**MODUL**

**ALGORITMA DAN PEMROGRAMAN**

****

**Disusun oleh :**

**Femi Dwi Astuti, S.Kom, M.Cs.**

**SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER**

**AKAKOM**

**YOGYAKARTA**

**2019**

# KATA PENGANTAR

# DAFTAR ISI

[KATA PENGANTAR ii](#_Toc15905388)

[DAFTAR ISI iii](#_Toc15905389)

[MODUL 1 NAMA POKOK BAHASAN 1](#_Toc15905390)

# MODUL 1 PENGANTAR ALGORITMA

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| Mahasiswa dapat membuat algoritma sederhana | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Text Editor (Microsoft Word, Notepad, dll) | |

|  |  |
| --- | --- |
|  | DASAR TEORI |

1. **Pengertian Algoritma**

‘Algoritma’ diambil dari nama belakang seorang tokoh matematikawan Persia bernama Muhammad ibn Musa al-Khuwarizmi (lahir tahun 730an, meninggal antara tahun 835 dan 850). Algoritma merupakan metode umum yang digunakan untuk menyelesaikan kasus-kasus tertentu. Algoritma adalah langkah – langkah logis tertentu untuk menyelesaikan suatu masalah. Guna algoritma adalah untuk membantu seseorang dalam menyelesaikan suatu masalah berdasarkan pada pola pikirnya masing-masing. Dalam menuliskan algoritma, dapat digunakan bahasa natural atau menggunakan notasi matematika, sehingga masih belum dapat dijalankan pada komputer.

Dalam kehidupan sehari-hari, kita sudah melakukan penyusunan algoritma untuk menyelesaikan permasalahan atau tantangan yang dihadapi. Misalnya algoritma memasak nasi, algoritma berangkat kuliah, algoritma masuk ke dalam suatu ruangan, algoritma menggunakan komputer, dll. Algoritma dapat digunakan untuk merepresentasikan suatu urutan kejadian secara logis dan dapat diterapkan di semua kejadian sehari-hari. Jadi algoritma adalah jembatan untuk mempermudah pemahaman alur kerja suatu proses.

1. **Ciri – ciri algoritma:**
2. Ada masukan (*input*).
3. Ada proses.
4. Ada keluaran (*output*).
5. Memiliki perintah yang jelas dan tidak membingungkan (*ambigu*).
6. Harus mempunyai tanda akhir (*stopping role)*

.

1. **Sifat algoritma :**
2. Tidak menggunakan simbol atau sintaks dari suatu bahasa pemrograman.
3. Tidak tergantung pada suatu bahasa pemrograman.
4. Notasi-notasinya dapat digunakan untuk seluruh bahasa manapun.

|  |  |
| --- | --- |
|  | PRAKTIK |

1. Buatlah algoritma untuk memasak telur dadar

|  |
| --- |
| 1. Siapkan mangkok untuk mengocok telur 2. Pecah dan masukkan telur ke dalam mangkok 3. Masukkan garam ke dalam mangkok 4. Kocok telur dalam mangkok 5. Nyalakan kompor 6. Siapkan wajan dan panaskan dikompor 7. Tuangkan minyak ke dalam wajan 8. Tuangkan telur yang sudah dikocok kedalam wajan 9. Masak telur hingga matang 10. Angkat telur 11. Matikan kompor |

1. Buatlah algoritma menentukan apakah suatu bilangan termasuk genap atau ganjil

|  |
| --- |
| 1. Masukkan sebuah bilangan bulat 2. Bagi bilangan yang sudah dimasukkan dengan angka 2 3. Jika menghasilkan sisa pembagian ( 0 ) maka bilangan tersebut genap 4. Jika menghasilkan sisa pembagian bukan (0) maka bilangan tersebut ganjil |

1. Buatlah algoritma untuk menghitung luas segitiga

|  |
| --- |
| 1. Masukkan alas 2. Masukkan tinggi 3. Hitung LuasSegitiga = 0,5\*alas\*tinggi 4. Tampilkan LuasSegitiga |

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Buat algoritma dalam kalimat deklaratif untuk :
2. Memasak mie instan
3. Menghitung luas lingkaran
4. Mendaftar mahasiswa baru di STMIK AKAKOM

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Buatlah algoritma dalam kalimat deklaratif untuk melakukan daftar ulang di STMIK AKAKOM

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, hal (3-5)
2. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, hal (21-37, 42-43)

# MODUL 2 PSEUDOCODE DAN FLOWCHART

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat membuat algoritma dalam berbagai bentuk (pseudocode, flowchart) 2. Mahasiswa dapat menggunakan alat bantu visual seperti flowgorithm | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Text Editor (Microsoft Word, Notepad, dll) 2. Flowgorithm | |

|  |  |
| --- | --- |
|  | DASAR TEORI |

1. **Penyajian Algoritma**

Algoritma biasa disajikan dalam tiga bentuk yaitu :

1. Kalimat deklaratif
2. Pseudocode
3. Flowchart
4. **Pseudocode**

*Pseudocode* adalah bentuk informal untuk mendeskripsikan algoritma yang mendekati struktur bahasa pemrograman tertentu. Tidak ada suatu standar untuk menyusun algoritma menggunakan pseudocode. Oleh karena *pseudocode* lebih cocok digunakan untuk menyusun algoritma dengan kasus yang besar dan kompleks, maka sangat dianjurkan kepada programer pemula untuk mulai menggunakan pseudocode dalam menyelesaikan masalah. Tujuan dari penggunaan *pseudocode* adalah supaya :

1. lebih mudah dibaca oleh manusia
2. lebih mudah untuk dipahami
3. lebih mudah dalam menuangkan ide/hasil pemikiran

Seorang programer yang ingin menerapkan algoritma tertentu, terutama yang kompleks atau algoritma baru, biasanya akan memulainya dengan membuat deskripsi dalam bentuk *pseudocode*. Setelah *pseudocode* tersebut jadi, maka langkah selanjutnya hanya tinggal menterjemahkannya ke bahasa pemrograman tertentu. *Pseudocode* ini biasanya disusun dalam bentuk yang terstruktur dengan pendekatan sekuensial (berurutan) dari atas ke bawah.

