



MODUL PRAKTIKUM BAHASA PEMROGRAMAN

OLEH :AUGURY EL RAYEB, CHAERUL ANWAR

SISTEM INFORMASI
UNIVERSITAS PEMBANGUNAN JAYA
2018

Bahasa Pemrograman (Pemrograman Visual dengan Java) Modul Praktikum

Topik

Penggunaan Netbeans IDE, Hello World, variable dan Operator aritmetika.

Tujuan

Setelah menyelesaikan praktikum ini, mahasiswa diharapkan mampu:

1. Menggunakan Netbeans IDE untuk membuat program Java
2. Membuat program Hello World
3. Membuat program dengan menggunakan variable dan operator aritmetika.

Alat dan Bahan

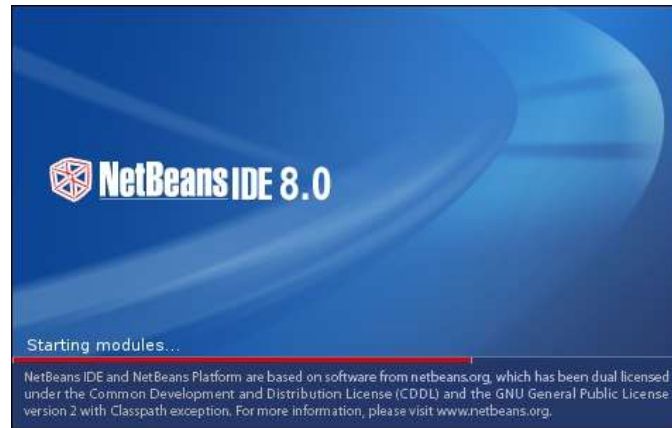
- Komputer dengan **Java Development Kit (JDK)** dan aplikasi BlueJ sudah terinstall
- Informasi download dan instalasi JDK dan Netbeans dapat diperoleh di;
<http://www.oracle.com/us/technologies/java/jdk-7-netbeans-download-432126.html>

Rangkuman/Teori

[Menjelaskan cara penggunaan aplikasi Netbeans]

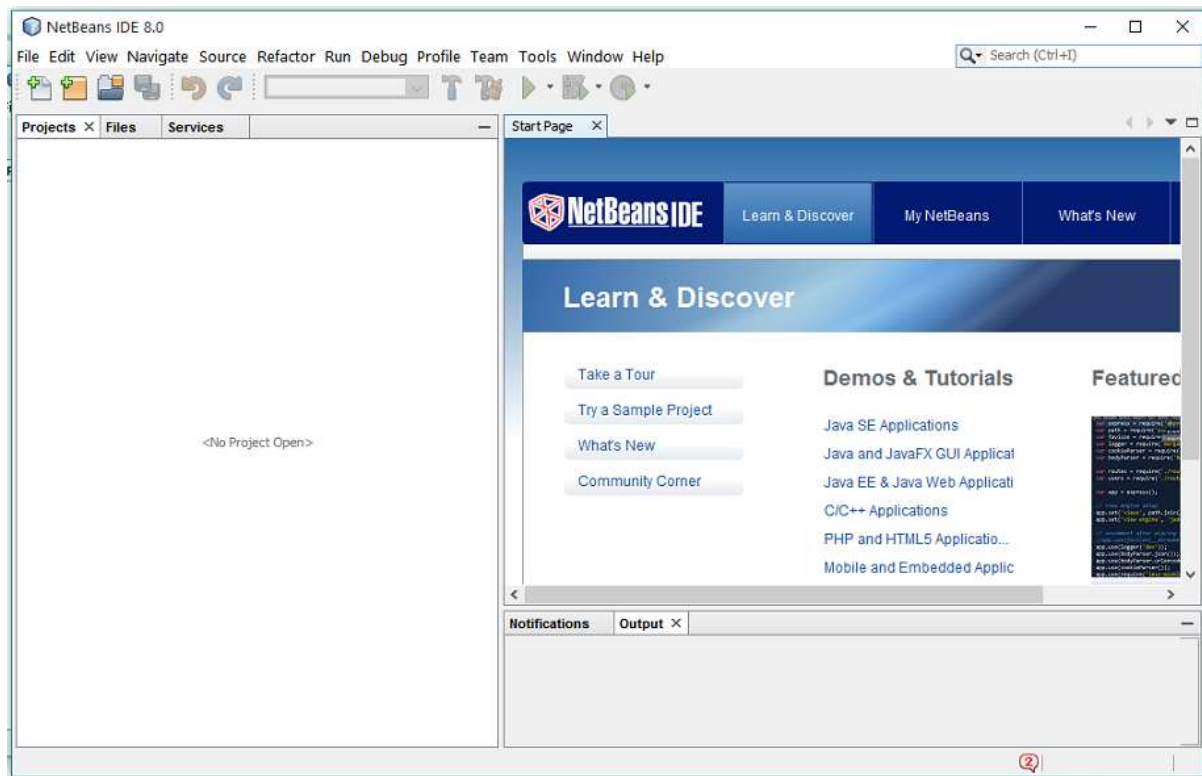
Netbeans merupakan salah satu aplikasi IDE (Integrated Development Environment) yang dapat digunakan untuk membantu pembuatan program java. Untuk menjalankan Netbeans dapat dilakukan dengan click icon Netbeans IDE pada menu windows.

Berikut adalah tampilan loading Netbeans 8.0 saat dijalankan.



Gambar 1. Tampilan Netbeans saat loading

Netbeans

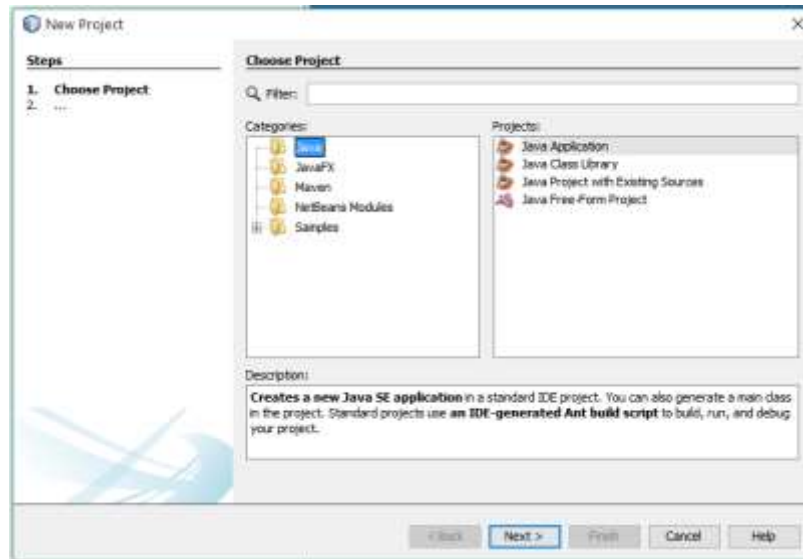


Gambar 2. Tampilan Aplikasi Netbeans 8

Membuat Project

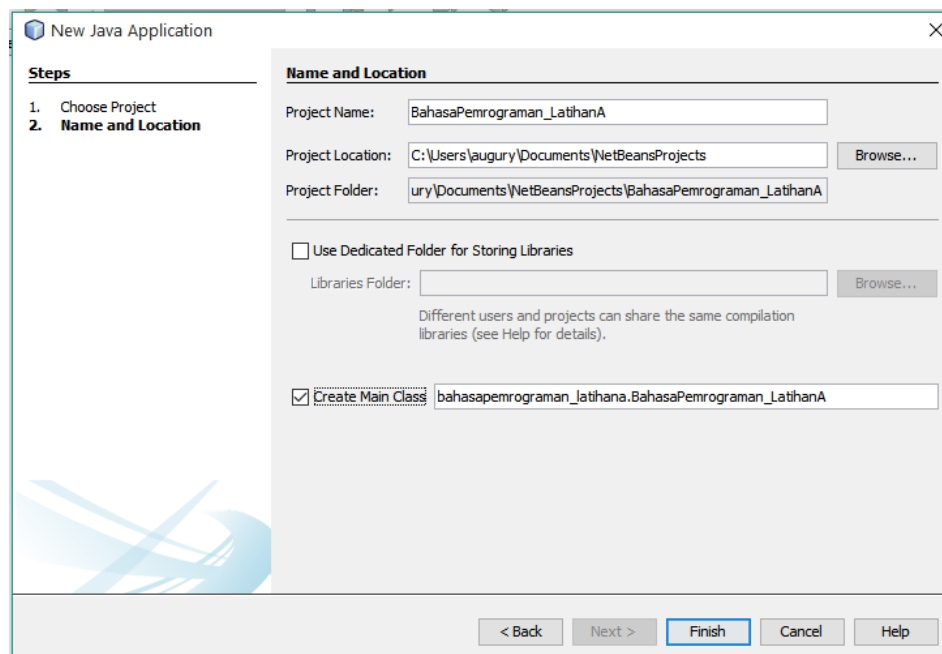
Sebelum membuat class kita harus membuat project yang akan menampung class-class yang kita buat.

Klik menu File → New Project , kemudian akan tampil dialog untuk memilih jenis project yang akan kita buat.



Gambar 3. Dialog *New Project* untuk memilih jenis project
Pada dialog choose project tersebut, untuk saat ini pilih;

- **Java** pada *Categories*,
- **Java Application** pada *Projects*, dan
- Klik tombol **Next**



Gambar 4. Dialog *New Java Application* untuk menulis nama project dan menentukan folder project.

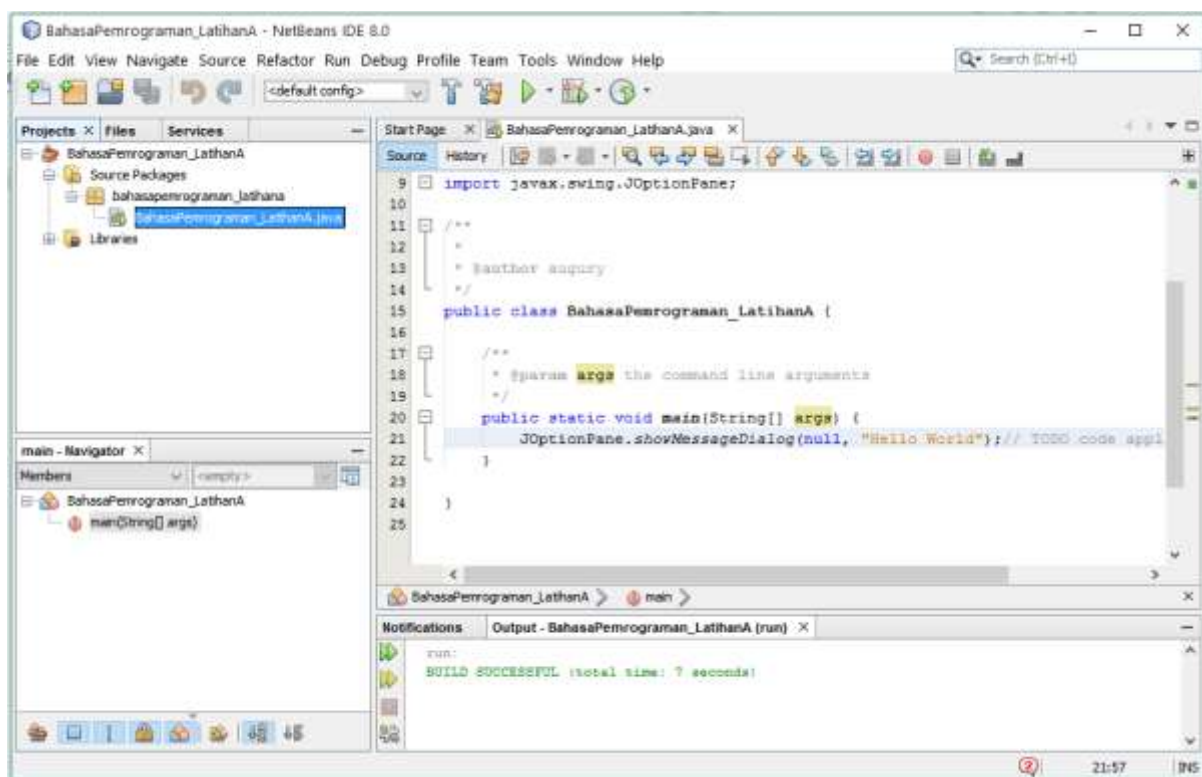
Pada dialog *New Java Application* tersebut, lakukan hal berikut;

- tulis nama project pada isian **Project Name** (nama project tidak boleh ada spasi, upayakan diawali dengan huruf besar untuk tiap satu kata, jangan menggunakan special characters kecuali _ dan – Pada contoh dialog diatas nama project: **BahasaPemrograman_LatihanA**
- Tentukan lokasi penyimpanan project pada isian **Project Location** (boleh ditulis melalui isian atau dengan cara pilih melalui tombol **Browse**)
- Pada **Check Box Create Main Class** (untuk saat ini) pastikan di contrenng (**checked**).
- Tekan tombol **Finish**.

Setelah proses-proses tersebut di atas, maka akan dibuat project **Java Application** dengan nama project **BahasaPemrograman_LatihanA**.

Untuk mencoba output program anda bisa menambahkan perintah berikut di dalam block method **public static void main(String [] args)** , yaitu di antara block { dan } method.

