BufferedReader, dan JOptionPane.

B. Uraian Materi

1. Input/Output

Input merupakan sebuah masukan yang berisi perintah atau instruksi yang diinputkan melalui keyboard ke komputer untuk diproses melalui bahasa pemrograman tertentu, yang kemudian akan mengeluarkan hasil atau yang disebut dengan output program.

Dalam pemrograman java operasi I/O menggunakan fungsi *streams*. *Streams* merupakan sebuah abstraksi dari suatu yang digunakan untuk menulis, menghasilkan, membaca atau mendapatkan informasi. Secara umum *streams* dalam java dibagi menjadi dua, sebagai berikut:

- a. *Byte streams*, fungsi ini digunakan untuk menangani operasi I/O yang menggunakan *character*.
- b. *Character streams*, digunakan untuk mengatasi operasi I/O yang menggunakan *character*.

Berikut beberapa streams, yang dapat digunakan untuk input pada pemrograman Java:

1) *Byte Stream*

Dalam pemrograman, java menyediakan dua class abstrak yang menjadi superclass tertinggi untuk Byte Stream, yaitu:

- ✓ *InputStream* untuk membaca input
- ✓ *OutputStream* untuk menuliskan output.

2) *Character Stream*

Character stream dalam java terdiri dari dua class abstrak yang menjadi superclass tertinggi, yaitu:

- ✓ *Reader* untuk membaca input
- ✓ *Writer* untuk menuliskan output

3) *Variabel Stream Standar*

Secara umum, pemrograman java sudah menyediakan tiga variabel streams yang dapat langsung digunakan, yang merupakan member dari public static dari class System, yaitu:

- ✓ *System.out* merupakan output stream standar, dimana secara default outputnya adalah console

- ✓ System.in merupakan input stream standar, dimana secara default inputnya adalah dari keyboard.
- ✓ System.err merupakan output stream yang digunakan untuk mencetak pesan yang salah pada console (Default).

2. Jenis-Jenis Input/Output

a. Input Scanner

Input scanner merupakan class yang menyediakan fungsi-fungsi untuk mengambil input dari keyboard. Berikut langkah-langkah untuk menggunakan fungsi Scanner pada pemrograman Java:

- 1) Meng-import class Scanner yang terdapat pada java dengan code `import java.util.Scanner;`
- 2) Membuat objek referensi sebagai media untuk menginputkan data, dengan code `(Scanner objekReferensi = new Scanner(System.in))`
- 3) Memanggil method khusus untuk melakukan inputan data melalui objek referensi.

b. Input BufferedReader

Input BufferedReader merupakan sebuah fungsi dalam pemrograman java yang digunakan untuk membaca variabel yang diinputkan dalam program. BufferedReader dalam java berfungsi untuk membaca sebuah bilangan atau karakter. Berikut langkah-langkah menggunakan input BufferedReader dalam Java:

- 1) Meng-import Class InputStreamReader, IOException dan BufferedReader.
- 2) Menuliskan objek referensi dengan code `BufferedReader objekReferensi = new BufferedReader (new InputStreamReader(System.in))`
- 3) Memanggil method `readLine()` melalui objek referensi
- 4) Menuliskan poin dua dan tiga ke dalam blok try-catch.

Berikut tabel konversi tipe data pada BufferedReader:

Tabel 5. Tabel Konversi Tipe Data pada BufferedReader

Class	Konversi ke Tipe Data	Pemakaian
Boolean	boolean	<code>Boolean.parseBoolean(..);</code>
Byte	byte	<code>Byte.parseByte(...);</code>
Character	char	<code>String.charAt(<index>);</code>
Short	short	<code>Short.parseShort(...);</code>
Integer	int	<code>Integer.parseInt(...);</code>
Long	long	<code>Long.parseLong(...);</code>
Float	float	<code>Float.parseFloat(...);</code>

Double	double	Double.parseDouble(...);
--------	--------	--------------------------

c. Input/Output JOptionPane

Fungsi JOptionPane merupakan sebuah kelas dalam java yang menyediakan jendela dialog. JOptionPane pada java dapat digunakan untuk mengambil input, menampilkan informasi, menampilkan pesan error, dan menampilkan dialog konfirmasi. Dalam JOptionPane terdapat empat fungsi yang digunakan yaitu:

- ✓ ShowConfirmDialog() digunakan untuk menampilkan dialog konfirmasi
- ✓ showInputDialog() digunakan untuk menampilkan dialog input
- ✓ ShowMessageDialog() digunakan untuk menampilkan pesan/informasi
- ✓ ShowOptionDialog() digunakan untuk menampilkan dialog pilihan.