Contoh Pseudocode :

Pseudo code untuk menghitung jumlah dua buah bilangan :

Deklarasi

A, B : integer {input}

Jumlah : integer {output}

Deskripsi

Masukkan (A, B)

Jumlah 🡨 A+B

Tampilkan (Jumlah)

1. **Flowchart**

Dalam membuat algoritma, diperlukan suatu mekanisme atau alat bantu untuk menuangkan hasil pemikiran mengenai langkah-langkah penyelesaian masalah yang sistematis dan terurut. Pada dasarnya untuk bisa menyusun solusi diperlukan kemampuan *problem-solving* yang baik. Oleh karena itu, sebagai sarana untuk melatih kemampuan tersebut terdapat sebuah *tool* (alat) yang dapat digunakan, yakni *flowchart*.

Secara formal, *flowchart* didefinisikan sebagai skema penggambaran dari algoritma. Tabel berikut menampilkan simbol-simbol yang digunakan dalam menyusun *flowchart*.

Tabel 1. Simbol-simbol dalam *flowchart*

|  |  |
| --- | --- |
|  | ***Terminator***  Sebagai simbol ’START’ atau ’END’ untuk memulai atau mengakhiri flowchart. |
|  | ***Input/Output***  Digunakan untuk menuliskan proses menerima data atau mengeluarkan data |
|  | **Proses**  Digunakan untuk menuliskan proses yang diperlukan, misalnya operasi aritmatika |
|  | ***Conditional / Decision***  Digunakan untuk menyatakan proses yang membutuhkan keputusan |
|  | ***Preparation***  Digunakan untuk memberikan nilai awal |
|  | ***Arrow***  Sebagai penunjuk arah dan alur proses |
|  | ***Connector (On-page)***  Digunakan untuk menyatukan beberapa arrow |
|  | ***Connector (Off-page)***  Digunakan untuk menghubungkan flowchart yang harus digambarkan pada halaman yang berbeda. Biasanya pada simbol ini diberi label sebagai penanda, misalnya angka 1 atau A. |
|  | ***Display***  Digunakan untuk menampilkan data ke *monitor* |

Keuntungan menggunakan *flowchart* adalah penggunaan diagram untuk menggambarkan tahapan proses, sehingga lebih mudah dilihat dan dipahami. Namun demikian, flowchart juga memiliki kelemahan, yakni jika digunakan untuk menggambarkan proses atau algoritma untuk skala kasus yang besar, maka akan dibutuhkan banyak kertas

1. **Flowgorithm**

***Flowgorithm*** merupakan aplikasi gratis yang dapat digunakan untuk mempelajari algoritma pemrograman komputer melalui diagram alir (flowchart). Aplikasi ini dikembangkan oleh **Devin Cook** di **Universitas Negeri Sacramento** pada Tahun 2017 lalu. Flowgorithm dapat diunduh di <http://www.flowgorithm.org/>

Contoh penyajian algoritma dalam kalimat deklaratif :

1. Masukkan nilai panjang dan simpan dalam variabel **Panjang**
2. Masukkan nilai lebar dan simpan dalam variabel**Lebar**
3. Hitung Luas dengan rumus: **Luas = Panjang \* Lebar**
4. Tampilkan **Luas** Persegi Panjang

Contoh penyajian algoritma dalam flowchart dengan flowgorithm :

Jika kita menggambar flowchart di *Flowgorithm* kita bisa membuatnya menjadi seperti ini:

|  |
| --- |
| [https://4.bp.blogspot.com/-1hWr74rVD8c/XBEeq41X6EI/AAAAAAAACNM/gikB8Z88L8s9BsFjGG4mh5QXfj4muSKEgCLcBGAs/s400/MainFlowchartPP.png](https://4.bp.blogspot.com/-1hWr74rVD8c/XBEeq41X6EI/AAAAAAAACNM/gikB8Z88L8s9BsFjGG4mh5QXfj4muSKEgCLcBGAs/s1600/MainFlowchartPP.png) |
| Gambar 1. Flowchart di Flowgorithm |

***Flowchart*** ini jika dijalankan maka akan ada beberapa pertanyaan tentang panjang dan lebar persegi panjang dan kita dapat menginput nilainya. Hasil lengkapnya terlihat pada gambar di bawah ini.

|  |
| --- |
| [https://1.bp.blogspot.com/-FBlV_ubyadg/XBEfgb4XeuI/AAAAAAAACNU/-1_70Elpsus4nb9H9l4Kn-_SeS_BYKn4wCLcBGAs/s400/tampilan_run_flowchart.jpg](https://1.bp.blogspot.com/-FBlV_ubyadg/XBEfgb4XeuI/AAAAAAAACNU/-1_70Elpsus4nb9H9l4Kn-_SeS_BYKn4wCLcBGAs/s1600/tampilan_run_flowchart.jpg) |
| Gambar 2. Tampilan ketika Flowchart dijalankan |

Flowgorithm juga dapat mengenerate flowchart menjadi source code program dan mendukung banyak bahasa pemrograman. Untuk bahasa pemrograman yang didukung diantaranya terlihat pada gambar berikut ini:

|  |
| --- |
| [https://2.bp.blogspot.com/-cjjV4_BsAAA/XBEgRoK1XOI/AAAAAAAACNc/O124svm_EbIVfTWRvYt6nL3CxgeeJJHIACLcBGAs/s400/sc_fg_dukung.jpg](https://2.bp.blogspot.com/-cjjV4_BsAAA/XBEgRoK1XOI/AAAAAAAACNc/O124svm_EbIVfTWRvYt6nL3CxgeeJJHIACLcBGAs/s1600/sc_fg_dukung.jpg) |
| Gambar 3. Bahasa Pemrograman yang di dukung Flowgorithm |