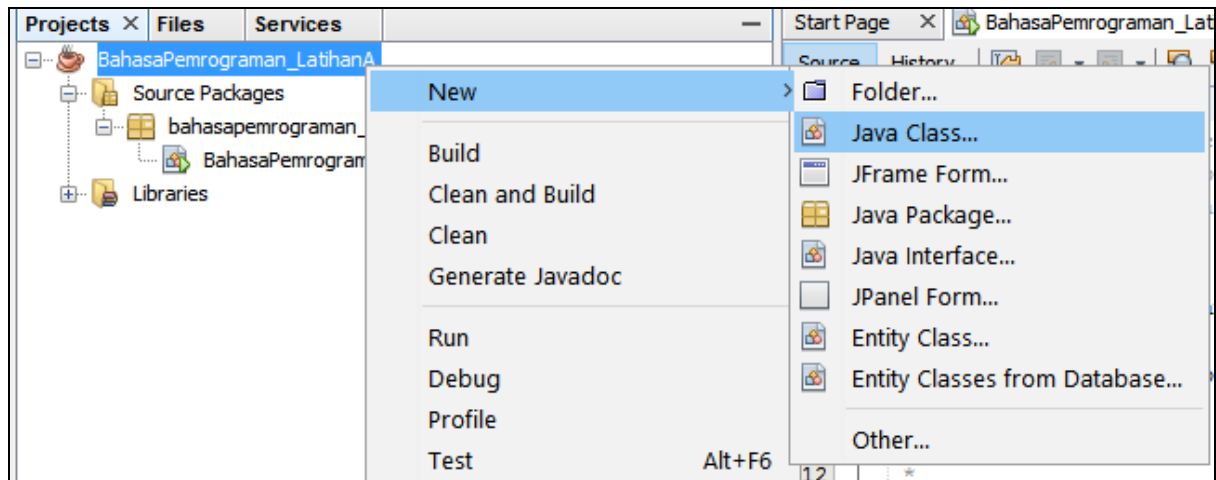
Tampilan Netbeans IDE setelah dibuat project akan tampil seperti berikut:



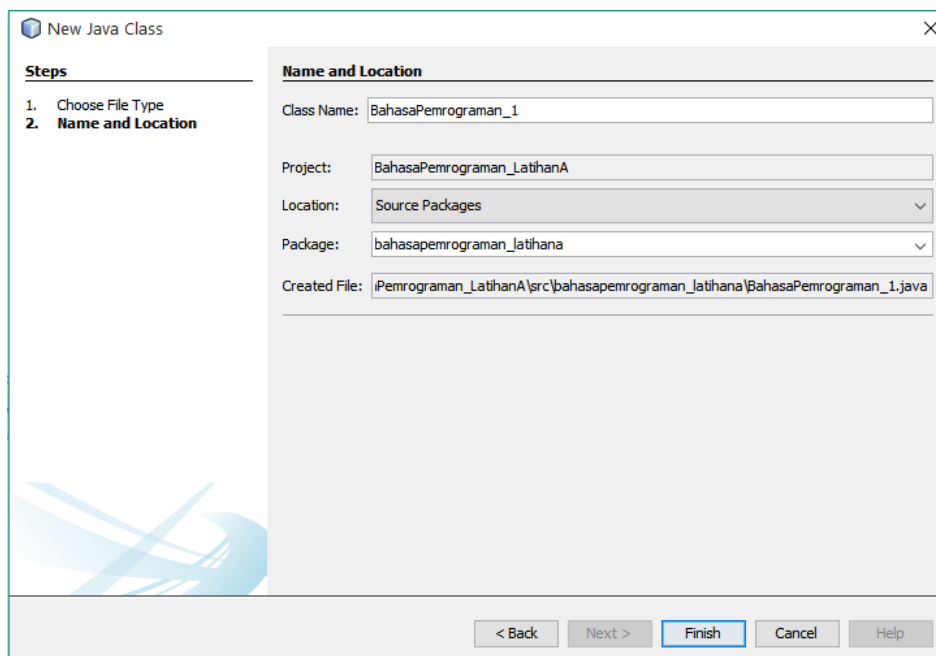
Gambar 5. Netbeans IDE dengan Project Java Application

Membuat Class

Selanjutnya untuk membuat class, **Klik kanan** pada **nama project** → **New** → **Java Class**, kemudian akan tampil **panel New Java Class**.

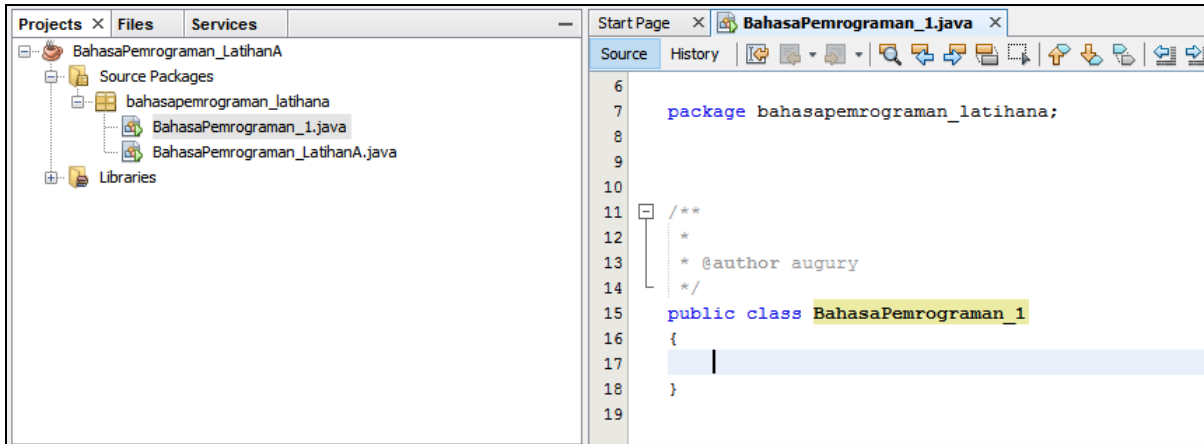


Gambar 6. Menu untuk menambah *Java Class* pada *project*



Gambar 7. Panel New Java Class.

Kemudian isikan nama class yang akan dibuat pada isian **Class Name** (disarankan nama class diawali dengan huruf besar, untuk setiap awal kata pada nama class gunakan huruf besar, *nama class tidak boleh ada spasi dan nama class akan dijadikan nama file*). Pada isian Package, pilih paket yang ada pada pilihan tersebut yaitu package dari project yang sudah kita bikin.



Gambar 8. Tampilan *code editor*

Pada editor akan ditampilkan kode dasar untuk class yang tadi dibuat. Selanjutnya tinggal ditambahkan kode program sesuai aplikasi yang ingin dibuat.

[Menjelaskan tentang *variable*]

Variable merupakan suatu *identifier* yang akan menampung suatu nilai, nilai pada *variable* tersebut dapat berubah-ubah tergantung proses yang terjadi di dalam program. Sebelum kita dapat menggunakan *variable* terlebih dahulu kita harus mendeklarasikannya dengan aturan *syntax* seperti berikut:

<Type> <nama variable>;

<Type> merupakan tipe dari data yang akan ditampung oleh *variable*, berikut adalah tipe-tipe dasar (*primitive type*) yang dapat digunakan:

Primitive Type	Data / Nilai yang ditampung	Jangkauan Nilai
byte	8-bit integer	-128 to 127
short	16-bit integer	-32,768 to 32,767
int	32-bit integer	-2,147,483,648 to 2,147,483,647
long	64-bit integer	-2^{63} to $2^{63} - 1$
float	32-bit floating-point	6 significant digits (10^{-46} , 10^{38})

double	64-bit floating-point	15 significant digits (10^{-324} , 10^{308})
char	Unicode character	
boolean	Boolean variable	false and true
Type Variable yang Umum Digunakan, Selain Type di Atas:		
String	Dapat menampung kata, atau beberapa kata	

Contoh penulisan *variable*:

Int kecepatan;

Char pilihanAnda;

[Menjelaskan tentang menampilkan *output* ke sistem konsol (*console system*)]

Untuk menampilkan output ke sistem konsol, gunakan instruksi dengan syntax berikut:

System.out.print(<variable atau tulisan yang akan ditampilkan>);

[Menjelaskan tentang membuat *Input* dan *Output* dengan *JoptionPane*]

Sebelum menggunakan JoptionPane kita harus melakukan import JoptionPane, dengan menuliskan perintah berikut di atas class :

import javax.swing.JoptionPane;

Output Visual Melalui Dialog Form

Untuk menampilkan output dengan menggunakan JoptionPane, tuliskan perintah dengan *syntax* berikut:

JoptionPane.showMessageDialog(null, <variable atau tulisan yang akan ditampilkan>);

Input Visual Melalui Dialog Form

Untuk menggunakan input (sebagai masukan dari *user*) secara visual, tuliskan perintah dengan **syntax** berikut:

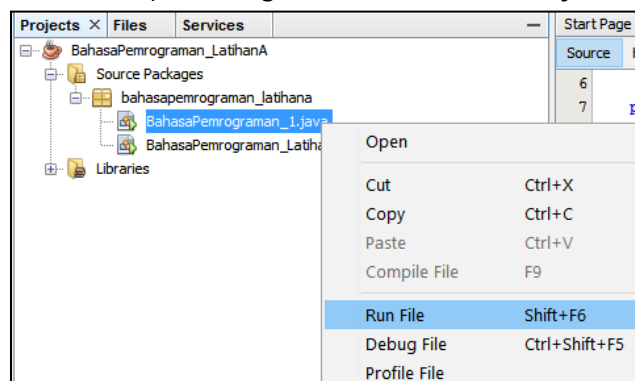
String strInputUser = JOptionPane.showInputDialog(null, <tulisan pada dialog yang memberitahu user apa yang harus user *input*>);

Praktek

Pada bagian ini, akan dibahas mengenai praktek yang akan dilakukan. Untuk mempermudah pencatatan hasil praktek dan penarikan kesimpulan terhadap hasil praktek tersebut, Anda dapat menggunakan form pada **Appendix 1**.

Praktek 1: Menampilkan tulisan Hello World

- Langkah 1: Buat class dengan nama **Lab_2_HelloWorld**
- Langkah 2: Lakukan modifikasi kode pada class dengan menambahkan method main, dan tambahkan instruksi yang menampilkan tulisan: Hello World (menggunakan visual swing JOptionPane) di dalam method main.
- Langkah 3: Jalankan class yang sudah dimodifikasi tersebut, dengan cara **klik kanan pada nama file** → **Run File** (atau dengan menekan tombol **Shift + F6** pada keyboard).



- Langkah 4: Jika ada kesalahan, lakukan perbaikan yang diperlukan.

Praktek 2: Membuat Variable

- Langkah 1: Buat class dengan nama **Lab_2_Var**
- Langkah 2: Lakukan modifikasi kode pada praktek 1 dengan menambahkan variable yang kemudian variable tersebut diisi dengan nama anda.
- Langkah 3: Lakukan modifikasi agar saat class dijalankan akan menampilkan tulisan Hello disertai dengan isi variable nama.

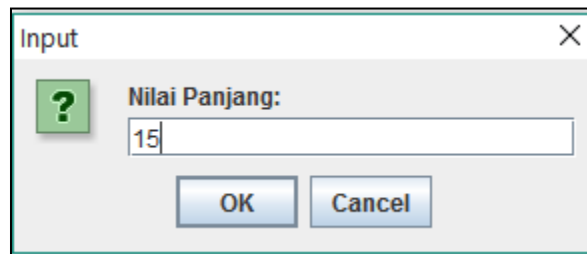
- Langkah 4: Jalankan class yang sudah dimodifikasi tersebut, dengan cara **klik kanan pada nama file → Run File** (atau dengan menekan tombol **Shift + F6** pada keyboard).
- Langkah 5: Jika ada kesalahan, lakukan perbaikan yang diperlukan.

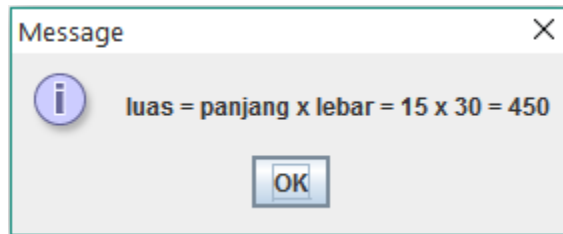
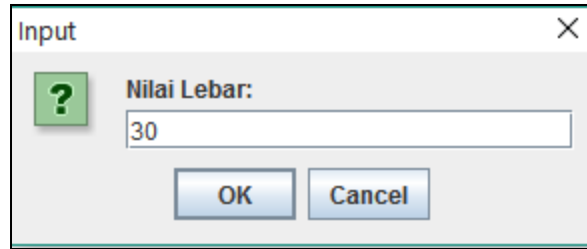
Praktek 3: Membuat Class untuk menghitung luas persegi panjang

- Langkah 1: Buat class baru dengan nama **HitungLuasPersegiPanjang**.
- Langkah 2: Modifikasi kode dengan menambahkan:
- variable dengan nama **panjang**, type byte dan isi dengan nilai 15.
 - variable dengan nama **lebar**, type byte dan isi dengan nilai 30.
 - variable dengan nama **luas**, type short, isinya merupakan hasil perkalian dari variable panjang dengan variable lebar.
- Langkah 3: Modifikasi kode agar saat class dijalankan akan menampilkan seperti berikut;
(Contoh:) $\text{Luas} = \text{panjang} \times \text{lebar} = 15 \times 30 = 450$
Dimana 15 merupakan isi variable panjang, 30 merupakan isi variable lebar.
(Untuk menampilkan gunakan visual swing JOptionPane).
- Langkah 4: Jalankan class yang sudah dimodifikasi tersebut, dengan cara **klik kanan pada nama file → Run File** (atau dengan menekan tombol **Shift + F6** pada keyboard).
- Langkah 5: Jika ada kesalahan, lakukan perbaikan yang diperlukan.

Praktek 4: Luas persegi panjang dengan Input dan Output Visual

- Langkah 1: Buat class baru dengan nama **VisualLuasPersegiPanjang**.
- Langkah 2: Modifikasi kode dengan menambahkan:
- variable dengan nama **panjang**, type byte dan isi dengan nilai 15.
 - variable dengan nama **lebar**, type byte dan isi dengan nilai 30.
 - variable dengan nama **luas**, type short, isinya merupakan hasil perkalian dari variable panjang dengan variable lebar.
- Langkah 3: Modifikasi kode agar saat class dijalankan akan menampilkan input (untuk membaca panjang dan lebar) dan output (menampilkan luas) secara visual (dengan *Dialog Form*) dengan tampilan seperti berikut;





- Langkah 4: Jalankan class yang sudah dimodifikasi tersebut, dengan cara **klik kanan pada nama file → Run File** (atau dengan menekan tombol **Shift + F6** pada keyboard).
- Langkah 5: Jika ada kesalahan, lakukan perbaikan yang diperlukan.

Analisa Hasil Praktek

Berdasarkan praktek yang telah Anda lakukan, buat kesimpulan:

Tugas

1. Buat program yang akan melakukan perhitungan luas segitiga, dan menampilkan hasilnya.
2. Buat program yang akan melakukan perhitungan luas lingkaran, dan menampilkan hasilnya.

Appendix 1: Form Data Praktek

Nama :

NIM :

Tgl :

Penggalan <i>Source code</i>	Keluaran / <i>Output</i>

Bahasa Pemrograman (Pemrograman Visual dengan Java) Modul Praktikum

Topik

Structured Control (Kendali terstruktur).

Tujuan

Setelah menyelesaikan praktikum ini, mahasiswa diharapkan mampu:

4. Membuat program yang memerlukan penggunaan *selection control* seperti; *if statement* dan *switch statement*.
5. Membuat program yang memerlukan penggunaan *iteration control* seperti; *for statement*, *while statement*.
6. Membuat program yang memerlukan penggunaan *break statement* dan *continue statement*.
7. Membuat program yang dapat mengolah suatu kalimat atau *string*.

Alat dan Bahan

- Komputer dengan **Java Development Kit (JDK)** dan aplikasi BlueJ sudah terinstall
- Informasi download dan instalasi JDK dan Netbeans dapat diperoleh di;
<http://www.oracle.com/us/technologies/java/jdk-7-netbeans-download-432126.html>

Rangkuman/Teori

[Menjelaskan tentang *conditional statement*]

Conditional statements biasa digunakan untuk menguji suatu kondisi, kondisi dari statement tersebut biasanya untuk menentukan arah dari flow suatu algoritma.