Berikut langkah-langkah dalam membuat JOptionPane pada Java:

- ✓ Meng-import class JOptionPane dengan code Import javaz.swing.JOptionPane

PERTEMUAN 6

ARRAY SATU DIMENSI DAN MULTI DIMENSI

A. Tujuan Pembelajaran

Setelah mengikuti perkuliahan ini mahasiswa diharapkan dapat memahami konsep Array, Pendeklarasian Array, Pengaksesan Elemen pada Array, Konsep Array Satu Dimensi, dan Konsep Array Dua Dimensi dalam pemrograman, serta mampu mengimplementasikannya ke dalam program sederhana.

B. Uraian Materi

1. Pengenalan Array

Array merupakan sebuah tipe data yang digunakan untuk menampung atau menyimpan lebih dari satu nilai dalam program. Dalam bahasa pemrograman java, Array didefinisikan sebagai suatu wadah yang menyediakan ruang penyimpanan pada sejumlah item yang memiliki tipe yang sama.

Fungsi array dalam program digunakan untuk mengelompokkan berbagai informasi yang saling berkaitan. Item dalam array selalu diberikan penomoran dimulai dari nol hingga nilai maksimum tertentu, yang ditentukan saat program array dibuat.

2. Pendeklarasian Array

Dalam program array perlu dideklarasikan seperti sebuah variabel. Proses pendeklarasian array, diawali dengan membuat daftar tipe data yang akan digunakan, kemudian diikuti dengan sepasang tanda kurung, dan diikuti oleh nama identifiernya (nama array). Contoh:

```
Int [] coba; atau bisa ditulis seperti ini: int coba [];
```

- ✓ Syntax deklarasi array

```
tipeData nama[];
```

- ✓ Syntax inisialisasi array

```
tipeData nama[] = new tipeData[jumlahElemen];
```

```
tipeData nama[] = {nilai1, nilai2, ...};
```

Setelah mendeklarasikan array pada program, selanjutnya menentukan berapa panjang array yang akan kita gunakan dengan sebuah konstruktor. Dalam java proses penentuan panjang array disebut dengan *instantiation* (inisiasi). Contoh:

```
Int coba []; // Deklarasi array
```

```
Coba= new int [50]; //Inisiasi panjang array atau panjang objek. Atau bisa juga ditulis seperti berikut:
```

```
Int coba[] = new int [50];
```

3. Pengaksesan Elemen pada Array

Indeks merupakan angka yang digunakan untuk menyatakan urutan dari elemen pada variabel array. Penomoran indeks pada variabel array akan selalu diawali dengan nilai nol. Oleh karena itu, nomor indeks pada elemen terakhir N-1, dimana N merupakan jumlah total dari elemen.

Sebagai catatan penting, dalam array jika suatu tipe data sudah dideklarasikan dan dikonstruksikan, maka nilai yang tersimpan dalam array akan diinisialisasi sebagai Nol. Sehingga, jika programmer menggunakan tipe data string, maka array tidak akan diinisialisasi menjadi string kosong. Oleh karena itu, disarankan untuk membuat atau mendeklarasikan string array secara eksplisit. Berikut contoh akses elemen array:

```
Public class Contoh_Array {
    Public static void main (String[] args) {
        Int coba [] = new int [50];
        For(int I = 0; i<50; i++) {
            System.out.println(coba[i]);}}}
```

4. Array Satu Dimensi

Array satu dimensi merupakan sebuah deklarasi array pada satu variabel yang memiliki tipe yang sama. Pada array satu dimensi hanya terdapat satu baris dengan sejumlah kolom. Setiap elemen pada array mempunyai tipe data yang sama dan memiliki index array yang dimulai dengan nol. Pada bahasa pemrograman java, String merupakan array satu dimensi dari character. Berikut contoh tampilan array satu dimensi:

```
Public class Array_SatuDimensi{
    Public static void main (String[] args){
        //Mendeklarasikan nama variabel bertipe array dengan tipe data int
        // dan menentukan jumlah pada elemen array yang akan ditampilkan.
        Int bulan[] = new int [10];
        For(int I = 0; i<10; i++){
            System.out.println (bulan[i]);
        }}}
```

Pada contoh program array satu dimensi di atas dapat kita lihat bahwa class array satu dimensi memiliki 10 indeks, dimana isi 10 indeks ditampilkan menggunakan perulangan for.