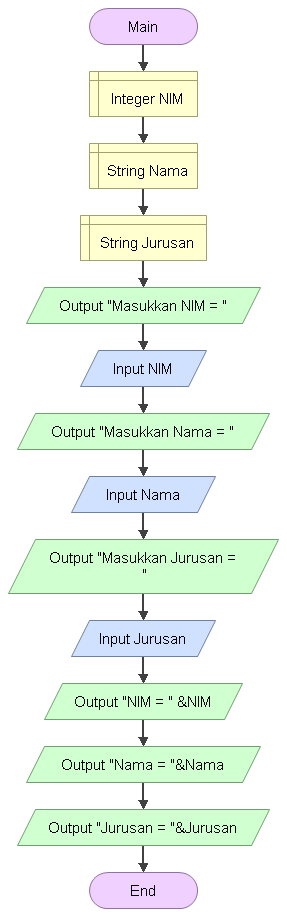
|  |  |
| --- | --- |
|  | PRAKTIK |

1. Buat pseudocode dan flowchart untuk menampilkan data pribadi

Pseudocode

|  |
| --- |
| Deklarasi  NIM, Nama, Jurusan : String {input}  NIM, Nama, Jurusan : String {output}  Deskripsi  Masukkan (NIM, Nama, Jurusan)  Tampilkan (NIM, Nama, Jurusan) |

Flowchart

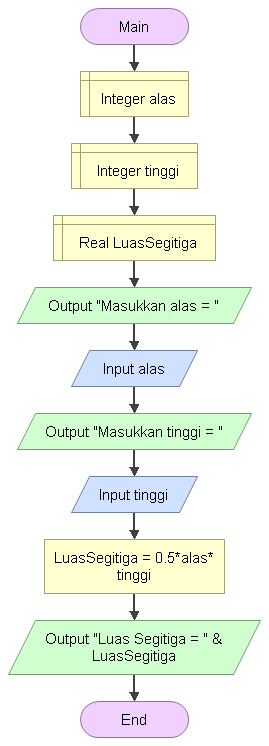


1. Sajikan praktik modul 1 nomor 3 (menghitung luas segitiga) dalam bentuk pseudocode dan flowchart

Pseudocode

|  |
| --- |
| Deklarasi  alas, tinggi : integer {input}  LuasSegitiga : float {output}  Deskripsi  Masukkan (alas, tinggi)  LuasSegitiga 🡨 0,5\*alas\*tinggi  Tampilkan (LuasSegitiga) |

Flowchart



|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Modifikasi praktik 1 dengan menambahkan data alamat dan nomor telepon
2. Modifikasi praktik 2 untuk menghitung luas persegi panjang

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Buat algoritma dan sajikan dalam bentuk pseudocode dan flowchart untuk menghitung angsuran pembayaran SPA dengan ketentuan :

Angsuran 1 : 40% dari SPA

Angsuran 2 : 30% dari SPA

Angsuran 3 : 20% dari SPA

Angsuran 4 : 10% dari SPA

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, hal (843,847)
2. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, hal (45-53)

# MODUL 3 TIPE DATA, VARIABEL, OPERATOR

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat mengimplementasikan berbagai macam tipe data 2. Mahasiswa dapat menggunakan variabel dan konstanta 3. Mahasiswa mampu menggunakan operator unary dan binary | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Text Editor (Microsoft Word, Notepad, dll) 2. Flowgorithm | |

|  |  |
| --- | --- |
|  | DASAR TEORI |

1. **Tipe Data**

Secara umum, tipe data terdiri dari :

1. Karakter

Himpunaan karakter terdiri dari huruf, digit maupun simbol-simbol lainya (termasuk spasi, karakter kontrol).

Contoh :

Huruf : A, a, B, b, C, c ...............

Digit : 0, 1, 2, 3, 4, 5 ...........

Simbol dan lainnya : \_ - + \* dan sebagainya

1. Numeric
2. **byte**: Tipe data byte adalah 8-bit integer bertanda *two's complement*. Tipe ini mempunyai nilai minimum -128 dan nilai maksimumnya adalah 127.
3. **short**: Tipe data short merupakan integer 16 bit *two's complement* yang mempunyai nilai minimum -32,768 dan nilai maksimum 32,767.
4. **int**: Tipe data int adalah integer 32 bit *two's complement*. Dia mempunyai nilai minimum -2,147,483,648 sedangkan nilai maksimumnya adalah 2,147,483,647 (inclusive).
5. **long**: Tipe data long adalah integer 64 bit *two's complement*. Nilai minimumnya adalah -9,223,372,036,854,775,808 sedangkan nilai maksimumnya adalah 9,223,372,036,854,775,807 (inclusive). Gunakan tipe data ini pada saat anda memerlukan jangkau nilai yang lebih besar daripada yang bisa disajikan oleh int.

* **float**: Tipe data float merupakan *single-precision 32-bit IEEE 754 floating point*.
* **double**: Tipe data double adalah *double-precision 64-bit IEEE 754 floating point*.

Tabel 2.1. Batas nilai floating point

|  |  |  |
| --- | --- | --- |
| **Parameter** | **float** | **double** |
| N | 24 | 53 |
| K | 8 | 11 |
| E*max* | +127 | +1023 |
| E*min* | -126 | -1022 |

1. **boolean: Tipe data boolean hanya mempunyai dua nilai yang mungkin, yaitu true dan false. Gunakan tipe data ini untuk flag-flag sederhana untuk menjejak/menelusuri kondisi true atau false. Tipe data ini merepresentasikan satu bit informasi, tetapi ukurannya tidak didefinisikan dengan tepat.**
2. **char: Tipe data char adalah karakter Unicode 16 bit. Tipe data ini mempunyai nilai minimum '\u0000' (atau 0) dan nilai maksimum '\uffff' (atau 65,535).**
3. **Variabel dan konstanta**

Variabel digunakan dalam program untuk menyimpan suatu nilai, dan nilai yang ada padanya dapat diubah selama ekseskusi berlangsung.

Konstanta adalah nilai yang tetap.

Contoh :

float jumlah;

jumlah = 10;

Konstanta bernama

Hal ini dapat dilakukan dengan menggunakan kata kunci const.