Conditional statements sering digunakan dalam:

- Percabangan (selection/decision)
- Perulangan (repetition/looping)

Terdapat dua operator yang biasanya digunakan dalam *conditional statements*:

- *Relational operator*
- *Logical operator*

Berikut adalah tabel yang berisi daftar *relational operator* dan tabel yang berisi daftar *logical operator* yang digunakan dalam pemrograman java.

Tabel 1. Relational Operator

Nama Operator	Operasi
==	Equality
!=	Inequality
<	Less than
>	Greater than
<=	Less than or equal
>=	Greater than or equal

Tabel 2. Logical Operator

x	y	AND	OR	NOT
		x && y	x y	!x
false	false	false	false	true
false	true	false	true	true
True	false	false	true	false
true	True	true	true	false

[Menjelaskan tentang *if statement*]

Salah satu instruksi yang dapat digunakan untuk mengendalikan alur program adalah *if statement*. Dengan *if statement* kita bisa mengarahkan alur program sesuai kondisi (dari hasil *conditional statement*) yang ditentukan pada *if statement*.

Syntax if statement:

```

if( <conditional statement> ) {
    <kumpulan instruksi jika kondisi terpenuhi>
}

```

```

        else {
            <kumpulan instruksi jika kondisi tidak terpenuhi>
        }
atau;

        if( <conditional statement> ) {
            <kumpulan instruksi jika kondisi terpenuhi>
        }

```

[Menjelaskan tentang *switch statement*]

Salah satu instruksi yang dapat digunakan untuk mengendalikan alur program adalah *switch statement*. Dengan *switch statement* kita bisa mengarahkan alur sesuai dengan kecocokan antara nilai dari variable uji dengan nilai uji.

Syntax switch statement:

```

switch( <variable uji> ) {
    case <nilai uji_1>: <kumpulan instruksi jika nilai variable cocok dengan nilai uji 1 terpenuhi>;
                        break;
    case <nilai uji_2>: <kumpulan instruksi jika nilai variable cocok dengan nilai uji 2 terpenuhi>;
                        break;
    case <nilai uji_3>: <kumpulan instruksi jika nilai variable cocok dengan nilai uji 1 terpenuhi>;
                        break;
    dst....
    default: <kumpulan instruksi jika nilai variable tidak ada yang cocok dengan nilai uji>;
}

```

[Menjelaskan tentang *for statement*]

Salah satu instruksi yang dapat digunakan untuk mengendalikan alur program yang berupa perulangan adalah *for statement*. Dengan *for statement* kita bisa mengarahkan alur agar berulang sesuai dengan jumlah yang ditentukan.

Syntax for statement:

```

for(<inisialisasi>; <batas akhir>; <multiplikasi> ) {
    <kumpulan instruksi yang akan diulang>
}

```

Contoh:

```

for(int i=1; i<=5; i++) {
    System.out.print("Perulangan ke – ");
}

```

```
        System.out.println(i);  
    }
```

[Menjelaskan tentang *while statement*]

Salah satu instruksi yang dapat digunakan untuk mengendalikan alur program yang berupa perulangan adalah *while statement*. Dengan *while statement* kita bisa mengarahkan alur agar berulang hingga *conditional statement* pada *while statement* tersebut terpenuhi.

Syntax while statement:

```
while(<conditional statement> ) {  
    <kumpulan instruksi yang akan diulang>  
}
```

atau;

```
do {  
    <kumpulan instruksi yang akan diulang>  
} while(<conditional statement> );
```

Contoh:

```
x=0;  
while(x<6) {  
    System.out.print("Perulangan ke – ");  
    System.out.println(x);  
    x++;  
}
```

[Menjelaskan tentang *break dan continue statement*]

break merupakan instruksi yang bisa digunakan untuk keluar dari suatu perulangan walaupun kondisi atau jumlah perulangan belum selesai.

continue merupakan instruksi yang bisa digunakan untuk mem-bypass beberapa instruksi didalam perulangan.

[Menjelaskan tentang tipe data *String*]

String dalam pemrograman Java adalah suatu class, dan penanganannya tidak sama dengan type data primitive (spt; int, char, double, boolean,..dsb).

Variable dengan tipe data *String* pada pemrograman Java untuk selanjutnya akan disebut sebagai object. Berikut adalah deklarasi untuk membuat object *String*;

***String* <nama_object>;**

Contoh:

***String* x;** ← perintah ini akan membuat object dengan nama x, dengan tipe *String*.

***String* pesan = "Hallo";** ← perintah ini akan membuat object dengan nama pesan, dengan tipe *String*, dan mengisikan tulisan Hallo pada object pesan.

Method pada *String*

Method yang dapat digunakan oleh object tipe *String* diantaranya adalah:

- **length()**

Untuk mengetahui panjang karakter dari suatu string.

Contoh:

```
String kalimat = "ABCDEFGH";  
int panjang = kalimat.length();
```

- **charAt(int n)**

Untuk mengetahui karakter yang tersimpan pada posisi ke-n dalam suatu *String*.

Contoh:

```
String kalimat = "ABCDEFGH";  
char huruf = kalimat.charAt(4);
```

- **substring(int b)**

Untuk mengetahui potongan string yang tersimpan pada suatu *String* mulai dari posisi ke-b sampai posisi akhir *String*.

- **substring(int b, int e)**

Untuk mengetahui potongan string yang tersimpan pada suatu *String* mulai dari posisi ke-b, sampai posisi ke-e.

Contoh:

```
String kalimat = "ABCDEFGH";  
String potongan1 = kalimat.substring(4);  
String potongan2 = kalimat.substring(4, 6);
```

- **lastIndexOf(<karakterYangDicari>)**

Untuk mengetahui posisi index terakhir dari suatu karakter yang dicari pada suatu *String*.

Contoh:

```
String kalimat = "AB#CDE#FGHI";  
int idx = kalimat.lastIndexOf('#');
```

- **indexOf(<karakterYangDicari>, <indexAwalPencarian>)**

Untuk mengetahui posisi index dari suatu karakter yang dicari pada suatu String, pencarian dimulai pada posisi index seperti yang dituliskan pada <indexAwalPencarian>.

Contoh:

```
String kalimat = "AB#CDE#FGHI";  
int idx = kalimat.indexOf('#', 3);
```

- **toCharArray()**

Untuk mengubah object string menjadi array karakter. Biasa digunakan untuk menyalin seluruh karakter pada object String ke object array.

Contoh:

```
String kalimat = "AB#CDE#FGHI";  
Char [] arrayKarakter = kalimat.toCharArray();
```

Konversi pada String

Terdapat beberapa cara konversi angka menjadi string:

- **typeAsal.toString(angka)**

Konversi angka yang memiliki type seperti yang disebutkan pada typeAsal menjadi string.

Contoh:

```
String tipeDouble = Double.toString(14.2); //hasilnya adalah "14.2"  
String tipeFloat = Float.toString(1420000000); //hasilnya adalah "1.42E9"  
int nilai = 3;  
System.out.println(Integer.toString(nilai*4); //hasilnya adalah "12"
```

- **Integer.toString(angka, format)**

konversi dari integer menjadi string dengan format bilangan yang ditentukan

format: **2** (untuk bilangan biner), **8** (untuk bilangan oktal), **10** (untuk bilangan desimal), **16** (untuk bilangan heksadesimal).

Contoh:

```
int nilai = 3;  
System.out.println(Integer.toString(nilai*4, 2); //hasilnya adalah "1100"  
System.out.println(Integer.toString(nilai*4, 16); //hasilnya adalah "C"
```

Cara konversi dari String menjadi angka:

- **Integer.parseInt(string);**

Melakukan konversi dari string (string suatu angka) menjadi angka tipe integer.

Double.parseDouble(string);

Melakukan konversi dari string (string suatu angka) menjadi angka tipe double.

Contoh:

String diameter = "22";

String pi = "3.14";

Double kelilingLingkaran = **Integer.parseInt(diameter)** * **Double.parseDouble(pi)**;

System.out.println(kelilingLingkaran);

Praktek

Pada bagian ini, akan dibahas mengenai praktek yang akan dilakukan. Untuk mempermudah pencatatan hasil praktek dan penarikan kesimpulan terhadap hasil praktek tersebut, Anda dapat menggunakan form pada **Appendix 1**.

Praktek 1: Menggunakan *if statement*

Langkah 5: Buat class dengan nama **Lab_2_if**.

Langkah 6: Tambahkan kode program agar program:

- Membaca *input user*
- Mencocokkan apakah *input user* adalah kurang dari 10.
- Jika *input user* kurang dari 10 maka akan tampil dialog yang berisi pesan: "Input anda kurang dari 10"
- Jika *input user* lebih besar atau sama dengan 10 maka akan tampil dialog yang berisi pesan: "Input anda sangat banyak"

Langkah 7: Jalankan class yang sudah dimodifikasi tersebut, dengan cara **klik kanan pada nama file → Run File** (atau dengan menekan tombol **Shift + F6** pada keyboard).

Langkah 8: Jika ada kesalahan, lakukan perbaikan yang diperlukan.

Praktek 2: Menggunakan *switch statement*

Langkah 1: Buat class dengan nama **Lab_2_switch**.

Langkah 2: Tambahkan kode program agar program:

- Membaca *input user* dengan menggunakan dialog input (JOptionPane), dengan tulisan **masukkan jenis membership anda:**
- Mencocokkan apakah *input user*.
- Jika *input user* adalah "GOLD" , maka akan tampil dialog yang berisi pesan; "Terima kasih atas *input* anda sebagai member" , kemudian menampilkan dialog lagi yang berisi pesan; "Karena membership anda adalah GOLD, anda mendapatkan poin *voucher* sebesar 50 ribu rupiah" .

- iv. Jika *input user* adalah "SILVER" , maka akan tampil dialog yang berisi pesan; "Terima kasih atas *input* anda sebagai member" , kemudian menampilkan dialog lagi yang berisi pesan; "Karena membership anda adalah SILVER, anda mendapatkan poin *voucher* sebesar 30 ribu rupiah" .
- v. Jika *input user* adalah "BRONZE" , maka akan tampil dialog yang berisi pesan; "Terima kasih atas *input* anda sebagai member" , kemudian menampilkan dialog lagi yang berisi pesan; "Karena membership anda adalah BRONZE, anda mendapatkan poin *voucher* sebesar 10 ribu rupiah" .
- vi. Jika *input user* adalah bukan dari jenis membership yang ada (GOLD, SILVER, , BRONZE) maka akan tampil dialog yang berisi pesan; "Terima kasih atas *input* anda sebagai konsumen" , kemudian menampilkan dialog lagi yang berisi pesan; "Meskipun anda bukan member, anda berhak mendapat potongan sebesar 5000 rupiah dari total belanja" .

Langkah 3: Jalankan class yang sudah dimodifikasi tersebut, dengan cara **klik kanan pada nama file → Run File** (atau dengan menekan tombol **Shift + F6** pada keyboard).

Langkah 4: Jika ada kesalahan, lakukan perbaikan yang diperlukan.

Praktek 3: Mengolah data dengan String

Langkah 6: Buat class baru dengan nama **Lab_2_PeriksaKarakterString** dan lengkapi hingga seperti berikut:

```
import javax.swing.JOptionPane;
public class Lab_2_PeriksaKarakterString {
    public static void main(String[] args) {
        String kalimat;
        char cari;
        int jumlah=0;

        kalimat = JOptionPane.showInputDialog(null,"Masukkan kalimat anda! ");
        cari = JOptionPane.showInputDialog(null,"Masukkan karakter yg ingin dihitung jumlahnya! ").charAt(0);
        for (int i = 0; i < kalimat.length(); i++) {
            if (cari == kalimat.charAt(i)) {
                jumlah++;
            }
        }
        JOptionPane.showMessageDialog(null, "jumlah huruf " + cari + " = " + jumlah);
    }
}
```

Langkah 7: Pelajari kode program tersebut dan buat analisis proses apa saja yang dilakukan oleh program tersebut!

Praktek 4: While statement

Langkah 6: Buat class baru dengan nama **Lab_2_PeriksaKarakterString_2**.

Langkah 7: Salin isi dari main program pada praktek 3

Langkah 8: Modifikasi program sehingga perulangannya menggunakan while – *statement*

Langkah 9: Jalankan class yang sudah dimodifikasi tersebut, dengan cara **klik kanan pada nama file → Run File** (atau dengan menekan tombol **Shift + F6** pada keyboard).

Langkah 10: Jika ada kesalahan, lakukan perbaikan yang diperlukan.

Analisa Hasil Praktek

Berdasarkan praktek yang telah Anda lakukan, buat kesimpulan:

Tugas

4. Lanjutkan praktek jika pelaksanaan di Lab belum selesai.

Appendix 1: Form Data Praktek

Nama :

NIM :

Tgl :

Penggalan <i>Source code</i>	Keluaran / <i>Output</i>

Bahasa Pemrograman (Pemrograman Visual dengan Java) Modul Praktikum

Topik

String, Array & FileReader

Tujuan

Setelah menyelesaikan praktikum ini, mahasiswa diharapkan mampu:

8. Membuat program dengan menggunakan object String untuk proses manipulasi terhadap object String.
9. Membuat program dengan menggunakan object array untuk proses manipulasi terhadap object array.
10. Membuat program yang dapat membaca file teks.

Alat dan Bahan

- Komputer dengan **Java Development Kit (JDK)** dan aplikasi BlueJ sudah terinstall
- Informasi download dan instalasi JDK dapat diperoleh di;
<http://www.oracle.com/technetwork/java/javase/downloads/index.html>
- Informasi download dan instalasi eclipse dapat diperoleh di;
- <http://www.csee.umbc.edu/courses/undergraduate/341/fall08/Lectures/Eclipse/intro-to-eclipse.pdf>

Rangkuman/Teori

[Menjelaskan tentang tipe data *String*]

String dalam pemrograman Java adalah suatu class, dan penanganannya tidak sama dengan type data primitive (spt; int, char, double, boolean,...dsb).

Variable dengan tipe data String pada pemrograman Java untuk selanjutnya akan disebut sebagai object. Berikut adalah deklarasi untuk membuat object String;

String <nama_object>;

Contoh:

String x; ← perintah ini akan membuat object dengan nama x, dengan tipe String.

String pesan = "Hallo"; ← perintah ini akan membuat object dengan nama pesan, dengan tipe String, dan mengisi tulisan Hallo pada object pesan.

Method pada String

Method yang dapat digunakan oleh object tipe String diantaranya adalah:

- **length()**

Untuk mengetahui panjang karakter dari suatu string.

Contoh:

```
String kalimat = "ABCDEFGH";  
int panjang = kalimat.length();
```

- **charAt(int n)**

Untuk mengetahui karakter yang tersimpan pada posisi ke-n dalam suatu String.