5. Array Multidimensi

Array multidimensi merupakan array yang memiliki nilai ukuran lebih dari dua. Array multi dimensi merupakan gabungan dari beberapa array satu dimensi. Pada program array multi dimensi akan terdiri dari baris (row) dan kolom (column), sehingga array multidimensi disebut juga dengan matriks

Pada beberapa kondisi dalam array multidimensi dibutuhkan penulisan variabel array menggunakan nomor indeks dua bilangan, seperti pada matriks. Baris dan kolom dalam matriks merupakan nilai bilangan. Cara pendeklarasian array multidimensi sama halnya dengan pendeklarasian pada array satu dimensi. Akan tetapi, pada array dua dimensi atau multidimensi terdapat indeks array dalam array yang pertama. Pada kurung siku yang pertama digunakan untuk mendeklarasikan elemen baris dan kurung siku yang kedua digunakan untuk mendeklarasikan elemen kolom. Berikut contoh pendeklarasian array dua dimensi atau array multidimensi:

```
Int array_multi [][] = new int [4][3];
```

Contoh program array multidimensi:

```
Public class Array_Multi{
    Public static void main (String [] args){
    Int A[] [] = new int [3][4];
    Int baris, kolom, isi;
    Isi = 1;
    For (baris = 0; baris<3; baris++){
        For(kolom = 0; kolom<4; kolom++){
        A[baris][kolom] = isi;
        Isi++;}}
    For (baris = 0; baris<3; baris++){
        For(kolom = 0; kolom<4; kolom++){
        System.out.print(A[baris] [kolom]);}
        System.out.println();}}}
```

PERTEMUAN 7 PEMROGRAMAN MODULAR

A. Tujuan Pembelajaran

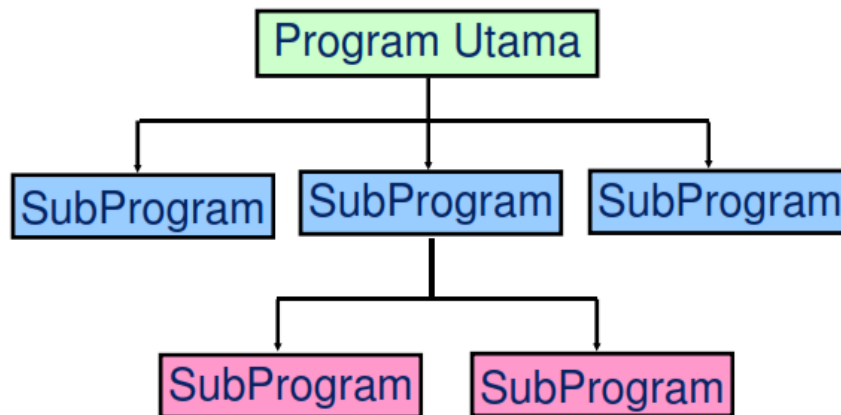
1. Mahasiswa dapat memahami Konsep Pemrograman Modular
2. Mahasiswa dapat menjelaskan fungsi dan prosedur dalam Pemrograman Modular
3. Mahasiswa dapat memahami dan menjelaskan perbedaan fungsi dan prosedur
4. Mahasiswa dapat mengimplementasikan fungsi dan prosedur dalam Java

B. Uraian Materi

1. Konsep Pemrograman Modular

a. Definsi Pemrograman Modular

Pemrograman Modular merupakan sebuah teknik pemrograman yang membuat program besar menjadi beberapa bagian program yang lebih kecil, sehingga dapat mudah dipahami dan digunakan kembali. Definisi lain dari pemrograman modular adalah sebuah metode program yang digunakan untuk menyelesaikan masalah dengan memecah masalah tersebut menjadi sub-sub masalah yang biasa disebut dengan subprogram. Pemrograman modular dapat memungkinkan kita memanggil kembali subprogram yang sudah didefinisikan saat diperlukan. Berikut skema dari pemrograman modular:



Gambar 1. Skema Pemrograman Modular

b. Sifat dan Syarat Modul

Pemrograman modular memiliki sifat dan syarat yang harus dipenuhi, sebagai berikut:

- Dalam pemrograman modular, satu modul hanya mengerjakan satu pekerjaan yang sesuai dengan pemecahan masalah.