Contoh :

const float PHI = 3.14;

1. **Operator**

Operator adalah simbol khusus yang menyajikan operasi khusus pada satu, dua, atau tiga operand dan kemudian mengembalikan hasilnya. Operator-operator tersebut digunakan untuk membentuk ekspresi. Sebuah konstanta atau entitas dengan sendirinya merupakan ekspresi, demikian juga kombinasi dari konstanta dan variabel dengan operator. Suatu ekspresi yang diikuti dengan sebuah titik koma adalah sebuah pernyataan. Operator mempunyai sifat- sifat

Tabel 2.2 Sifat Operator

|  |  |  |
| --- | --- | --- |
| **Sifat** | **Keterangan** | **Contoh** |
| Unary | Operator yang hanya melibatkan 1 operand | -1 |
| Binary | Operator yang hanya melibatkan 2 operand | 1 + 2 |
| Tenary | Operator yang hanya melibatkan 3 operand | 1 + 2 \* 2 |

Pada saat kita akan menggunakan operator harus mengetahui terlebih dahulu operator yang mana yang mempunyai presedence yang lebih tinggi. Operator di dalam tabel 2.1. dituliskan sesuai dengan presedence ordernya. Semakin ke bawah, maka presedence-nya lebih rendah. Operator dengan presedence yang lebih tinggi dikerjakan lebih dulu dari pada operator dengan presedence yang lebih rendah. Operator yang ada di dalam baris yang sama mempunyai presedence yang sama. Pada saat operator dari presedence yang sama muncul di dalam ekspresi yang sama, harus diatur yang mana yang harus dikerjakan lebih dulu. Semua operator biner kecuali untuk operator pemberian dikerjakan dari kiri ke kanan. Operator pemberian dikerjakan dari kanan ke kiri.

**Tabel 2.3 Precedence Operator**

|  |  |
| --- | --- |
| **Operator** | **Precedence** |
| *Postfix* | expr++ expr-- |
| *Unary* | ++expr --expr +expr -expr ~ ! |
| Multiplikasi | \* / % |
| Aditif | + - |
| Pergeseran | << >> >>> |
| Relasional | < > <= >= instanceof |
| Persamaan | == != |
| bitwise AND | & |
| bitwise exclusive OR | ^ |
| bitwise inclusive OR | | |
| Logika AND | && |
| Logika OR | || |
| *Ternary* | ? : |
| Pemberian | = += -= \*= /= %= &= ^= |= <<= >>= >>>= |

**Operator Aritmatika**

Tabel 2.4 Operator Aritmatika

|  |  |
| --- | --- |
| **Operator** | **Keterangan** |
| + | Penjumlahan (tanda plus) |
| - | Pengurangan (tanda minus) |
| \* | Perkalian |
| / | Pembagian |
| % | Sisa Pembagian |

Operator \*, / dan % mempunyai prioritas yang sama, tetapi lebih tinggi daripada + atau -.

|  |  |
| --- | --- |
|  | PRAKTIK |

1. Buat algoritma sajikan dalam bentuk pseudocode untuk menghitung nilai A yang dieroleh dari A=B+C\*D

|  |
| --- |
| Deklarasi  B, C, D : integer {input}  A : integer {output}  Deskripsi  Deskripsi  Masukkan (B,C,D)  A 🡨 B+C\*D  Tampilkan (A) |

1. Modifikasi praktik 1 dimana C diperoleh dari B\*2, Buat pseudocodenya

|  |
| --- |
| Deklarasi  B, D : integer {input}  A, C : integer {output}  Deskripsi  Masukkan (B, D)  C 🡨 B\*2  A 🡨 B+C\*D  Tampilkan (A) |

1. Buat pseudocode dan flowchart untuk menghitung hasil modulo (sisa hasil bagi) dari dua buah bilangan

|  |
| --- |
| Deklarasi  Bil1, Bil2 : integer {input}  Hasil : integer {output}  Deskripsi  Masukkan (Bil1, Bil2)  Hasil 🡨 Bil1%Bil2  Tampilkan (Hasil) |

1. Buat pseudocode dan flowchart untuk menghitung hasil konversi dari meter ke centimeter dan dari meter ke inci.

|  |
| --- |
| Deklarasi  m : float {input}  cm, inci : float {output}  Deskripsi  Masukkan (m)  cm 🡨 m\*100  inci 🡨 m\*100/2,54  Tampilkan (cm)  Tampilkan (inci) |

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Modifikasi praktik 1 dengan membuat flowchartnya
2. Buat pseudocode dan flowchart untuk menjumlahkan, mengalikan, mengurangkan dan membagi dua buah bilangan

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Disampaikan oleh dosen pengampu dikelas

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, hal ((41-70,93-99))

# MODUL 4 PENGENALAN JAVA DAN SEKUENSIAL

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat menggunakan bahasa pemrograman JAVA 2. Mahasiswa dapat mengimplementasikan bahasa pemrograman untuk menyelesaikan masalah sederhana secara sekuensial | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Textpad 2. JDK 3. Flowgorithm | |
|  | DASAR TEORI |

1. **Pengenalan JAVA**

Java adalah bahasa pemrograman multi platform. Java tidak menyediakan IDE khusus seperti halnya bahasa pemrograman yang lain. Pemrogram bisa menggunakan IDE yang *support* ke Java, misalnya Netbeans, Eclips, TexPad, dan lain-lain. Editor teks bisa digunakan semisal Notepad. Jika editor yang digunakan tidak support Java, kita tinggal menyimpannya dalam ekstensi .java kemudian kompilasi dan menjalankannya menggunakan command prompt.

1. Pengenal (identifier)

Pengenal atau identifier adalah suatu nama yang bisa dipakai dalam pemrograman untuk menyatakan :

* variabel
* konstanta bernama
* tipe data
* fungsi
* label
* obyek

Contoh : noMhs; no\_Mhs;

1. Kata Kunci

Pengenal sistem yang mempunyai makna khusus bagi kompiler. Kegunaan dari golongan ini tidak dapat diubah.

Contoh :

case, char, const, do, else, for, return, void, while, dan lain-lain

1. Pernyataan Masukan Dan Keluaran Dalam Java

Dalam bahasa pemrograman Java, kita akan menggunakan pernyataan masukan dan keluaran. Pernyataan masukan adalah pernyataan untuk mendapatkan masukan dari keyboard. Sedangkan pernyataan keluaran adalah pernyataan untuk menampilkan sesuatu nilai ke layar.