Contoh:

```
String kalimat = "ABCDEFGH";  
char huruf = kalimat.charAt(4);
```

- **substring(int b)**

Untuk mengetahui potongan string yang tersimpan pada suatu String mulai dari posisi ke-b sampai posisi akhir String.

- **substring(int b, int e)**

Untuk mengetahui potongan string yang tersimpan pada suatu String mulai dari posisi ke-b, sampai posisi ke-e.

Contoh:

```
String kalimat = "ABCDEFGH";  
String potongan1 = kalimat.substring(4);  
String potongan2 = kalimat.substring(4, 6);
```

- **lastIndexOf(<karakterYangDicari>)**

Untuk mengetahui posisi index terakhir dari suatu karakter yang dicari pada suatu String.

Contoh:

```
String kalimat = "AB#CDE#FGH";  
int idx = kalimat.lastIndexOf('#');
```

- **indexOf(<karakterYangDicari>, <indexAwalPencarian>)**

Untuk mengetahui posisi index dari suatu karakter yang dicari pada suatu String, pencarian dimulai pada posisi index seperti yang dituliskan pada <indexAwalPencarian>.

Contoh:

```
String kalimat = "AB#CDE#FGHI";  
int idx = kalimat.indexOf('#', 3);
```

- **toCharArray()**

Untuk mengubah object string menjadi array karakter. Biasa digunakan untuk menyalin seluruh karakter pada object String ke object array.

Contoh:

```
String kalimat = "AB#CDE#FGHI";  
Char [] arrayKarakter = kalimat.toCharArray();
```

Konversi pada String

Terdapat beberapa cara konversi angka menjadi string:

- **typeAsal.toString(angka)**

Konversi angka yang memiliki type seperti yang disebutkan pada typeAsal menjadi string.

Contoh:

```
String tipeDouble = Double.toString(14.2); //hasilnya adalah "14.2"  
String tipeFloat = Float.toString(1420000000); //hasilnya adalah "1.42E9"  
int nilai = 3;  
System.out.println(Integer.toString(nilai*4); //hasilnya adalah "12"
```

- **Integer.toString(angka, format)**

konversi dari integer menjadi string dengan format bilangan yang ditentukan

format: **2** (untuk bilangan biner), **8** (untuk bilangan oktal), **10** (untuk bilangan desimal), **16** (untuk bilangan heksadesimal).

Contoh:

```
int nilai = 3;  
System.out.println(Integer.toString(nilai*4, 2); //hasilnya adalah "1100"  
System.out.println(Integer.toString(nilai*4, 16); //hasilnya adalah "C"
```

Cara konversi dari String menjadi angka:

- **Integer.parseInt(string);**

Melakukan konversi dari string (string suatu angka) menjadi angka tipe integer.

Double.parseDouble(string);

Melakukan konversi dari string (string suatu angka) menjadi angka tipe double.

Contoh:

```
String diameter = "22";
```

```
String pi = "3.14";
Double kelilingLingkaran = Integer.parseInt(diameter) * Double.parseDouble(pi);

System.out.println(kelilingLingkaran);
```

[Array]

Array dalam pemrograman Java adalah suatu type data reference sehingga jika kita membuat variable dengan type array maka untuk selanjutnya akan disebut sebagai object.

Berikut adalah beberapa cara deklarasi untuk membuat object Array;

```
int [] array1;
array1 = new int [100];
```

Berdasarkan instruksi-instruksi di atas kita membuat object **array1** dengan kapasitas tampung sebanyak **100 angka tipe integer**.

atau

```
char [] arrayC = new char [100];
```

Berdasarkan instruksi di atas kita membuat object **arrayC** dengan kapasitas tampung sebanyak **100 karakter**.

atau

```
int [] array2 = { 3, 4, 10, 6 };
```

Berdasarkan instruksi di atas kita membuat object **array2** dengan kapasitas tampung sebanyak **4 angka tipe integer**, dan array1 sudah diisi dengan angka 3, 4, 10, 6.

Mengisikan nilai ke object tipe array

Untuk mengisikan suatu nilai pada object tipe array digunakan instruksi sebagai berikut:

```
objArray[<noIndeks>] = <nilai>
```

keterangan:

objArray adalah nama object dengan type array yang akan diisi dengan suatu nilai

<noIndeks> adalah no indeks posisi nilai akan diisi pada object array (indeks pada array dimulai dari 0)

Contoh:

```
char [] arrayC = new char[3];
arrayC[0] = 'K';
arrayC[1] = 'a';
arrayC[2] = 'k';
```

Perulangan for Khusus untuk Object Array

Object array memiliki perulangan for khusus, berikut adalah penulisan instruksi perulangan tersebut:

for(type <namaVariable>: <namaObjectArray>)

keterangan:

type adalah type untuk variable yang menampung isi array, jadi type harus sesuai dengan type dari object array (jika object array tipe char, maka type adalah char).

<namaVariable> adalah nama variable yang nantinya akan menampung isi array.

<namaObjectArray> adalah nama object array yang akan dibaca isinya indeks demi indeks.

Perulangan akan dilakukan sebanyak kapasitas dari object array.

Contoh:

```
char [] arrayC = new char[3];
arrayC[0] = 'K';
arrayC[1] = 'a';
arrayC[2] = 'k';
for(char huruf:arrayC) {
    System.out.print(huruf);
}
```

Method length()

Method length adalah method yang dapat digunakan oleh object array untuk mengetahui kapasitas atau panjang dari suatu object array.

<namaObjectArray>.length;

keterangan:

<namaObjectArray> adalah nama dari object array yang akan dilihat kapasitasnya atau panjangnya.

Instruksi di atas akan menghasilkan suatu angka integer sebesar kapasitas atau panjang dari object array.

Contoh:

1.
 int kapasitasArray = arrayC.length;
2.
 for(int i=0;i<arrayC.length;i++){
 System.out.print(arrayC[i]);
 }

← perintah ini akan membuat object dengan nama x, dengan tipe String.

String pesan = "Halo"; ← perintah ini akan membuat object dengan nama pesan, dengan tipe String, dan mengisi tulisan Halo pada object pesan.

[Class Scanner]

Untuk membaca input melalui sistem konsol pada pemrograman Java dapat digunakan class Scanner. Untuk menggunakan class Scanner kita harus melakukan import class pada awal pemrograman seperti berikut:

```
Import java.util.Scanner;
```

Selanjutnyadidalam tubuh program buat object dengan type Scanner seperti berikut:

```
Scanner inputKonsol = new Scanner(System.in);
```

Method

Selanjutnya untuk membaca masukan (*input*) dapat digunakan method-method pada object tersebut (method yang digunakan disesuaikan dengan tipe masukan yang akan dibaca).

next()

Untuk membaca input string

hasNext()

Untuk verifikasi apakah ada input string berikutnya (outputnya: boolean)

nextInt()

Untuk membaca integer

hasNextInt()

Untuk verifikasi apakah ada input integer berikutnya (outputnya: boolean)

nextDouble()

Untuk membaca double

hasNextDouble()

Untuk verifikasi apakah ada input double berikutnya (outputnya: boolean)

nextLine()

Untuk membaca satu baris String.

hasNextLine()

Untuk verifikasi apakah ada input baris berikutnya (outputnya: boolean)

Contoh:

```
import java.util.Scanner;
public class Scanner_BacaInput {
    public static void main(String[] args) {
        Scanner input_1 = new Scanner(System.in);
        System.out.print("Nama : ");
        String nama = input_1.nextLine();
        System.out.println("Usia : ");
        int usia = input_1.nextInt();
        System.out.println("Terima kasih " + nama + "(" + usia + ")");
    }
}
```

[FileReader]

Untuk membuka file dan membaca isinya digunakan class Scanner dan class FileReader. Class FileReader digunakan untuk membuka dan membaca file, sedangkan class Scanner digunakan untuk membaca/mengambil isi file.

Untuk menggunakan class Scanner dan class FileReader kita harus melakukan import class pada awal pemrograman seperti berikut:

```
import java.util.Scanner;
import java.io.FileReader;
import java.io.FileNotFoundException;
```

class FileNotFoundException digunakan untuk melakukan pemeriksaan, apakah file yang akan dibuka ada atau tidak ada.

Selanjutnya di dalam tubuh program buat object dengan type Scanner seperti berikut:

```
Scanner inputFile = new Scanner(new FileReader(<namaFile>));
```

Parameter yang digunakan pada Scanner adalah **new FileReader(<namaFile>)** (menggantikan System.in), hal ini untuk tujuan memberitahu Scanner bahwa yang dibaca adalah dari file (melalui FileReader). Instruksi FileReader memiliki exception/pengecualian kalau file yang ingin dibuka tidak ditemukan, untuk itu maka instruksi tersebut harus diletakkan di dalam block **try ... catch** dengan menggunakan **FileNotFoundException** sebagai pendeteksi kejadian (even handler).

Contoh:

```

import java.util.Scanner;
import java.io.FileReader;
import java.io.FileNotFoundException;

public class membacaDariFile {
    public static void main(String[] args) {
        String barisIsi;
        try {
            Scanner inputFile = new Scanner(new FileReader("daftarNama.txt"));
            while(inputFile.hasNextLine()){
                barisIsi = inputFile.nextLine();
                System.out.println(barisIsi);
            }
        }
        catch(FileNotFoundException e) {
            System.out.println(e);
        }
    }
}

```

Praktek

Pada bagian ini, akan dibahas mengenai praktek yang akan dilakukan. Untuk mempermudah pencatatan hasil praktek dan penarikan kesimpulan terhadap hasil praktek tersebut, Anda dapat menggunakan form pada **Appendix 1**.

Praktek 1: Array

Langkah 9: Buat class dengan nama: **lab_3_Array**

Langkah 10: Tambahkan kode program untuk membuat object array dengan nama kalimat, dan isi kalimat (object array) dengan karakter seperti berikut:

```
'k', 'a', 'k', 'i', 'k', 'a', 'k', 'a', 'k', 'k', 'e', 'n', 'a', 'k', 'a', 'y', 'u', 't', 'a', 'k', 'b', 'e', 'r', 'p', 'a', 'k',
'u', 'k', 'a', 'k', 'u', 'k', 'a', 'k', 'u'
```

Langkah 11: Tambahkan kode program sehingga program anda dapat menampilkan seluruh isi object array.

Langkah 12: Tambahkan kode program sehingga program anda dapat menghitung jumlah huruf k yang terdapat pada object array.

Langkah 13: Tambahkan kode program sehingga tampilan output hasil program anda pada konsol adalah seperti contoh berikut:

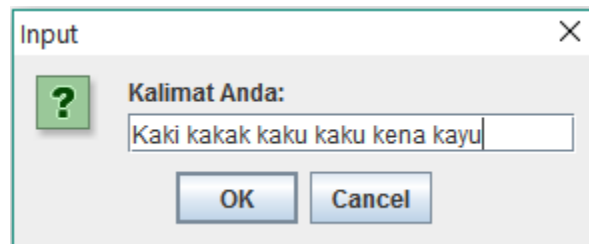
Isi object array: kakikakakkenakayutakberpakukakukaku

Jumlah huruf dalam array= 35
Jumlah huruf k dalam array= 13

Praktek 2: String

Langkah 6: Buat class dengan nama: **lab_3_String**.

Langkah 7: Tambahkan kode program untuk membaca kalimat masukan user.



Langkah 8: Tambahkan kode program yang menampilkan seluruh kalimat masukan dari user.

Langkah 9: Tambahkan kode program yang menampilkan seluruh masukan dari user, tetapi ditampilkan kata demi kata. Dan buat tampilan output hasil program anda pada konsol adalah seperti contoh berikut:

```
Kaki kakak kaku kaku kena kayu
Kaki
kakak
kaku
kaku
kena
kayu
```

Langkah 10: Jika ada kesalahan, lakukan perbaikan yang diperlukan.

Praktek 3: Membaca File teks

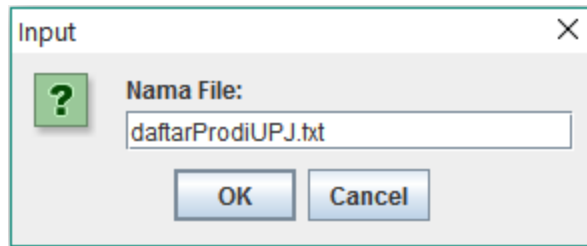
Langkah 8: Buat file teks dengan nama: **daftarNama.txt** pada project anda.

Caranya:

- click kanan pada project → new → Untitled Teks File.
- Tulis isinya dengan nama teman anda (satu baris untuk tiap satu nama).
- Click Save → pilih project anda → tulis nama file: daftarNama.txt

Langkah 9: Buat class dengan nama: **lab_3_FileReader**.

Langkah 10: Tambahkan kode program sehingga meminta user memasukkan nama file teks, dengan tampilan seperti berikut:



kemudian nama file teks masukan user tersebut dibaca oleh program anda, dan isi file teks tersebut ditampilkan baris demi baris.

Langkah 11: Buat tampilan output program anda agar memiliki tampilan hasil pada konsol seperti contoh berikut:

Daftar Prodi di UPJ

**SIF
KOM
TIF
DKV
ARS
MGT
TSI
AKT
PSI
DPI**

Analisa Hasil Praktek

Berdasarkan praktek yang telah Anda lakukan, buat kesimpulan:

Tugas

6. Buat program yang akan membaca isi file teks, kemudian program tersebut menghitung jumlah huruf tertentu (sesuai keinginan user).
7. Buat program pengolahan String yang menggunakan method **startsWith()** pada class String. Mengenai penggunaan method **startsWith()** anda cari informasi-nya pada text book atau internet.

Appendix 1: Form Data Praktek

Nama :

NIM :

Tgl :

Penggalan <i>Source code</i>	Keluaran / <i>Output</i>

Bahasa Pemrograman

(Pemrograman Visual dengan Java)

Modul Praktikum

Topik

Penggunaan java Swing JDesktopPane (Swing Control) dan JInternalFrame (swing containers) pada aplikasi dengan menggunakan Netbeans IDE.

Tujuan

Setelah menyelesaikan praktikum ini, mahasiswa diharapkan mampu:

11. Membuat aplikasi GUI dengan menggunakan JDesktopPane dengan Netbeans IDE.
12. Membuat aplikasi GUI dengan menggunakan JDesktopPane untuk menjalankan JDialog di dalamnya dengan Netbeans IDE.
13. Membuat aplikasi GUI dengan menggunakan JDesktopPane untuk menjalankan JInternalFrame di dalamnya dengan Netbeans IDE.
14. Membuat aplikasi GUI dengan menggunakan JDesktopPane untuk menjalankan JInternalFrame yang dibuat pada class terpisah di dalamnya dengan Netbeans IDE

Alat dan Bahan

- Komputer dengan **Java Development Kit (JDK)** dan Netbeans IDE sudah terinstall
- Informasi download dan instalasi JDK dan Netbeans dapat diperoleh di;
<https://netbeans.org/downloads/index.html>

Rangkuman/Teori

[Menjelaskan cara membuat internal Frame (frame di dalam frame) pada aplikasi menggunakan JDesktopPane dan JInternalFrame dengan Netbeans IDE]

JDesktopPane

JDesktopPane Class digunakan untuk membuat multiple-document interface (MDI) atau virtual desktop. JDesktopPane biasa digunakan sebagai parent (induk) dari JInternalFrame, JDesktopPane berfungsi sebagai Desktop anager yang memfasilitasi pemasangan atau JInternalFrame di dalamnya.