- Sifat dari modul adalah merahasiakan sesuatu. Modul tidak akan memberitahukan jika sedang melakukan pekerjaan, namun langsung memberikan hasil sesuai yang diinginkan.
- Dalam pemrograman modular, setiap modul harus bersifat *independent*. Dimana, modul tidak terikat pada modul yang lain kecuali pada input dan outputnya.

c. Kelebihan Pemrograman Modular

- Program lebih pendek
- Mudah dibaca dan dimengerti
- Mudah didokumentasi
- Mengurangi kesalahan dan mudah mencari kesalahan
- Teknik pemrograman modular dapat membuat program besar dan kompleks
- Teknik pemrograman modular membantu dalam membuat algoritma
- Pemrograman modular memudahkan pihak lain memahami program yang kita buat

d. Kelemahan Pemrograman Modular

- Terdapat kesulitan dalam implementasi, karena tidak adanya formulasi khusus yang dapat dijadikan sebagai acuan dalam membuat algoritma dan program
- Diharuskan untuk banyak berlatih agar dapat menguasai pemrograman modular

2. Fungsi dalam Pemrograman Modular

a. Definisi Fungsi

Fungsi merupakan sebuah kumpulan instruksi/perintah/program yang kemudian dikelompokkan menjadi satu. Fungsi terletak terpisah dari program yang menggunakan fungsi tersebut, memiliki nama yang unik, dan digunakan untuk mengerjakan satu tujuan tertentu. Dalam pemrograman fungsi memiliki nilai kembalian.

b. Kelebihan dari Fungsi

- Fungsi dapat melakukan pendekatan top-down dan divide-and-conquer:
 - ✓ Top-down: Menelusuri program sangat mudah
 - ✓ Divide-and-conquer: Dimana program besar akan dipisah menjadi program yang lebih kecil
- Kode program menjadi lebih pendek, mudah dibaca dan dipahami.
- Program yang dibuat dapat dikerjakan oleh beberapa orang. Sehingga program dapat selesai lebih cepat, dan memudahkan koordinasi.
- Menggunakan alur dan logika yang sederhana, sehingga mudah mencari kesalahan pada program

- Kesalahan pada kode program dapat dialokasikan menjadi satu modul.
- Dapat melakukan modifikasi program pada suatu modul tertentu, sehingga tidak akan mengganggu program secara keseluruhan atau program utama.
- Memudahkan dalam dokumentasi
- Fungsi dalam program digunakan untuk mengantisipasi penulisan program yang dilakukan secara berulang-ulang.
- Reusability: Dimana fungsi dapat digunakan oleh program atau fungsi lain.

c. Rancangan Pembuatan Fungsi

Hal yang perlu diperhatikan dalam membuaata sebuah fungsi, sebagai berikut:

- Memiliki data yang akan dijadikan sebagai inputan
- Mengetahui informasi apa yang akan diberikan oleh fungsi yang dibuat kepemanggilnya
- Mengetahui algoritma yang akan digunakan untuk mengolah data menjadi sebuah informasi.

d. Fungsi Void

- Fungsi void dalam pemrograman sering juga disebut dengan prosedur.
- Fungsi void merupakan fungsi yang tidak dapat mengembalikan nilai keluaran.
- Ciri dari fungsi void adalah:
 - ✓ Tidak terdapat keyword return
 - ✓ Tidak terdapat tipe data pada deklarasi fungsi
 - ✓ Hanya menggunakan keyword void
- Fungsi void tidak dapat langsung ditampilkan hasilnya
- Fungsi void tidak memiliki nilai kembalian
- Keyword void akan dapat digunakan jika fungsi tersebut tidak memiliki paramter apapun.

e. Fungsi Non-void

- Fungsi non-void dapat disebut juga dengan function
- Fungsi non-void dapat mengembalikan nila keluaran yang berasal dari proses fungsi tersebut.
- Ciri dari fungsi non-void adalah:
 - ✓ Pada fungsi non-vid terdapat keyword return
 - ✓ Terdapat tipe data yang mengawali deklarasi fungsi
 - ✓ Tidak terdapat keyword void.