Komponen keluaran

Untuk menampilkan ke layar secara tekstual, java mempunyai fasilitas output, yaitu dengan perintah

System.out.print( *hal yang akan ditampilkan, bisa lebih dari satu*);

untuk menampilkan ke layar dan setelah selesai tidak berpindah baris. Akibatnya perintah keluaran berikutnya akan ditampilkan pada baris yang sama. Dan perintah

System.out.println(*hal yang akan ditampilkan, bisa lebih dari satu*);

untuk menampilkan ke layar dan setelah selesai berpindah baris. Akibatnya perintah keluaran setelah itu akan ditampilkan pada baris berikutnya.

System.out.println(“Hasil penjumlahan “+hasil);

//menampilkan isi variabel hasil

Komponen masukan

Untuk memasukkan sebuah nilai ke variabel yang sudah didefinisikan digunakan kelas Scanner (kelas ini disediakan mulai Java versi 1.5).

**a. Input data bertipe Integer**

Untuk menginputkan data dengan tipe integer digunakan method nextInt() di dalam kelas Scanner.

**b. Input data bertipe String**

Untuk menginputkan data dengan tipe integer digunakan method next() atau nextLine() di dalam kelas Scanner.

**c. Input tipe yang lain**

Untuk input data dari keyboard dengan tipe yang lain dan disediakan oleh kelas Scannr adalah sebagai berikut

* nextBoolean : input tipe boolean (true atau false saja)
* nextShort : input tipe short integer
* nextLong : input tipe long integer
* nextFloat : input tipe float
* nextDouble : input tipe double

1. **Sekuensial**

Sekuensi adalah pemrograman sederhana yang hanya dapat memecahkan masalah-masalah yang sederhana. Masalah yang dapat diselesaikan dengan sekuensi saja biasanya hanya satu masalah kecil yang berdiri sendiri. Pembahasan sebelumnya, sejauh ini menggunakan prinsip sekuensi. Disamping itu, pada bagian ini kita juga akan mempelajari bagaimana membuat pseudocode sebelum implementasi ke bahasa pemrograman.

|  |  |
| --- | --- |
|  | PRAKTIK |

1. Buka textpad, Ketikkan program untuk praktik 1, 2, 3, 4 yang ada di pertemuan 3

**Praktik 1**

|  |
| --- |
| 1. public class Mod4\_1 2. { 3. public static void main(String args[]) 4. { 5. int B,C,D,A; 6. B=5; 7. C=3; 8. D=6; 9. A = B+C\*D; 10. System.out.println("Hasil A = "+A); 11. } 12. } |

1. Simpan dengan nama Mod4\_1.java
2. Tekan menu tools di text pad, kemudian pilih external tools, pilih compile java
3. Tekan menu tools di text pad, kemudian pilih external tools, pilih run java application
4. Amati outputnya, mengapa bisa demikian, jelaskan di laporan

**Praktik 2**

|  |
| --- |
| import java.util.Scanner;  public class Mod4\_3  {  public static void main(String args[])  {  Scanner masuk=new Scanner(System.in);  int Bil1,Bil2,Hasil;  System.out.print("Masukkan Bil 1 = ");  Bil1=masuk.nextInt();  System.out.print("Masukkan Bil 2 = ");  Bil2=masuk.nextInt();  Hasil=Bil1%Bil2;  System.out.println("Hasil = "+Hasil);  }  } |

Amati outputnya, mengapa bisa demikian, jelaskan di laporan

**Praktik 3**

|  |
| --- |
| import java.util.Scanner;  public class Mod4\_4  {  public static void main(String args[])  {  Scanner masuk=new Scanner(System.in);  int m;  float cm,inci;  System.out.print("Masukkan meter = ");  m=masuk.nextInt();  cm=m\*100;  inci=m\*100/2.54f;  System.out.println("Hasil konversi cm = "+cm);  System.out.println("Hasil konversi inci = "+inci);  }  } |

Amati outputnya, mengapa bisa demikian, jelaskan di laporan

**Praktik 4**

Modifikasi praktik 1, dimana nilai B, C dan D dapat dimasukkan oleh user, jalankan dan amati outputnya, jelaskan di laporan

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Modifikasi praktik 1 dimana C diperoleh dari B\*2
2. Buat program untuk menampilkan biodata dimana data dimasukkan oleh user. Data yang perlu disimpan dan ditampilkan adalah NIM, nama, tempat lahir, tanggal lahir, usia, no hp, e-mail, jurusan, alamat
3. Buat pseudocode dan program untuk menghitung luas lingkaran

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

* 1. Buat flowchart dan program untuk menghitung gaji dimana total gaji terdiri dari gaji pokok + tunjangan kinerja yang diperoleh dari 20% gaji pokok + tunjangan lain diperoleh dari 10% tunjangan kinerja
  2. Buat pseudocode dan program untuk menghitung konversi dari celcius ke reamur dan celcius ke fahrenheit

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Deitel P., Dietel H., 2015, *Java How to Program Tenth edition*, Deitel & Associates, Inc., Prentice Hall, New Jersey, hal (9-10,13-15,32-42)
2. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, hal (15-18,33-36,71-76)
3. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, hal (55,108)

# MODUL 5 SELEKSI IF

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat mengimplementasikan konsep seleksi if untuk menyelesaikan kasus yang sederhana | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Textpad 2. JDK 3. Flowgorithm | |
|  | DASAR TEORI |

**Seleksi**

Seleksi digunakan untuk mengarahkan suatu proses itu berjalan. Seleksi adalah  
Suatu program untuk mengambil keputusan berdasarkan suatu kondisi. Seleksi di Java memiliki dua macam bentuk yaitu : Seleksi dengan **if...else** dan **switch...case**.  
Suatu program komputer yang dibuat menggunakan algoritma tertentu pada umumnya memiliki perintah-perintah yang hanya akan dikerjakan pada situasi dan kondisi tertentu. Ini sangat mirip dengan kondisi yang ada di dunia nyata. Di dunia nyata biasanya kita juga memiliki pilihan-pilihan tertentu untuk melakukan sesuatu (apa yang harus dilakukan/akan kita kerjakan). Sebagai contoh, misalkan kita mungkin memiliki suatu pilihan sebagai berikut :