JInternalFrame

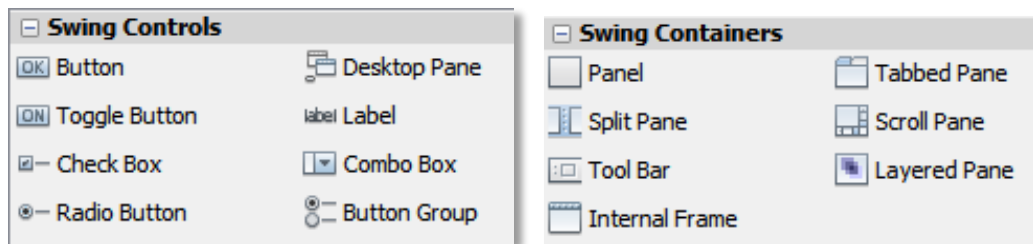
JInternalFrame merupakan lightweight frame (frame sederhana) yang biasanya digunakan di dalam JDesktopPane.

Terdapat dua cara untuk menggunakan JInternalFrame

- Menggunakan **Internal Frame dalam file class yang sama dengan desktopPane**. Biasanya cukup dengan menambahkan JInternalFrame (Internal Frame) (dari palette → swing containers → Internal Frame) ke dalam desktopPane.
- Menggunakan **Internal Frame dari file class terpisah dari file class tempat desktopPane**, dengan membuat suatu file class dengan jenis JInternalFrame secara terpisah kemudian class jenis JInternalFrame tersebut dipanggil untuk tampil dan digunakan di dalam desktopPane.

Membuat Aplikasi dengan Menggunakan JInternalFrame dalam File Class yang Sama dengan File Class Tempat JDesktopPane

Untuk membuat JInternalFrame di dalam desktopPane pada aplikasi berbasis java biasanya digunakan widget-widget berikut:



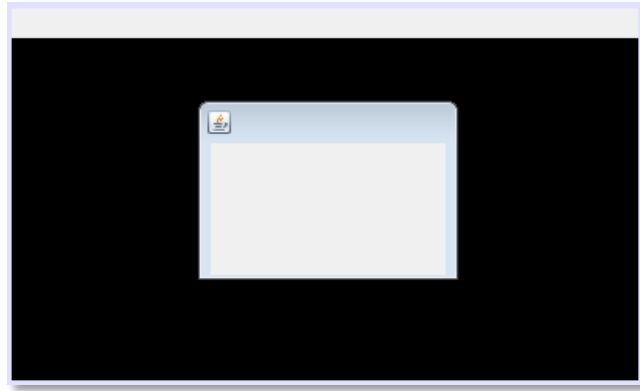
Gambar 1. Widget Desktop Pane dan Widget Internal Frame.

1. Masukkan Desktop Pane ke Frame Aplikasi (click & drag Desktop Pane dari Swing Controls ke Frame Aplikasi).



Gambar 2. JDesktopPane di dalam JFrame aplikasi.

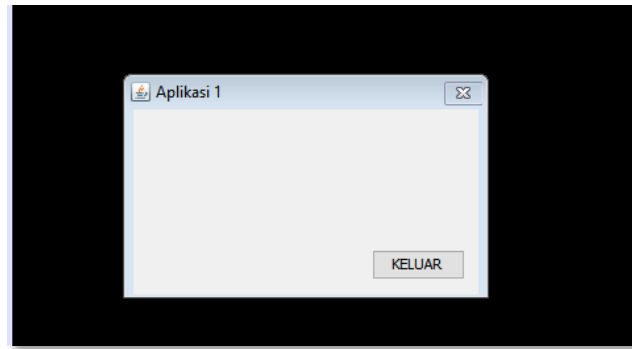
2. Ubah variabel name **JDesktopPane** (pada tab code properties) sesuai kegunaan (misal: **variable name: dpUtama**)
3. Masukkan Internal Frame ke JDesktopPane (click & drag Internal Frame dari Swing Containers ke desktop), Atur ukuran InternalFrame sesuai kebutuhan.



Gambar 3. JInternalFrame di dalam JDesktopPane

4. Ubah variabel name **JInternalFrame** (pada tab code properties) sesuai kegunaan (misal: **variable name: ifrAplikasi1**).
5. Ubah title JInternalFrame (pada tabel properties) sesuai kegunaan (misal: **Title: Aplikasi 1**)
6. Tambahkan Button ke JInternalFrame untuk tombol keluar dari JInternalFrame.
7. Ubah variabel name JButton (pada tab code properties) sesuai kegunaan (misal: **variable name: btnKeluar**).
8. Ubah text pada Button (pada tabel properties) sesuai kegunaan (misal: **Text: KELUAR**)
9. Agar **btnKeluar** dapat berfungsi menutup internalFrame saat di click, tambahkan kode program untuk menutup ifrAplikasi1 (JInternalFrame) pada btnKeluar (Jbutton) dengan cara: *right-click* btnKeluar → Events → Action → actionPerformed

```
private void btnKeluarActionPerformed(java.awt.event.ActionEvent evt) {  
    ifrAplikasi1.dispose(); //instruksi untuk menutup ifrAplikasi1 (JInternalFrame)  
}
```



Gambar 4. JInternalFrame dengan button keluar.

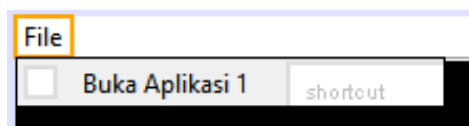
Secara default JInternalFrame akan tampil pada desktop saat aplikasi dijalankan, untuk itu kita harus menutupnya saat aplikasi tampil (componentShown) pertama kali dengan cara seperti pada langkah berikut:

10. Buat kode program untuk membuat JInternalFrame ditutup saat ComponentShown, melalui **navigator** *right-click* JFrame → Events → Component → componentShown.

Kemudian tulis instruksi yang diinginkan pada area kode tersebut, misal seperti berikut:

```
private void formComponentShown(java.awt.event.ComponentEvent evt) {
    ifrAplikasi1.dispose(); //instruksi untuk menutup ifrAplikasi1 (JInternalFrame)
}
```

11. Tambahkan JMenuBar dan JMenuItem ke JFrame sebagai sarana untuk membuka ifrAplikasi1 (JInternalFrame).
12. Ubah text JMenuItem (pada tab properties) sesuai kegunaan (misal: **Text: Buka Aplikasi 1**)



Gambar 5. Menu pada aplikasi

13. Ubah variabel name JMenuItem (pada tab code properties) sesuai kegunaan (misal: **variable name: mnuBukaAplikasi1**).
14. Agar **mnuBukaAplikasi1** dapat berfungsi untuk membuka ifrAplikasi1 (JInternalFrame) saat di click, tambahkan kode program pada mnuBukaAplikasi1 (JMenuItem) dengan cara: *right-click* mnuBukaAplikasi1 → Events → Action → actionPerformed

```
private void mnuBukaAplikasi1ActionPerformed(java.awt.event.ActionEvent evt) {
    if( ! ( ifrAplikasi1.isVisible() ) ) { // Memeriksa apakah ifrAplikasi1 sedang tidak tampil
```

```

        this.dpUtama.add(ifrAplikasi1); // Memasukkan ifrAplikasi1 ke desktopPane
        ifrAplikasi1.setVisible(true); // Menampilkan ifrAplikasi1
    }
    else {
        JOptionPane.showMessageDialog(this, "Aplikasi 1 sudah dibuka");
    }
}
}

```

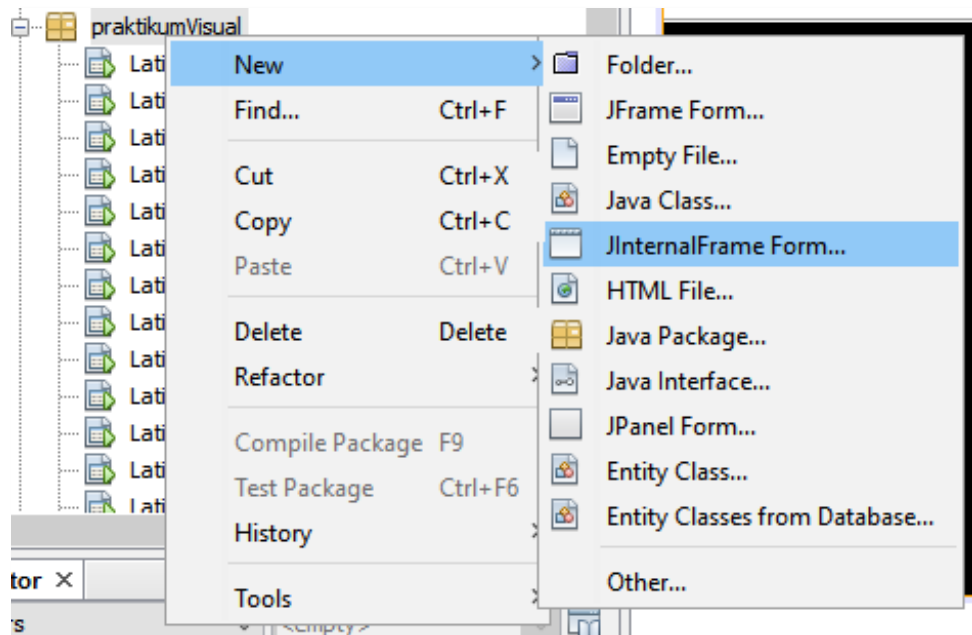
15. Selesai

Membuat Aplikasi dengan Menggunakan JInternalFrame dari File Class yang Terpisah dari File Class Tempat JDesktopPane

Membuat File Class Aplikasi1 jenis JInternalFrame

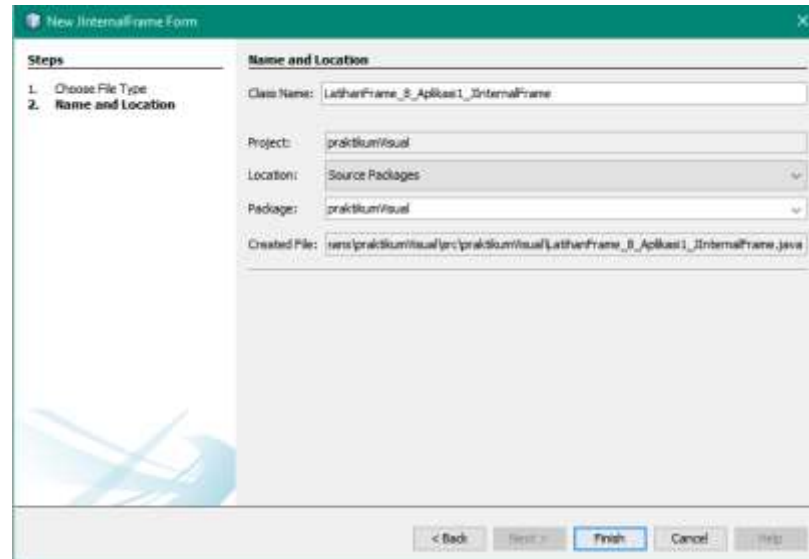
Untuk membuat file class Aplikasi1 dengan jenis JInternalFrame yang nanti akan digunakan di dalam aplikasi dengan JDesktopPane, gunakan langkah-langkah berikut:

1. Pada package tempat aplikasi utama (aplikasi dengan JDesktopPane), **right-click** → **New** → **JInternalFrame Form**



Gambar 6. Mambuat Class jenis JInternalFrame.

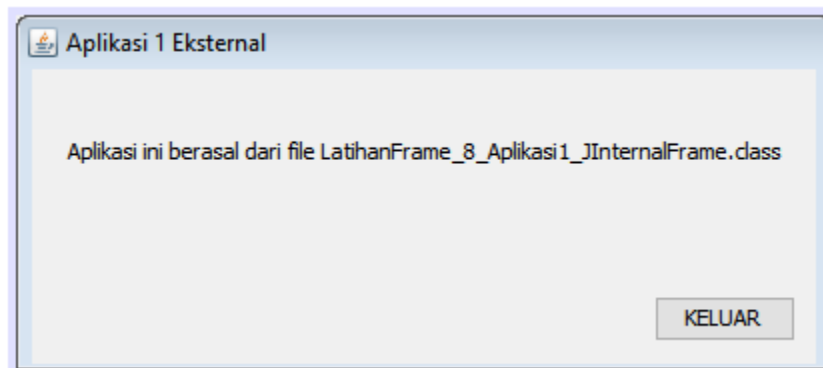
2. Pada Class Name isi dengan nama class yang diinginkan (sesuaikan dengan kegunaannya), pada contoh Class Name: **LatihanFrame_8_Aplikasi1_JInternalFrame**



Gambar 7. Form Pembuatan Class Jenis JInternalFrame

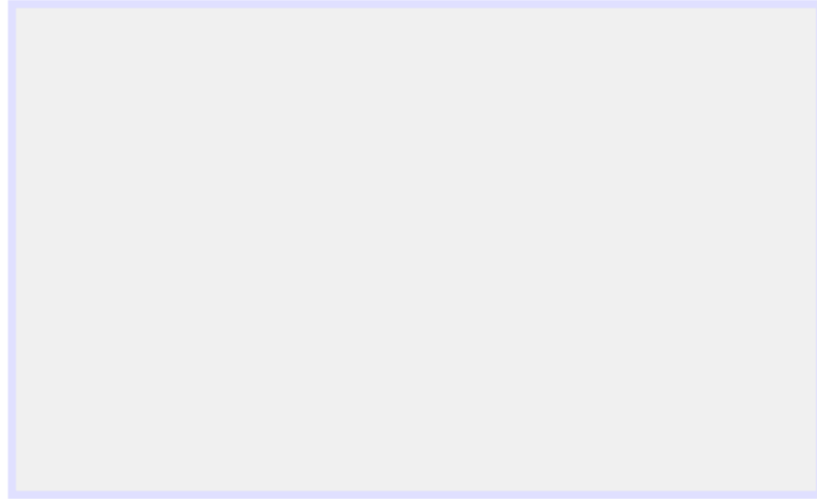
3. Ubah title **JInternalFrame** (pada tab properties) sesuai kegunaan (misal: **Title: Aplikasi 1**)
4. Tambahkan widget yang diperlukan sesuai kebutuhan.
5. Tambahkan JButton untuk menutup Aplikasi 1, pada contoh setting JButton:
 - a. Variable name: **btnKeluar**
 - b. Title: **KELUAR**
6. Tambahkah kode program pada btnKeluar **btnKeluar** dapat berfungsi menutup Aplikasi 1 saat di click: *right-click* btnKeluar → Events → Action → actionPerformed

```
private void btnKeluarActionPerformed(java.awt.event.ActionEvent evt) {  
    this.dispose(); //instruksi untuk menutup Aplikasi 1  
}
```



Gambar 8. Tampilan Class **LatihanFrame_8_Aplikasi1_JInternalFrame**

7. Membuat Desktop pada aplikasi utama: Buka file class aplikasi utama (*double-click* nama class).



Gambar 9. Frame Aplikasi Utama

8. Masukkan Desktop Pane ke Frame Aplikasi Utama (click & drag Desktop Pane dari Swing Controls ke Frame Aplikasi).



Gambar 10. JDesktopPane di dalam JFrame aplikasi.