- Memiliki nilai kembalian
- Fungsi non-void, hasilnya dapat langsung ditampilkan

Berikut struktur dasar dari fungsi dalam pemrograman:

```
Static TipeDataKembalian namaFungsi() {
//Statement atau kode fungsi
}
```

Pada Program tersebut:

- `Static` dapat diartikan sebagai tindakan dalam membuat fungsi yang dapat dipanggil tanpa membuat instansiasi objek
- `TipeDataKembalian` merupakan tipe data dari sebuah nilai yang akan dikembalikan saat fungsi dieksekusi.
- `namaFungsi()` merupakan nama dari fungsinya yang ditulis dengan huruf kecil pada awal nama. Berikut contoh fungsi:

```
static void ucapSalam() {
System.out.println("Selamat Pagi");
}
```

Berikut cara memanggil fungsi dalam fungsi `main`:

```
Public static void main (String[] args){
ucapSalam();
}
```

Adapaun contoh struktur fungsi dengan parameter sebagai berikut:

```
Static TipeData namaFungsi(TipeData namaParameter, TipeData
nama ParameterLain) {
// kode fungsi
}
```

3. Prosedur dalam Pemrograman Modular

Prosedur merupakan sebuah instruksi yang diberikan nama untuk melakukan tujuan tertentu dalam sebuah program. Sama halnya pada fungsi, prosedur melakukan pekerjaan dengan

mekanisme panggilan-kembali (call-return mechanism). Akan tetapi tidak mengembalikan nilai papaun. Berikut parameter yang digunakan dalam prosedur:

- Dalam prosedur terdapat paramter input dan paramter output.
- Parameter boleh kosong (tidak terdapat parameter input dan output)
- Parameter yang ditulis pada bagian definisi disebut dengan parameter formal. Sedangkan parameter yang ditulis pada pemanggilan disebut dengan parameter aktual

4. Manfaat Menggunakan Fungsi dan Prosedur

- Mampu memecahkan program yang kompleks dan besar menjadi subprogram yang lebih kecil
- Dapat meningkatkan kemampuan dalam menganalisis kesalahan. Apabila terjadi kesalahan, maka programmer cukup mencari fungsi dan prosedur yang digunakan, tanpa harus melihat pada seluruh program.

5. Perbedaan Fungsi dan Prosedur dalam Pemrograman Modular

Fungsi dan prosedur dalam pemrograman digunakan untuk mengelompokkan program besar menjadi program yang lebih kecil, sehingga dapat mengolah keseluruhan program dengan baik. Kesamaan dari fungsi dan prosedur adalah sama-sama sebuah modul dalam program. Namun, juga memiliki perbedaan yaitu Fungsi digunakan untuk merepresentasikan modul yang akan mengembalikan nilai kepada yang memanggilnya, sedangkan prosedur digunakan untuk merepresentasikan modul yang menjalankan suatu perintah tanpa memberikan pengembalian nilai pada pemanggilanya.

6. Implementasi Fungsi dan Prosedur dalam Java

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.ArrayList;

public class FungsiProsedur{
    static ArrayList listBuah = new ArrayList();
    static boolean isRunning = true;
    static InputStreamReader inputStreamReader = new
    InputStreamReader (System.in);
    static BufferedReader input = new
    BufferedReader (inputStreamReader);

    static void showMenu() throws IOException{
```



```

System.out.println("====MENU====");
System.out.println("[1] Show All Buah");
System.out.println("[2] Insert Buah");
System.out.println("[3] Edit Buah");
System.out.println("[4] Delete Buah");
System.out.print("PILIH MENU");

        Int                selectedMenu                =
Integer.valueOf(input.readLine());