***“Jika saya punya uang, maka saya akan membeli laptop”***

Dari pernyataan di atas, tentunya kita **hanya** akan membeli laptop jika (dan hanya jika) kita memang punya uang. Java (juga hampir semua bahasa pemrograman lainnnya) memiliki apa yang dinamakan sebagai **nilai kebenaran** (nilai boolean) yang nilainya benar (**TRUE**) atau salah (**FALSE**). Dengan demikian, dapat kita simpulkan dari pernyataan di atas, kata **hanya** akan membeli sepeda motor jika (dan hanya jika) pernyataan “Saya punya uang” bernilai **TRUE** (atau dengan kata lain “Saya benar-benar punya uang”); jika saya tidak punya uang, atau (dalam konteks ekspresi boolean) pernyataan “Saya punya uang” bernilai FALSE, saya tidak akan membeli laptop.  
**Macam seleksi if**

Pernyataan if dapat digunakan untuk mengambil suatu keputusan berdasarkan suatu kondisi.

Bentuk dari pernyataan ini ada tiga macam, yaitu :

* 1. **If**

Pernyataan if atau if tunggal ini hanya digunakan jika kondisinya hanya 1. Bentuk umumnya adalah :

|  |
| --- |
| if(kondisi)  {  Pernyataan;  } |

Keterangan :

* Kondisi digunakan untuk menentukan pengambilan keputusan
* Pernyataan, berisi perintah-perintah dan akan dijalankan jika kondisi bernilai benar, jika kondisi salah, tidak ada yang dikerjakan
  1. **If-Else**

Pernyataan if-else tidak berdiri sendiri tanpa adanya if, karena bentuk pernyataan else adalah merupakan suatu ingkaran dari pernyataan if. Pada dasarnya pernyataan if...else hanya dapat menangani 2 kemungkinan keadaan.  
Bentuk umumnya adalah :

|  |
| --- |
| if(kondisi)  {  Pernyataan1;  }  else  {  Pernyataan2;  } |

Keterangan :

* Kondisi digunakan untuk menentukan pengambilan keputusan
* Pernyataan1, berisi perintah-perintah dan akan dijalankan jika kondisi bernilai benar
* Pernyataan2, berisi perintah-perintah dan akan dijalankan jika kondisi bernilai salah
  1. **If-else if**

Pada dasarnya pernyataan if...else if dapat menangani lebih dari 2 kemungkinan keadaan/kondisi.

Bentuk umumnya adalah :

|  |
| --- |
| if(kondisi1)  {  Pernyataan1;  }  else if(kondisi2)  {  Pernyataan2;  }  else if(kondisi3)  {  Pernyataan3;  }  else  {  Pernyataan4;  } |

Keterangan :

* Kondisi1 digunakan untuk menentukan pengambilan keputusan
* Pernyataan1, berisi perintah-perintah dan akan dijalankan jika kondisi1 bernilai benar
* Pernyataan2, berisi perintah-perintah dan akan dijalankan jika kondisi2 bernilai benar
* Pernyataan3, berisi perintah-perintah dan akan dijalankan jika kondisi3 bernilai benar
* Pernyataan4, berisi perintah-perintah dan akan dijalankan jika kondisi1, kondisi2, kondisi3 bernilai salah.

**Operator Relasi**

Operator atau tanda operasi adalah suatu tanda atau simbol yang biasa dilibatkan dalam program untuk melakukan suatu operasi atau manipulasi suatu nilai. Operator relasi adalah operator penghubung yang berupa benar atau salah. Operator Relasi selalu menghasilkan tipe data boolean **true** atau **false**, operator ini sering digunakan untuk mengecek sebuah kondisi dan diletakkan pada percabangan if.

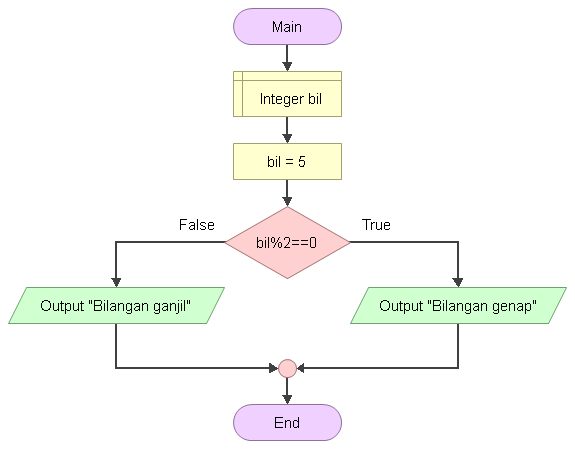
|  |  |  |
| --- | --- | --- |
| **Operator** | | **Arti** |
| < | | Lebih kecil |
| <= | | Lebih kecil atau sama dengan |
| > | | Lebih besar |
| >= | | Lebih besar atau sama dengan |
| == | | Sama dengan |
| != | | Tidak sama dengan |
|  | | PRAKTIK | | |

1. Buat program untuk menentukan apakah suatu bilangan merupakan bilangan 0

Program

|  |
| --- |
| public class modul5 {  public static void main(String[] args) {  int bil;  bil = 0;  if (bil==0) {  System.out.println("Bilangan 0");  }  }  } |