9. Ubah variabel name **JDesktopPane** (pada tab code properties) sesuai kegunaan (misal: **variable name: dpUtama**)
10. Membuat object (instant) sebagai representasi File Class eksternal pada Aplikasi Utama dengan cara:
- Membuat Field object (instant) dari class Aplikasi 1 (yaitu: **LatihanFrame_8_Aplikasi2_JInternalFrame**).
 - Mengaktifkan object (instant) yang sudah dibuat pada Frame Utama, dengan membuat kode program pada Frame ComponentShown

Contoh:

- a. *Click Source* kemudian pada **navigator class** *double-click* nama class utama (yang paling atas).

Pointer (cursor) akan mengarah ke deklarasi class utama, kemudian di bawah tulisan public class buat Field Object dengan kode berikut:

```
Private LatihanFrame_8_Aplikasi2_JInternalFrame ifrAplikasi2;
```

Lihat pada gambar di bawah ini:

```
public class LatihanFrame_8_jDeskTopPaneMenuOpenExternalClass extends javax.swing.JFrame {  
    //field  
    private LatihanFrame_8_Aplikasi1_JInternalFrame ifrAplikasi1;  
}
```

Gambar 11. Pembuatan field object ifrAplikasi1 sebagai representasi class eksternal

- b. Pada Frame *right-click* → Event → Component → componentShown , kemudian tambahkan kode program seperti berikut:

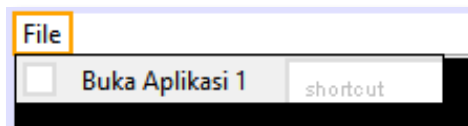
```
ifrAplikasi2 = new LatihanFrame_8_Aplikasi2_JInternalFrame();
```

Lihat Gambar di bawah ini:

```
private void formComponentShown(java.awt.event.ComponentEvent evt) {  
    ifrAplikasi1 = new LatihanFrame_8_Aplikasi1_JInternalFrame();  
}
```

Gambar 12. Mengaktifkan object (instant) ifrAplikasi1

11. Tambahkan JMenuBar dan JMenuItem ke JFrame sebagai sarana untuk membuka ifrAplikasi1 (JInternalFrame).
12. Ubah text JMenuItem (pada tab properties) sesuai kegunaan (misal: **Text: Buka Aplikasi 1**)



Gambar 5. Menu pada aplikasi

13. Ubah variabel name JMenuItem (pada tab code properties) sesuai kegunaan (misal: **variable name: mnuBukaAplikasi1**).
14. Agar **mnuBukaAplikasi1** dapat berfungsi untuk membuka ifrAplikasi1 (JInternalFrame) saat di click, tambahkan kode program pada mnuBukaAplikasi1 (JMenuItem) dengan cara: *right-click* mnuBukaAplikasi1 → Events → Action → actionPerformed

```

private void mnuBukaAplikasi1ActionPerformed(java.awt.event.ActionEvent evt) {
    // Membuat object ifrAplikasi1 dari class eksternal yang sudah kita buat:
    // LatihanFrame_8_jInternalFrame_1
    if ( ! ( ifrAplikasi1.isVisible() ) ) { // Memeriksa apakah ifrAplikasi1 sedang tidak tampil
        this.dpUtama.add(ifrAplikasi1); // Memasukkan ifrAplikasi1 ke desktopPane
        ifrAplikasi1.setVisible(true); // Menampilkan ifrAplikasi1
    }
    else {
        JOptionPane.showMessageDialog(this, "Aplikasi 1 sudah dibuka");
    }
}
}

```

15. Selesai

Analisa Hasil Praktek

Berdasarkan praktek yang telah Anda lakukan, buat kesimpulan:

Appendix 1: Form Data Praktek

Nama :

NIM :

Tgl :

Penggalan <i>Source code</i>	Keluaran / <i>Output</i>
------------------------------	--------------------------

--	--

Bahasa Pemrograman (Pemrograman Visual dengan Java) Modul Praktikum

Topik

Penggunaan java Swing JDesktopPane (Swing Control) dan JInternalFrame (swing containers) pada aplikasi dengan menggunakan Netbeans IDE.

Tujuan

Setelah menyelesaikan praktikum ini, mahasiswa diharapkan mampu:

15. Membuat aplikasi GUI dengan menggunakan JDesktopPane dengan Netbeans IDE.
16. Membuat aplikasi GUI dengan menggunakan JDesktopPane untuk menjalankan JDialog di dalamnya dengan Netbeans IDE.
17. Membuat aplikasi GUI dengan menggunakan JDesktopPane untuk menjalankan JInternalFrame di dalamnya dengan Netbeans IDE.
18. Membuat aplikasi GUI dengan menggunakan JDesktopPane untuk menjalankan JInternalFrame yang dibuat pada class terpisah di dalamnya dengan Netbeans IDE

Alat dan Bahan

- Komputer dengan **Java Development Kit (JDK)** dan Netbeans IDE sudah terinstall
- Informasi download dan instalasi JDK dan Netbeans dapat diperoleh di;
<https://netbeans.org/downloads/index.html>

Rangkuman/Teori

[Menjelaskan cara membuat internal Frame (frame di dalam frame) pada aplikasi menggunakan JDesktopPane dan JInternalFrame dengan Netbeans IDE]

JDesktopPane

JDesktopPane Class digunakan untuk membuat multiple-document interface (MDI) atau virtual desktop. JDesktopPane biasa digunakan sebagai parent (induk) dari JInternalFrame, JDesktopPane berfungsi sebagai Desktop anager yang memfasilitasi pemasangan atau JInternalFrame di dalamnya.

JInternalFrame

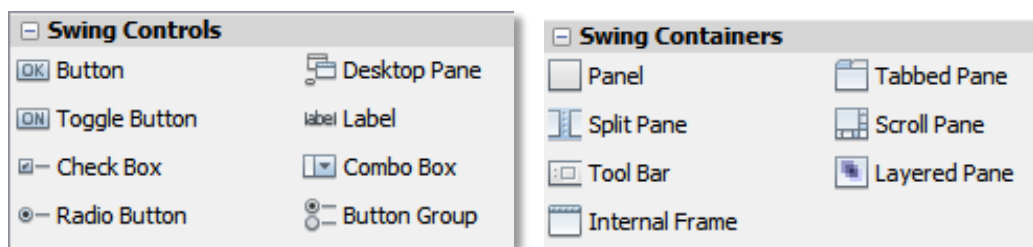
JInternalFrame merupakan lightweight frame (frame sederhana) yang biasanya digunakan di dalam JDesktopPane.

Terdapat dua cara untuk menggunakan JInternalFrame

- Menggunakan **Internal Frame dalam file class yang sama dengan desktopPane**. Biasanya cukup dengan menambahkan JInternalFrame (Internal Frame) (dari palette → swing containers → Internal Frame) ke dalam desktopPane.
- Menggunakan **Internal Frame dari file class terpisah dari file class tempat desktopPane**, dengan membuat suatu file class dengan jenis JInternalFrame secara terpisah kemudian class jenis JInternalFrame tersebut dipanggil untuk tampil dan digunakan di dalam desktopPane.

Membuat Aplikasi dengan Menggunakan JInternalFrame dalam File Class yang Sama dengan File Class Tempat JDesktopPane

Untuk membuat JInternalFrame di dalam desktopPane pada aplikasi berbasis java biasanya digunakan widget-widget berikut:



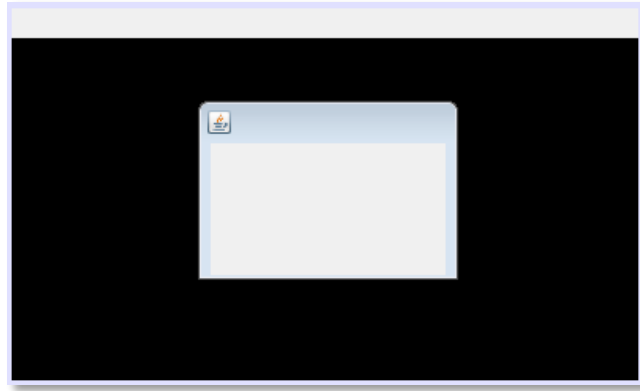
Gambar 1. Widget Desktop Pane dan Widget Internal Frame.

16. Masukkan Desktop Pane ke Frame Aplikasi (click & drag Desktop Pane dari Swing Controls ke Frame Aplikasi).



Gambar 2. JDesktopPane di dalam JFrame aplikasi.

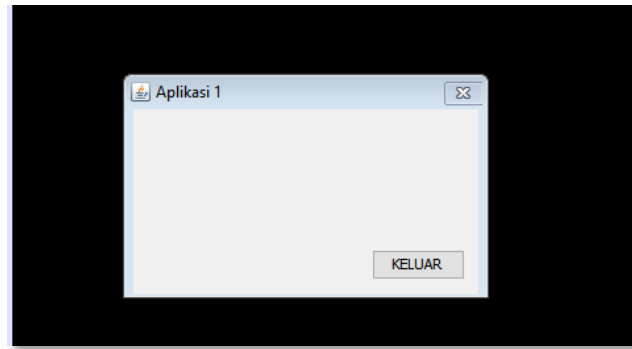
17. Ubah variabel name **JDesktopPane** (pada tab code properties) sesuai kegunaan (misal: **variable name: dpUtama**)
18. Masukkan Internal Frame ke JDesktopPane (click & drag Internal Frame dari Swing Containers ke desktop), Atur ukuran InternalFrame sesuai kebutuhan.



Gambar 3. JInternalFrame di dalam JDesktopPane

19. Ubah variabel name **JInternalFrame** (pada tab code properties) sesuai kegunaan (misal: **variable name: ifrAplikasi1**).
20. Ubah title JInternalFrame (pada tabel properties) sesuai kegunaan (misal: **Title: Aplikasi 1**)
21. Tambahkan Button ke JInternalFrame untuk tombol keluar dari JInternalFrame.
22. Ubah variabel name JButton (pada tab code properties) sesuai kegunaan (misal: **variable name: btnKeluar**).
23. Ubah text pada Button (pada tabel properties) sesuai kegunaan (misal: **Text: KELUAR**)
24. Agar **btnKeluar** dapat berfungsi menutup internalFrame saat di click, tambahkan kode program untuk menutup ifrAplikasi1 (JInternalFrame) pada btnKeluar (Jbutton) dengan cara: *right-click* btnKeluar → Events → Action → actionPerformed

```
private void btnKeluarActionPerformed(java.awt.event.ActionEvent evt) {  
    ifrAplikasi1.dispose(); //instruksi untuk menutup ifrAplikasi1 (JInternalFrame)  
}
```



Gambar 4. JInternalFrame dengan button keluar.

Secara default JInternalFrame akan tampil pada desktop saat aplikasi dijalankan, untuk itu kita harus menutupnya saat aplikasi tampil (componentShown) pertama kali dengan cara seperti pada langkah berikut:

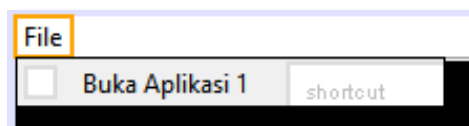
25. Buat kode program untuk membuat JInternalFrame ditutup saat ComponentShown, melalui **navigator** *right-click* JFrame → Events → Component → componentShown.

Kemudian tulis instruksi yang diinginkan pada area kode tersebut, misal seperti berikut:

```
private void formComponentShown(java.awt.event.ComponentEvent evt) {
    ifrAplikasi1.dispose(); //instruksi untuk menutup ifrAplikasi1 (JInternalFrame)
}
```

26. Tambahkan JMenuBar dan JMenuItem ke JFrame sebagai sarana untuk membuka ifrAplikasi1 (JInternalFrame).

27. Ubah text JMenuItem (pada tab properties) sesuai kegunaan (misal: **Text: Buka Aplikasi 1**)



Gambar 5. Menu pada aplikasi

28. Ubah variabel name JMenuItem (pada tab code properties) sesuai kegunaan (misal: **variable name: mnuBukaAplikasi1**).