Switch(selectedMenu) {
    Case 1:
        showAllBuah();
        break;
    case 2:
        insertBuah();
        break;
    case 3:
        editBuah();
        break;
    case 4:
        deleteBuah();
        break;
    case 5:
        System.exit(0);
        Break;
    default:
        System.out.println("Pilihan salah!");
}}
Static void showAllBuah() {

```

PERTEMUAN 8
UJIAN TENGAH SEMESTER (UTS)

PERTEMUAN 9-10

PEMROGRAMAN BERORIENTASI OBJEK

A. Tujuan Pembelajaran

1. Mahasiswa dapat memahami *Definisi Object oriented Programming (OOP)*
2. Mahasiswa dapat memahami ciri-ciri dari OOP (Encapsulation, Inheritance, Polymorphism)
3. Mahasiswa dapat memahami, kelas dan Objek
4. Mahasiswa dapat memahami Constructor OOP, Kata Kunci This, dan Kata Kunci Final dalam OOP.

B. Uraian Materi

1. Definisi Object Oriented Programming (OOP)

Object Oriented Programming (OOP) adalah sebuah metode pemrograman yang berorientasi pada objek. Dalam pemrograman tujuan dari OOP adalah untuk mengatasi lemahnya pendekatan pemrograman yang bersifat konvensional. Adapun prinsip dari Pemrograman OOP sebagai berikut:

a. *Encapsulation*

Encapsulation merupakan sebuah konsep terkait pengikatan terhadap data atau sebuah metode yang disatukan (dikapsulkan) menjadi suatu unit data. Prinsip encapsulation dalam pemrograman dapat mempermudah pembacaan kode pemrograman, mempermudah penggunaan suatu objek dari sebuah kelas.

b. *Inheritance*

Inheritance adalah prinsip dalam OOP yang digunakan untuk membuat sebuah class baru dengan fungsi turunan atau mirip dengan fungsi sebelumnya. Dalam pemrograman prinsip inheritance menggunakan sistem hirarki, dimana semakin jauh turunan atau subclassnya, maka semakin dekat kemiripan antar kelas dengan fungsinya.

c. *Polymorphism*

Polymorphism merupakan prinsip dalam OOP yang digunakan untuk memproses suatu pesan atau data lebih dari satu bentuk. Prinsip ini menjadi ciri utama dari OOP, jika dalam pemrograman tidak memiliki prinsip ini, maka tidak dapat dikatakan sebagai pemrograman berorientasi pada objek. Dalam polymorphism sebuah objek yang berbeda-beda dapat diakses melalui interface yang sama.

d. *Abstraction*

Abstraction merupakan prinsip OOP yang memungkinkan developer memerintahkan fungsi, tanpa harus mengetahui bagaimana fungsi tersebut bekerja. Dalam prinsip abstraction dapat

menyembunyikan informasi detail latar belakang dan hanya akan menampilkan informasi yang diperlukan.

2. Kelebihan Object Oriented Programming (OOP)

- a. *Paralle Development*, dimana masing-masing programmer dapat membangun *class* sendiri.
- b. *Reusable* dapat menggunakan class yang sudah pernah dibuat sebelumnya
- c. *Scalability* bertujuan untuk mempermudah kebutuhan dari program yang lebih rumit.

3. Kekurangan Object oriented Programming (OOP)

- a. Tidak Efisien

Dalam pemrograman menggunakan konsep OOP akan memakan daya pada CPU yang digunakan.

- b. Membutuhkan Manajemen Data yang Lebih Ketat

Dalam OOP akan muncul beberapa kode yang baru, apa bila terdapat kode-kode yang kurang berfungsi.

- c. Kemungkinan terjadi Duplikasi

Dengan mudahnya untuk dapat menggunkan program yang sudah ada sebelumnya, menjadikan pemrograman OOP seperti duplikasi program.

4. Struktur bagian dalam OOP

- a. *Class* merupakan sebuah kumpulan data dan fungsi pada suatu unit dengan tujuan tertentu
- b. *Objek* merupakan suatu wadah data dan fungsi yang menjadi satu ke dalam sebuah program komputer. Dalam OOP, objek merupakan konsep dasar dari modularitas dan struktur dari sebuah program komputer.
- c. *Constructor* merupakan sebuah metode khusus yang akan dieksekusi saat membuat objek (*instance*). Dalam pemrograman OOP digunakan untuk menyiapkan suatu data untuk class.
- d. Kata Kunci *This* merupakan sebuah perintah khusus yang digunakan untuk mengakses object dalam program. Dalam OOP perintah *this* digunakan untuk menghindari kesalahan akses terhadap property dan argument method yang memiliki nama yang sama.
- e. Kata Kunci *Final* merupakan sebuah perintah khusus yang digunakan untuk mencegah agar sebuah class, property, atau method tidak akan di timpa oleh nilainya. Dalam pemrograman keyword *final* masuk dalam kelompok *modifier*, dimana sebuah perintah yang akan memodifikasi perilaku default dari sebuah class, property, atay method.

PERTEMUAN 11-12

PEMROGRAMAN BERORIENTASI OBJEK LANJUTAN

A. Tujuan Pembelajaran

1. Mahasiswa dapat memahami Definisi Modifier dan Karakteristik Modifier
2. Mahasiswa dapat memahami dan menjelaskan Overload, Inner Class, Override, dan Abstraksi dalam OOP

B. Uraian Materi

1. Deifinisi Modifier

Modifier merupakan sebuah fitur penting dalam OOP yang digunakan untuk melakukan data hiding (Menyembunyikan Data). Fitur *Modifier* ini memungkinkan programmer dapat mengatur hak akses dan member class yang akan digunakan, agar tidak semua perintah masuk untuk mengakses atau dapat dikatakan tidak dapat diakses secara langsung. Fitur modifier memiliki tiga tipe akses sebagai berikut:

- a. *Public* merupakan sebuah label yang digunakan untuk menentukan sifat akses pada semua member yang mengikutinya, sehingga public ini memiliki sifat yang dapat diakses dari manapun.
- b. *Private* merupakan sebuah label yang digunakan untuk menentukan sifat akses ke semua member yang mengikutinya, sehingga private ini memiliki sifat yang tidak dapat diakses dari manapun kecuali melalui *friend function* dan berasal dari *class* itu sendiri.
- c. *Protected* merupakan sebuah label yang digunakan untuk menentukan sifat akses semua member yang mengikutinya, sehingga memiliki sifat tidak dapat diakses dari dalam class maupun anak class (*drived class*).

2. Bagian dari Modifier

- a. *Overload* merupakan aktivitas dalam membuat beberapa method yang memiliki nama yang sama, namun memiliki jumlah atau tipe parameter yang berbeda.
- b. *Inner Class* merupakan class yang berada dalam class lagi, dimana class adalah sebuah kerangka model/blueprint yang difungsikan untuk menempatkan atribut seperti variabel, method, konstruktor, dll.
- c. *Override* merupakan sebuah metode yang digunakan pada kelas induk atau superclass untuk mendefinisikan kembali class turunan atau subclass dengan menggunakan metode dan parameter yang sama. Berikut contoh program yang menggunakan metode override pada Java:

```
public class Induk{
public void panggilAku(){
    System.out.println("");
    System.out.println("Hallo, ini induk yang dipanggil");
}
}
public class Anak{
//method sama dengan method induk atau override
public void panggilAku(){
    System.out.println("");
    System.out.println("Hallo, ini anak yang dipanggil");
}
}
public class DemoOverride{
public static void main(String args[]){
    Anak a= new Anak();
    a.panggilAku();
}
}
```

REFERENSI

- [1] H. Ahmadian, H. Mizuardy, and K. Ar, *Pemrograman Visual dengan Java*. Unimal Press, Universitas Malikussaleh, 2017.
- [2] F. N. Hasanah and C. Taurusta, *Modul Pemrograman Berorientasi Objek, Pertama.*, vol. 4. UMSIDA PRESS, Universitas Muhammadiyah Sidoarjo, 2019.
- [3] G. Widodo, “DASAR-DASAR PEMROGRAMAN JAVA,” in *Teknik Informatika*, 2011, pp. 1–76.
- [4] A. S. Wahyu Eko Susanto, *Logika Dan Algoritma Untuk Pemula*. 2020.
- [5] Y. Rahmawati and U. Indahyanti, *Buku Ajar Pemrograman Dasar Menggunakan Visual Basic. Net 2013*, Pertama. UMUSIDA PRSS, Universitas Muhammadiyah Sidoarjo, 2020.
- [6] T. Setiadi, A. Tarmuji, Supriyanto, and A. Prahara, “Petunjuk Praktikum Pemrograman Berorientasi Objek,” in *Teknik Informatika*, U. A. D. Laboratorium Teknik Informatika, Ed. Program Studi Teknik Informatika, 2020, p. 65.