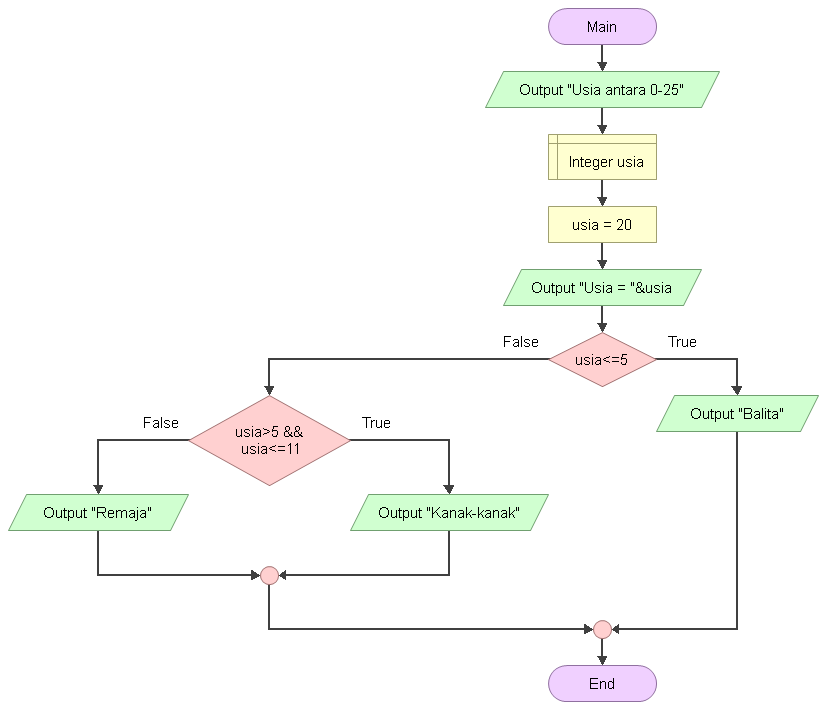
1. Jalankan dan amati outputnya
2. Modifikasi praktik 1 dengan bilangan dapat dimasukkan oleh user
3. Uji dan amati output jika Bilangan yang dimasukkan adalah 0
4. Uji dan amati output jika Bilangan yang dimasukkan selain 0
5. Buatlah flowchart dan program untuk menentukan apakah suatu bilangan termasuk genap atau ganjil



Program

|  |
| --- |
| class modul5\_1 {  public static void main(String[] args) {  int bil;  bil = 5;  if (bil % 2 == 0) {  System.out.println("Bilangan genap");  } else {  System.out.println("Bilangan ganjil");  }  }  } |

1. Jalankan dan amati outputnya
2. Modifikasi praktik 2 dengan bilangan dapat dimasukkan oleh user
3. Uji dan amati output jika Bilangan yang dimasukkan adalah 7
4. Uji dan amati output jika Bilangan yang dimasukkan selain 4
5. Buatlah flowchart dan ketikkan program untuk mengklasifikasikan usia seseorang antara usia 0 sampai 25.



Program

|  |
| --- |
| class modul5\_2 {  public static void main(String[] args) {  System.out.println("Usia antara 0-25");  int usia;  usia = 20;  System.out.println("Usia = " + usia);  if (usia <= 5) {  System.out.println("Balita");  } else {  if (usia > 5 && usia <= 11) {  System.out.println("Kanak-kanak");  } else {  System.out.println("Remaja");  }  }  }  } |

1. Jalankan dan amati outputnya
2. Modifikasi praktik 3 dengan menambahkan kondisi jika usia lebih dari 11 dan kurang dari atau sama dengan 25 disebut remaja dan jika lebih dari 25 disebut dewasa.

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Buat program untuk menentukan predikat kelulusan berdasarkan IPK mahasiswa dengan ketentuan sebagai berikut :

|  |  |
| --- | --- |
| **IPK** | **Predikat Kelulusan** |
| 2,75 – 2,99 | Memuaskan |
| 3,00 – 3,50 | Sangat memuaskan |
| 3,51 – 4,00 | Dengan pujian |

Catatan : IPK dapat dimasukkan oleh user

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Buat flowchart dan program untuk menentukan bilangan terbesar dan terkecil dari 3 buah bilangan
2. Buat program untuk menentukan apakah segitiga termasuk segitiga sama sisi, sama kaki atau segitiga sembarang berdasarkan panjang sisi-sisinya.

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Deitel P., Dietel H., 2015, *Java How to Program Tenth edition*, Deitel & Associates, Inc., Prentice Hall, New Jersey, hal (119-126,181-188)
2. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, hal (113-149)
3. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, hal (111-131)

# MODUL 6 SELEKSI SWITCH-CASE

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat mengimplementasikan konsep seleksi switch case untuk menyelesaikan kasus yang sederhana 2. Mahasiswa dapat mengubah dari satu bentuk seleksi ke bentuk seleksi yang lain | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Textpad 2. JDK 3. Flowgorithm | |

|  |  |
| --- | --- |
|  | DASAR TEORI |

1. Seleksi Switch-Case
2. Perbedaan If dengan Switch
3. Mengubah If menjadi Switch dan sebaliknya

|  |  |
| --- | --- |
|  | PRAKTIK |

1. Tuliskan praktik yang harus dikerjakan mahasiswa disini
2. ........

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Tuliskan soal latihan yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)
2. .....

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Tuliskan soal tugas yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Deitel P., Dietel H., 2015, *Java How to Program Tenth edition*, Deitel & Associates, Inc., Prentice Hall, New Jersey, hal (119-126,181-188)
2. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, hal (131)

# MODUL 7 SELEKSI BERTINGKAT

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat mengimplementasikan konsep seleksi bertingkat untuk menyelesaikan kasus | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Textpad 2. JDK 3. Flowgorithm | |

|  |  |
| --- | --- |
|  | DASAR TEORI |

1. Tuliskan Teori-teori yang mendukung disini
2. ........

|  |  |
| --- | --- |
|  | PRAKTIK |

1. Tuliskan praktik yang harus dikerjakan mahasiswa disini
2. ........

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Tuliskan soal latihan yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)
2. .....

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Tuliskan soal tugas yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Deitel P., Dietel H., 2015, *Java How to Program Tenth edition*, Deitel & Associates, Inc., Prentice Hall, New Jersey, Hal (119-126,181-188)
2. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, Hal (113-149)
3. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, Hal (111-155)

# MODUL 8 PERULANGAN WHILE

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat mengimplementasikan konsep perulangan while untuk menyelesaikan kasus | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Textpad 2. JDK 3. Flowgorithm | |

|  |  |
| --- | --- |
|  | DASAR TEORI |

1. Tuliskan Teori-teori yang mendukung disini
2. ........

|  |  |
| --- | --- |
|  | PRAKTIK |

1. Tuliskan praktik yang harus dikerjakan mahasiswa disini
2. ........

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Tuliskan soal latihan yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)
2. .....