29. Agar **mnuBukaAplikasi1** dapat berfungsi untuk membuka ifrAplikasi1 (JInternalFrame) saat di click, tambahkan kode program pada mnuBukaAplikasi1 (JMenuItem) dengan cara: *right-click* mnuBukaAplikasi1 → Events → Action → actionPerformed

```
private void mnuBukaAplikasi1ActionPerformed(java.awt.event.ActionEvent evt) {
    if( ! ( ifrAplikasi1.isVisible() ) ) { // Memeriksa apakah ifrAplikasi1 sedang tidak tampil
```

```

        this.dpUtama.add(ifrAplikasi1); // Memasukkan ifrAplikasi1 ke desktopPane
        ifrAplikasi1.setVisible(true); // Menampilkan ifrAplikasi1
    }
    else {
        JOptionPane.showMessageDialog(this, "Aplikasi 1 sudah dibuka");
    }
}

```

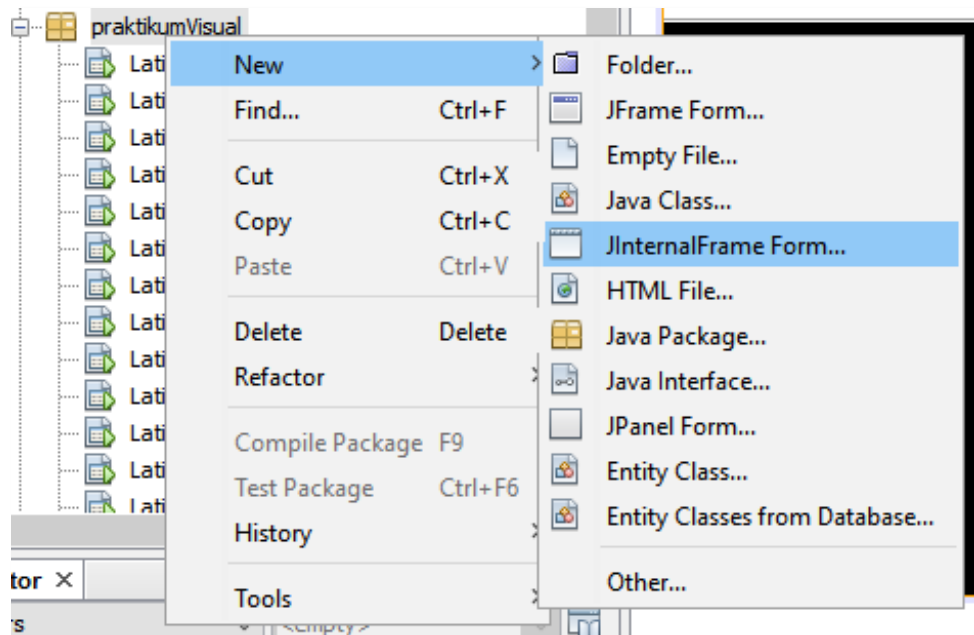
30. Selesai

Membuat Aplikasi dengan Menggunakan JInternalFrame dari [File Class yang Terpisah](#) dari File Class Tempat JDesktopPane

Membuat File Class Aplikasi1 jenis JInternalFrame

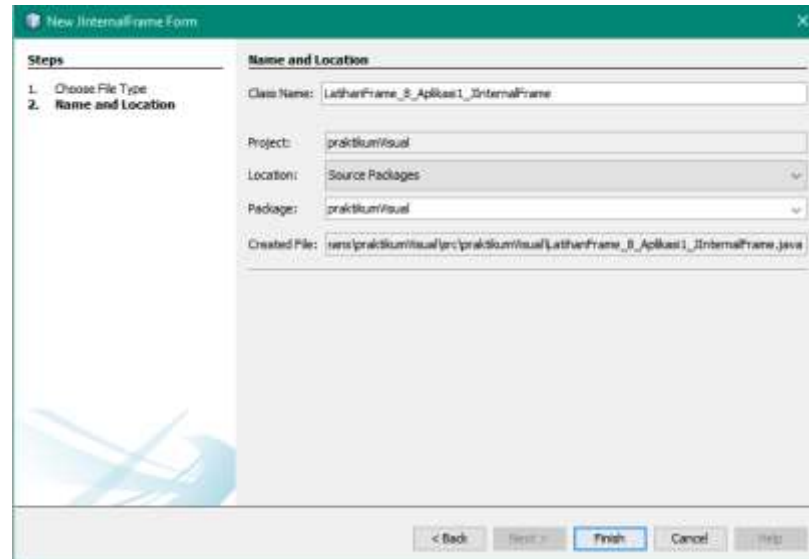
Untuk membuat file class Aplikasi1 dengan jenis JInternalFrame yang nanti akan digunakan di dalam aplikasi dengan JDesktopPane, gunakan langkah-langkah berikut:

16. Pada package tempat aplikasi utama (aplikasi dengan JDesktopPane), **right-click** → **New** → **JInternalFrame Form**



Gambar 6. Mambuat Class jenis JInternalFrame.

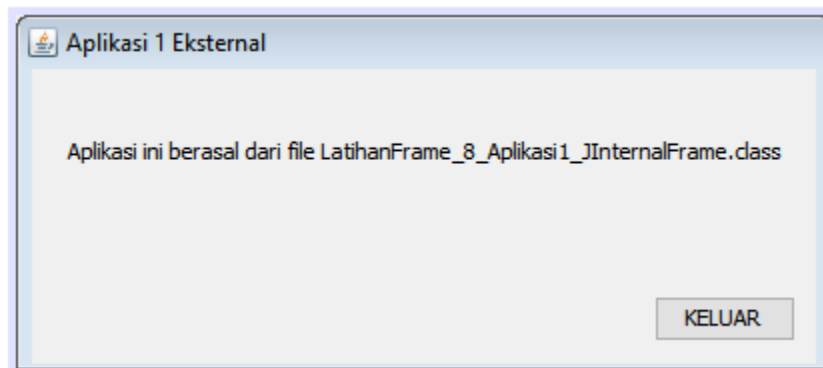
17. Pada Class Name isi dengan nama class yang diinginkan (sesuaikan dengan kegunaannya), pada contoh Class Name: **LatihanFrame_8_Aplikasi1_JInternalFrame**



Gambar 7. Form Pembuatan Class Jenis JInternalFrame

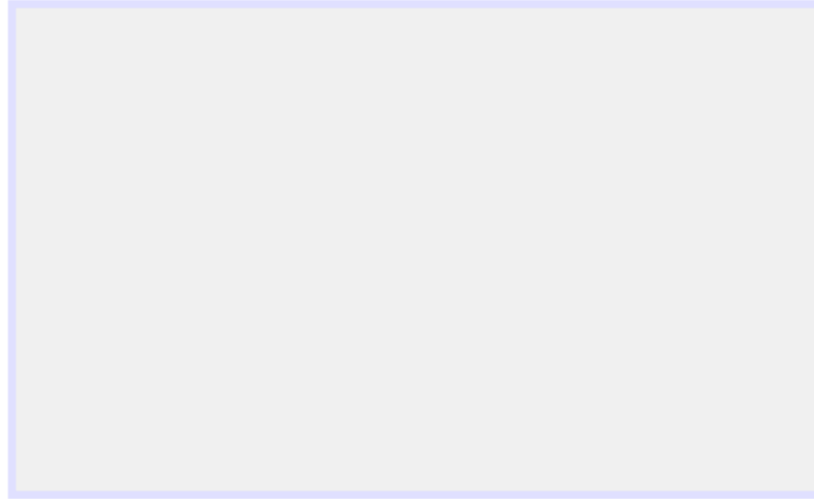
18. Ubah title **JInternalFrame** (pada tab properties) sesuai kegunaan (misal: **Title: Aplikasi 1**)
19. Tambahkan widget yang diperlukan sesuai kebutuhan.
20. Tambahkan JButton untuk menutup Aplikasi 1, pada contoh setting JButton:
- Variable name: **btnKeluar**
 - Title: **KELUAR**
21. Tambahkah kode program pada btnKeluar **btnKeluar** dapat berfungsi menutup Aplikasi 1 saat di click: *right-click* btnKeluar → Events → Action → actionPerformed

```
private void btnKeluarActionPerformed(java.awt.event.ActionEvent evt) {  
    this.dispose(); //instruksi untuk menutup Aplikasi 1  
}
```



Gambar 8. Tampilan Class **LatihanFrame_8_Aplikasi1_JInternalFrame**

22. Membuat Desktop pada aplikasi utama: Buka file class aplikasi utama (*double-click* nama class).



Gambar 9. Frame Aplikasi Utama

23. Masukkan Desktop Pane ke Frame Aplikasi Utama (click & drag Desktop Pane dari Swing Controls ke Frame Aplikasi).



Gambar 10. JDesktopPane di dalam JFrame aplikasi.

24. Ubah variabel name **JDesktopPane** (pada tab code properties) sesuai kegunaan (misal: **variable name: dpUtama**)

25. Membuat object (instant) sebagai representasi File Class eksternal pada Aplikasi Utama dengan cara:

- a. Membuat Field object (instant) dari class Aplikasi 1 (yaitu: **LatihanFrame_8_Aplikasi2_JInternalFrame**).
- b. Mengaktifkan object (instant) yang sudah dibuat pada Frame Utama, dengan membuat kode program pada Frame ComponentShown

Contoh:

- c. *Click Source* kemudian pada **navigator class** *double-click* nama class utama (yang paling atas).

Pointer (cursor) akan mengarah ke deklarasi class utama, kemudian di bawah tulisan public class buat Field Object dengan kode berikut:

```
Private LatihanFrame_8_Aplikasi2_JInternalFrame ifrAplikasi2;
```

Lihat pada gambar di bawah ini:

```
public class LatihanFrame_8_jDeskTopPaneMenuOpenExternalClass extends javax.swing.JFrame {  
    //field  
    private LatihanFrame_8_Aplikasi1_JInternalFrame ifrAplikasi1;  
}
```

Gambar 11. Pembuatan field object ifrAplikasi1 sebagai representasi class eksternal

- d. Pada Frame *right-click* → Event → Component → componentShown , kemudian tambahkan kode program seperti berikut:

```
ifrAplikasi2 = new LatihanFrame_8_Aplikasi2_JInternalFrame();
```

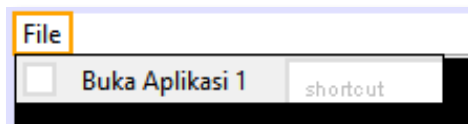
Lihat Gambar di bawah ini:

```
private void formComponentShown(java.awt.event.ComponentEvent evt) {  
    ifrAplikasi1 = new LatihanFrame_8_Aplikasi1_JInternalFrame();  
}
```

Gambar 12. Mengaktifkan object (instant) ifrAplikasi1

26. Tambahkan JMenuBar dan JMenuItem ke JFrame sebagai sarana untuk membuka ifrAplikasi1 (JInternalFrame).

27. Ubah text JMenuItem (pada tab properties) sesuai kegunaan (misal: **Text: Buka Aplikasi 1**)



Gambar 5. Menu pada aplikasi

28. Ubah variabel name JMenuItem (pada tab code properties) sesuai kegunaan (misal: **variable name: mnuBukaAplikasi1**).

29. Agar **mnuBukaAplikasi1** dapat berfungsi untuk membuka ifrAplikasi1 (JInternalFrame) saat di click, tambahkan kode program pada mnuBukaAplikasi1 (JMenuItem) dengan cara: *right-click* mnuBukaAplikasi1 → Events → Action → actionPerformed

```

private void mnuBukaAplikasi1ActionPerformed(java.awt.event.ActionEvent evt) {
    // Membuat object ifrAplikasi1 dari class eksternal yang sudah kita buat:
    // LatihanFrame_8_jInternalFrame_1
    if ( ! ( ifrAplikasi1.isVisible() ) ) { // Memeriksa apakah ifrAplikasi1 sedang tidak tampil
        this.dpUtama.add(ifrAplikasi1); // Memasukkan ifrAplikasi1 ke desktopPane
        ifrAplikasi1.setVisible(true); // Menampilkan ifrAplikasi1
    }
    else {
        JOptionPane.showMessageDialog(this, "Aplikasi 1 sudah dibuka");
    }
}
}

```

30. Selesai

Analisa Hasil Praktek

Berdasarkan praktek yang telah Anda lakukan, buat kesimpulan:

Appendix 1: Form Data Praktek

Nama :

NIM :

Tgl :

Penggalan <i>Source code</i>	Keluaran / <i>Output</i>
------------------------------	--------------------------

--	--

Bahasa Pemrograman (Pemrograman Visual dengan Java) Modul Praktikum

Topik

Penggunaan java Swing Menu (menu bar dan popup menu), JDialog (swing window pada aplikasi dengan menggunakan Netbeans IDE.

Tujuan

Setelah menyelesaikan praktikum ini, mahasiswa diharapkan mampu:

19. Membuat aplikasi GUI dengan menggunakan JMenuBar, JMenu dengan Netbeans IDE.
20. Membuat aplikasi GUI dengan menggunakan JPopupMenu dengan Netbeans IDE.
21. Membuat aplikasi GUI dengan menggunakan JDialog dengan Netbeans IDE
22. Membuat aplikasi GUI dengan menggunakan Menu Bar untuk memanggil JDialog dengan Netbeans IDE.
23. Membuat aplikasi GUI dengan menggunakan Popup Menu untuk memanggil JDialog dengan Netbeans IDE.

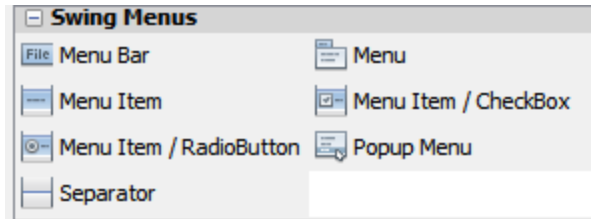
Alat dan Bahan

- Komputer dengan **Java Development Kit (JDK)** dan Netbeans IDE sudah terinstall
- Informasi download dan instalasi JDK dan Netbeans dapat diperoleh di;
<https://netbeans.org/downloads/index.html>

Rangkuman/Teori

[Menjelaskan cara membuat menu pada aplikasi dengan Netbeans IDE]

Untuk membuat menu pada aplikasi berbasis java biasanya digunakan widget-widget pada swing menu sebagai berikut:



Gambar 1. Widget swing menus

Pembuatan menu pada aplikasi bisa dibuat dengan melakukan kombinasi dari widget-widget swing menu seperti dapat dilihat pada gambar 1.

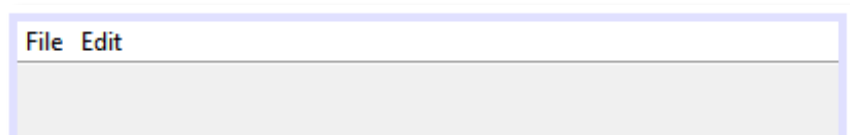
Terdapat 2 jenis menu yang dapat dibuat melalui Swing Menu, yaitu:

- **Jenis Menu Bar**
Menu Bar merupakan object menu yang melekat pada window aplikasi biasanya berada di bagian atas pada window aplikasi, pada Menu Bar biasanya ditambahkan Menu atau Menu Item sebagai isi atau sub-menu dari Menu bar.
- **Jenis Popup Menu**
Popup Menu merupakan object menu yang secara dinamic popup (tampil) pada posisi tertentu di dalam window aplikasi, pada Popup Menu biasanya ditambahkan Menu atau Menu Item sebagai isi atau sub-menu dari Popup Menu.

Membuat menu dengan Menu Bar

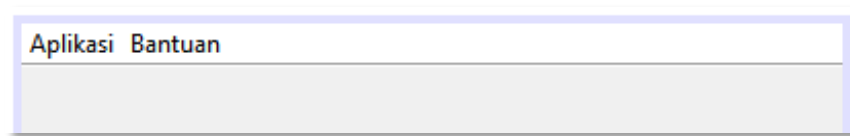
Untuk membuat menu dengan Menu Bar:

1. Masukkan Menu Bar ke Frame, (saran: letakkan di sisi atas Frame)



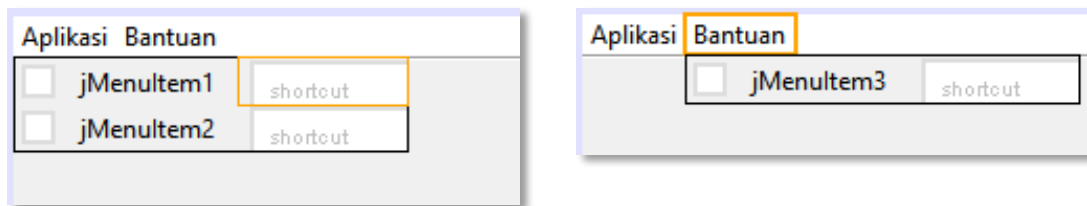
Gambar 2. Menu bar pada frame

2. Ubah variabel name (pada tab code properties) sesuai kegunaan (misal: variable name: mnuBar)
3. Ubah Text tiap-tiap JMenu (*right-click* pada tulisan di menu bar → pilih Edit Text) sesuai kegunaan. Misal perubahannya seperti berikut:



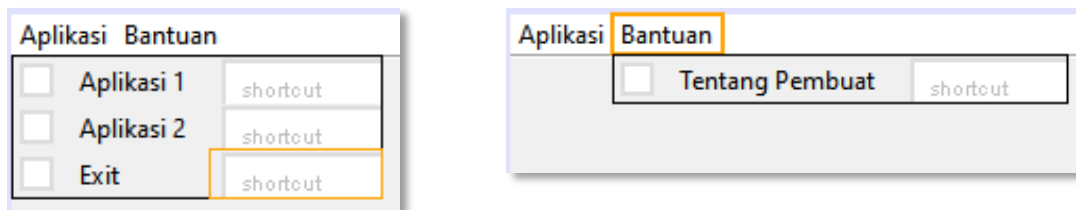
Gambar 3. Menu bar setelah di *edit text*

4. Ubah variable name untuk tiap-tiap JMenu sesuai teks yang tampil pada Menu Bar (untuk kemudahan mengingat saat membuat kode program), misal: **mnuAplikasi** untuk menu Aplikasi, dan **mnuBantuan** untuk menu Bantuan.
5. Tambahkan Menu Item ke JMenu untuk tiap-tiap JMenu sesuai kebutuhan aplikasi, misalkan seperti berikut:



Gambar 4. Menu bar beserta menu item

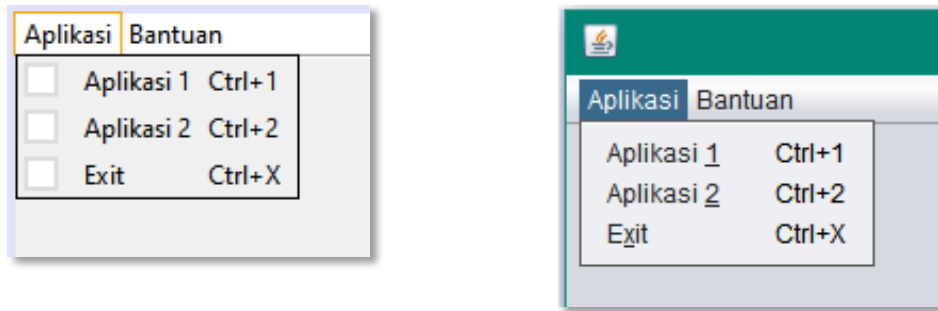
6. Ubah Text tiap-tiap JMenuItem (*right-click* pada tulisan di menu item → pilih Edit Text) sesuai kegunaan. Misal perubahannya seperti berikut:



Gambar 5. Menu item setelah di *edit text*

7. Ubah variable name untuk tiap-tiap JMenuItem sesuai teks yang tampil pada Menu Item (untuk kemudahan mengingat saat membuat kode program), misal: **mnuitemAplikasi1** untuk menu item Aplikasi 1, **mnuitemAplikasi2** untuk menu item Aplikasi 2 dan seterusnya.
8. Tambahkan shortcut jika diperlukan.
Untuk shortcut terdapat dua jenis:
 - a. **Accelerator**: Shortcut yang bisa langsung tanpa membuka menu (biasanya dengan kombinasi fungsi Ctrl).
Untuk membuatnya: click **menu item** yang ingin dibuatkan shortcutnya, kemudian pada **tab properties** → Accelerator: **Ctrl+CharYangDiinginkan**
Misal: Accelerator: Ctrl+x , untuk menu item Exit
 - b. **Mnemonic**: Shortcut yang berfungsi setelah menu dibuka (biasanya dengan langsung menekan tombol sesuai karakter yang di-underline)
Untuk membuatnya: click menu item yang ingin dibuatkan shortcutnya, kemudian pada **tab properties** → Mnemonic: **CharYangDiinginkan**
Misal: Mnemonic: x , untuk menu item Exit

Misal seperti berikut:



Gambar 6. Design dan tampilan aplikasi Menu item dengan shortcut

9. Untuk memberikan instruksi pada menu item:

Click menu Item → *Right-click* menu item → Events → Action → actionPerformed

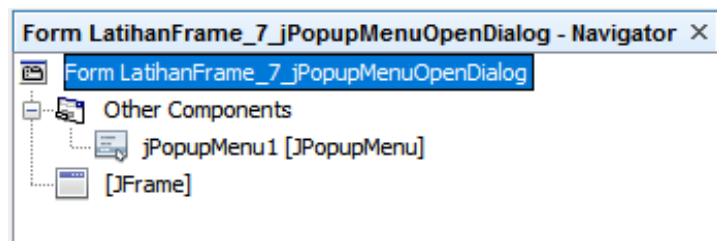
kemudian tulis instruksi yang diinginkan pada area kode tersebut, misal kodenya seperti berikut:

```
private void mnulItemExitActionPerformed(java.awt.event.ActionEvent evt) {  
    System.exit(0); //instruksi untuk exit window aplikasi  
}
```

Membuat menu dengan Popup Menu

Untuk membuat menu dengan Popup Menu:

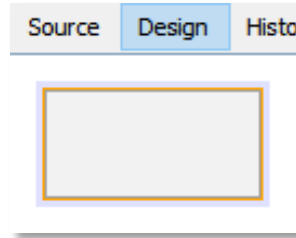
1. Masukkan Popup Menu ke Frame (jPopupMenu tidak akan terlihat)
2. Untuk mengakses dan melakukan setting terhadap JpopupMenu gunakan **navigator** class (karena JPopupMenu tidak terlihat pada JFrame) dan bahkan di navigator terlihat bahwa JPopupMenu masuk ke **other components** (tidak masuk ke JFrame).



Gambar 7. JPopupMenu di dalam *other components* pada *navigator*

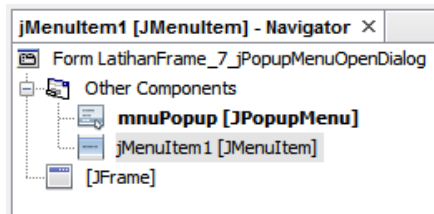
3. Untuk melihat PopupMenu, *double-click* pada jPopupMenu1 sehingga tulisan jPopupMenu1 menjadi bold.
4. Ubah variabel name (pada tab code properties) sesuai kegunaan (misal: **variable name: mnuPopup**)

5. Ubah ukuran Popup Menu (**preferred size** dan **Minimum Size**) *sehingga* layak dilihat (karena default-nya [0,0] sangat kecil), misal **preferred size: [100, 50]** dan **Minimum Size: [100, 50]**



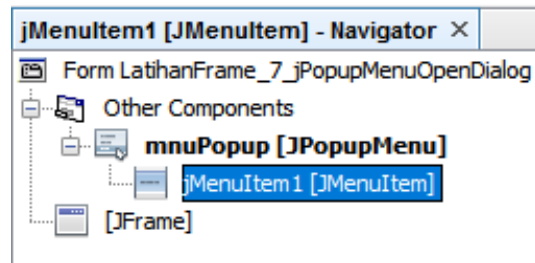
Gambar 8. Popup Menu terlihat dengan ukuran [100, 50]

6. Masukkan JMenuItem ke Popup Menu (JMenuItem tidak akan terlihat), untuk itu gunakan **Navigator class**.



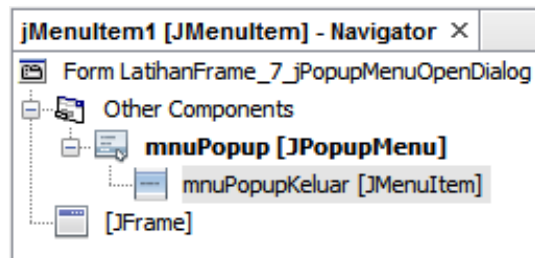
Gambar 9. jMenuItem pada Other Components

7. Pada Navigator, *click & drag* jMenuItem1 ke mnuPopup [JpopupMenu] sehingga jMenuItem sekarang berada di dalam mnuPopup [JpopupMenu] .



Gambar 10. jMenuItem berada dalam mnuPopup [JpopupMenu]

8. *Click* pada jMenuItem1 kemudian pada tab *code* ubah variable name sesuai fungsi dari menu item, misal; **Variable Name: mnuPopupKeluar**.



Gambar 11. JMenuItem dengan nama **mnuPopupKeluar** di dalam mnuPopup [JpopupMenu]

9. Pada mnuPopupKeluar Ubah Teksnya (pada tab properties → Text) sesuaikan dengan teks yang akan tampil pada popup menu, misal: **Text: Exit**
10. Ulangi langkah 6 – 9 untuk menambahkan menu item lainnya.
11. Untuk memberikan instruksi pada menu item (misal pada **mnuPopupKeluar**):
Click mnuPopupKeluar (pada navigator class) → Right-click mnuPopupKeluar → Events → Action → actionPerformed

Kemudian tulis instruksi yang diinginkan pada area kode tersebut, misal seperti berikut:

```
private void mnuPopupKeluarActionPerformed(java.awt.event.ActionEvent evt) {  
    System.exit(0); //instruksi untuk exit window aplikasi  
}
```

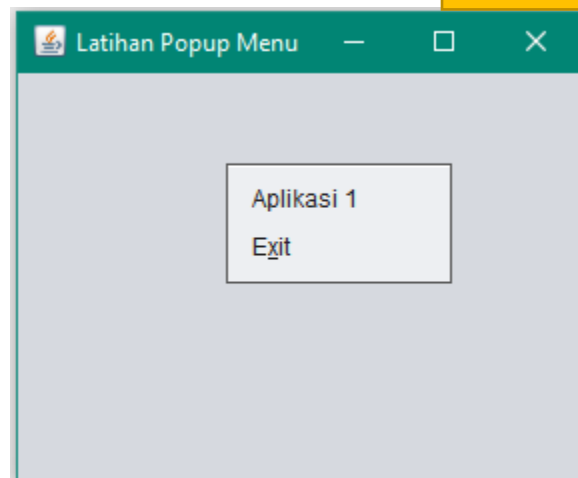
12. Untuk memberikan kode agar Popup Menu bisa tampil saat pengguna *right-click* pada frame
Right-click pada JFrame → Events → Mouse → MouseClicked

Kemudian tulis instruksi yang diinginkan pada area kode tersebut, misal seperti berikut:

```
private void formMouseClicked(java.awt.event.MouseEvent evt) {  
    if(evt.getButton()==MouseEvent.BUTTON3) {  
        mnuPopup.show(this, evt.getX(), evt.getY());  
    }  
}
```

memeriksa apakah right-click

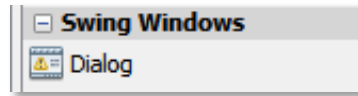
menampilkan popup menu
dengan koordinat sesuai dengan
posisi pointer



Gambar 12. Popup Menu tampil saat di *right-click* di dalam Frame

[Menjelaskan cara membuat dialog pada aplikasi dengan Netbeans IDE]

Untuk membuat dialog pada aplikasi berbasis java biasanya digunakan widget `JDialog` pada swing windows.

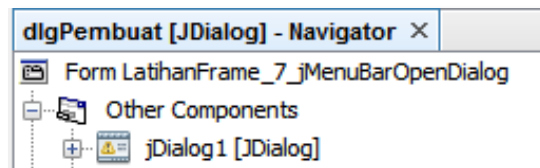


Gambar 13. `JDialog` pada swing windows.

Membuat dialog dengan `JDialog`

Untuk membuat dialog dengan Menggunakan `JDialog`:

1. Masukkan `JDialog` ke Frame (`JDialog` tidak akan terlihat).
2. Untuk mengakses dan melakukan setting terhadap `JDialog` gunakan **navigator** class, karena `JDialog` tidak terlihat dan bahkan di navigator terlihat bahwa `JDialog` masuk ke **other components** (tidak masuk ke `JFrame`).



Gambar 14. `JDialog` di dalam *other components* pada *navigator*

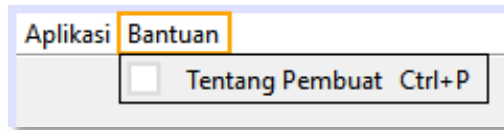
3. Untuk melihat dialog, *double-click* pada `JDialog1` sehingga tulisan `JDialog1` menjadi bold.
4. Ubah variabel name (pada tab code properties) sesuai kegunaan (misal: **variable name: `dlgPembuat`** , untuk membuat dialog tentang pembuat)
5. Ubah title (pada tab properties) sesuai judul dari dialog yang kita buat (misal: **Title: Tentang Pembuat**)
6. Ubah ukuran dialog (**preferred size** dan **Minimum Size**) *sehingga* layak dilihat (karena default-nya [0,0] sangat kecil), misal **preferred size: [290, 220]** dan **Minimum Size: [290, 220]**
7. Tambahkan widget button pada dialog, kemudian ubah **variable name: `btnOk`** dan **Text: OK**.
Untuk membuat instruksi pada `btnOk`:
right-click `btnOk` → events → Action → `actionPerformed`

Kemudian tulis instruksi yang diinginkan pada area kode tersebut, misal seperti berikut:

```
private void btnOkActionPerformed(java.awt.event.ActionEvent evt) {  
    dlgPembuat.dispose(); //instruksi untuk menutup dlgPembuat  
}
```

}

8. Untuk mengaktifkan `dlgPembuat` dari menu (Bantuan → Tentang Pembuat);



Gambar 15. Menu item Tentang Pembuat

Click kemudian *right-click* menu item Tentang Pembuat → events → Action → `actionPerformed`

Kemudian tulis instruksi yang diinginkan pada area kode tersebut, misal seperti berikut:

```
private void mnulItemPembuatActionPerformed(java.awt.event.ActionEvent evt) {  
    dlgPembuat.setLocationRelativeTo(this); //instruksi agar posisi dialog didalam window  
    dlgPembuat.setVisible(true); //instruksi menampilkan dlgPembuat  
}
```



Gambar 16. Tampilan `dlgPembuat` di dalam window aplikasi

Analisa Hasil Praktek

Berdasarkan praktek yang telah Anda lakukan, buat kesimpulan:

Tugas

9. Buat aplikasi perhitungan bangun yang akan memiliki dialog untuk fungsi sebagai berikut:
 - a. Program yang akan melakukan perhitungan luas segitiga, dan menampilkan hasilnya.
 - b. Program yang akan melakukan perhitungan luas lingkaran, dan menampilkan hasilnya.
 - c. Program yang akan melakukan perhitungan luas persegi, dan menampilkan hasilnya.
 - d. Program menampilkan tentang pembuat aplikasi.

Appendix 1: Form Data Praktek

Nama :

NIM :

Tgl :

Penggalan <i>Source code</i>	Keluaran / <i>Output</i>