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Tuliskan soal tugas yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Deitel P., Dietel H., 2015, *Java How to Program Tenth edition*, Deitel & Associates, Inc., Prentice Hall, New Jersey, Hal (119-126,181-188)
2. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, Hal (113-149)
3. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, Hal (111-155)

# MODUL 9 PERULANGAN WHILE

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat mengimplementasikan konsep perulangan while untuk menyelesaikan kasus | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Textpad 2. JDK 3. Flowgorithm | |

|  |  |
| --- | --- |
|  | DASAR TEORI |

1. Tuliskan Teori-teori yang mendukung disini
2. ........

|  |  |
| --- | --- |
|  | PRAKTIK |

1. Tuliskan praktik yang harus dikerjakan mahasiswa disini
2. ........

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Tuliskan soal latihan yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)
2. .....

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Tuliskan soal tugas yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Deitel P., Dietel H., 2015, *Java How to Program Tenth edition*, Deitel & Associates, Inc., Prentice Hall, New Jersey, Hal (119-126,181-188)
2. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, Hal (113-149)
3. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, Hal (111-155)

# MODUL 10 PERULANGAN WHILE

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat mengimplementasikan konsep perulangan while untuk menyelesaikan kasus | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Textpad 2. JDK 3. Flowgorithm | |

|  |  |
| --- | --- |
|  | DASAR TEORI |

1. Tuliskan Teori-teori yang mendukung disini
2. ........

|  |  |
| --- | --- |
|  | PRAKTIK |

1. Tuliskan praktik yang harus dikerjakan mahasiswa disini
2. ........

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Tuliskan soal latihan yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)
2. .....

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Tuliskan soal tugas yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Deitel P., Dietel H., 2015, *Java How to Program Tenth edition*, Deitel & Associates, Inc., Prentice Hall, New Jersey, Hal (119-126,181-188)
2. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, Hal (113-149)
3. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, Hal (111-155)

# MODUL 11 PERULANGAN WHILE

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat mengimplementasikan konsep perulangan while untuk menyelesaikan kasus | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Textpad 2. JDK 3. Flowgorithm | |

|  |  |
| --- | --- |
|  | DASAR TEORI |

1. Tuliskan Teori-teori yang mendukung disini
2. ........

|  |  |
| --- | --- |
|  | PRAKTIK |

1. Tuliskan praktik yang harus dikerjakan mahasiswa disini
2. ........

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Tuliskan soal latihan yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)
2. .....

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Tuliskan soal tugas yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Deitel P., Dietel H., 2015, *Java How to Program Tenth edition*, Deitel & Associates, Inc., Prentice Hall, New Jersey, Hal (119-126,181-188)
2. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, Hal (113-149)
3. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, Hal (111-155)

# MODUL 12 PERULANGAN WHILE

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat mengimplementasikan konsep perulangan while untuk menyelesaikan kasus | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Textpad 2. JDK 3. Flowgorithm | |

|  |  |
| --- | --- |
|  | DASAR TEORI |

1. Tuliskan Teori-teori yang mendukung disini
2. ........

|  |  |
| --- | --- |
|  | PRAKTIK |

1. Tuliskan praktik yang harus dikerjakan mahasiswa disini
2. ........

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Tuliskan soal latihan yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)
2. .....

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Tuliskan soal tugas yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Deitel P., Dietel H., 2015, *Java How to Program Tenth edition*, Deitel & Associates, Inc., Prentice Hall, New Jersey, Hal (119-126,181-188)
2. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, Hal (113-149)
3. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, Hal (111-155)

# MODUL 13 PERULANGAN WHILE

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat mengimplementasikan konsep perulangan while untuk menyelesaikan kasus | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Textpad 2. JDK 3. Flowgorithm | |

|  |  |
| --- | --- |
|  | DASAR TEORI |

1. Tuliskan Teori-teori yang mendukung disini
2. ........

|  |  |
| --- | --- |
|  | PRAKTIK |

1. Tuliskan praktik yang harus dikerjakan mahasiswa disini
2. ........

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Tuliskan soal latihan yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)
2. .....

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Tuliskan soal tugas yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Deitel P., Dietel H., 2015, *Java How to Program Tenth edition*, Deitel & Associates, Inc., Prentice Hall, New Jersey, Hal (119-126,181-188)
2. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, Hal (113-149)
3. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, Hal (111-155)

# MODUL 14 PERULANGAN WHILE

|  |  |
| --- | --- |
|  | CAPAIAN PEMBELAJARAN |
| 1. Mahasiswa dapat mengimplementasikan konsep perulangan while untuk menyelesaikan kasus | |

|  |  |
| --- | --- |
|  | KEBUTUHAN ALAT/BAHAN/SOFTWARE |
| 1. Textpad 2. JDK 3. Flowgorithm | |

|  |  |
| --- | --- |
|  | DASAR TEORI |

1. Tuliskan Teori-teori yang mendukung disini
2. ........

|  |  |
| --- | --- |
|  | PRAKTIK |

1. Tuliskan praktik yang harus dikerjakan mahasiswa disini
2. ........

|  |  |
| --- | --- |
|  | **LATIHAN** |

1. Tuliskan soal latihan yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)
2. .....

|  |  |
| --- | --- |
| **tugas.png** | **TUGAS** |

1. Tuliskan soal tugas yang harus dikerjakan mahasiswa di sini (modul untuk mahasiswa tidak perlu diberi jawaban; modul untuk dosen perlu disertakan jawaban)

|  |  |
| --- | --- |
|  | **REFERENSI** |

1. Deitel P., Dietel H., 2015, *Java How to Program Tenth edition*, Deitel & Associates, Inc., Prentice Hall, New Jersey, Hal (119-126,181-188)
2. Rosa A. S., 2018, *Logika Algoritma dan Pemrograman Dasar*, Modula, Bandung, Hal (113-149)
3. Abdul Kadir, 2012, *Algoritma & Pemrograman Menggunakan Java*, CV. Andi Offset, Yogyakarta, Hal (111-155)