

LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
MODUL 5
EXCEPTION HANDLING



Oleh :

Tegar Ferdian Firmansyah

2311103123

S1SI-07-C

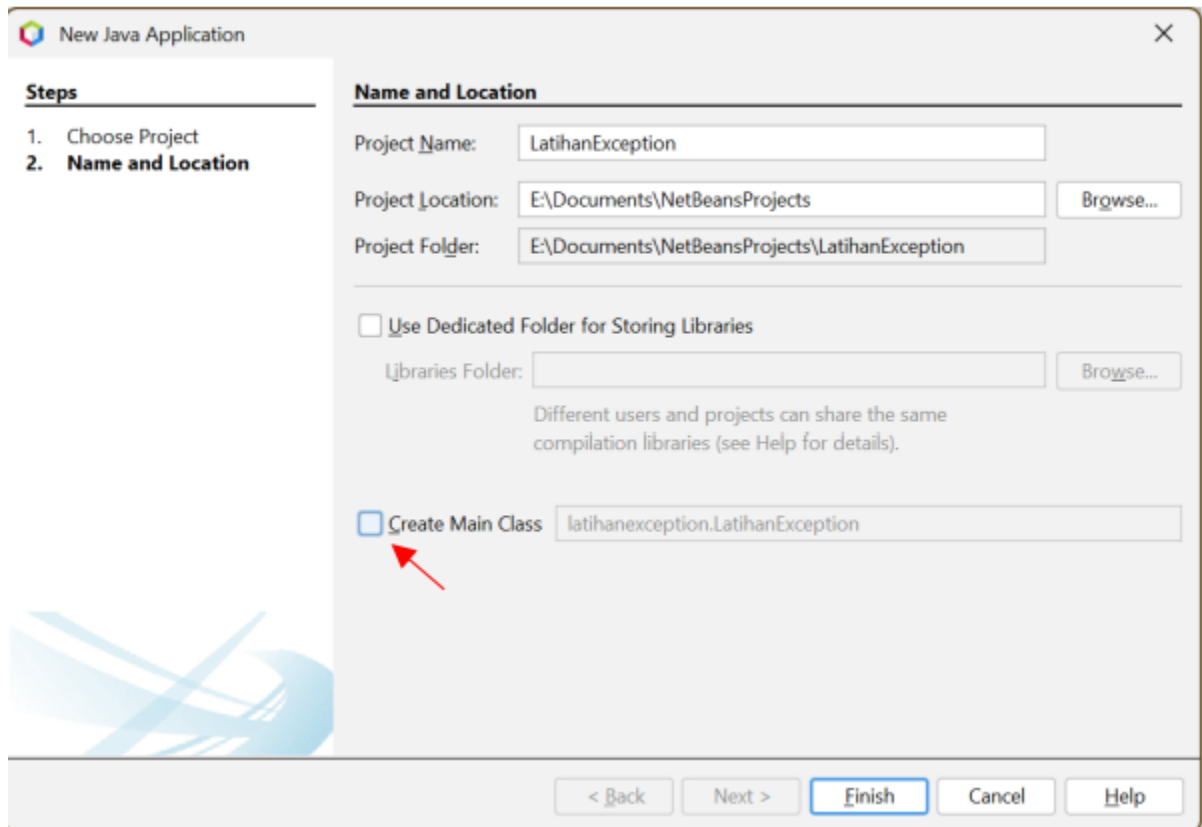
PROGRAM STUDI SISTEM INFORMASI
FAKULTAS REKAYASA INDUSTRI
UNIVERSITAS TELKOM PURWOKERTO
2024

A. Studi Kasus

Sistem Informasi Perpustakaan – Input Buku

B. GUIDED

1. Buat project baru pada Netbeans. Pada saat membuat project, hilangkan centang pada Create Main Class



2. Tambahkan package baru dengan nama perpustakaan dengan cara klik kanan pada Source Packages → New → Java Package...
3. Pada package perpustakaan, buat Class baru dengan nama Buku

Buku.java

```
package Perpustakaan;  
  
/**  
 * @author Tegar Ferdian Firmansyah
```

```
public class Buku {  
    private String judul;  
    private String pengarang;  
    private String isbn;  
  
    public Buku(String judul, String pengarang, String isbn){  
        this.judul = judul;  
        this.pengarang = pengarang;  
        this.isbn = isbn;  
    }  
  
    public String getJudul() {  
        return judul;  
    }  
  
    public String getPengarang() {  
        return pengarang;  
    }  
  
    public String getIsbn() {  
        return isbn;  
    }  
}
```

Penjelasan : Program di atas adalah implementasi kelas Buku dalam package Perpustakaan. Kelas ini berfungsi untuk merepresentasikan buku dalam sistem perpustakaan dengan tiga atribut: judul, pengarang, dan isbn. Konstruktor kelas ini digunakan untuk menginisialisasi nilai-nilai dari atribut tersebut. Setiap atribut diakses melalui metode getter yaitu getJudul(), getPengarang(), dan getIsbn(), yang masing-masing mengembalikan nilai dari atribut judul, pengarang, dan isbn. Program ini hanya menyediakan getter untuk mengakses data, tanpa

adanya metode setter atau manipulasi lainnya, menjadikannya sebagai representasi objek buku yang bersifat read-only.

4. Tambahkan tampilan program untuk input data dengan cara klik kanan pada package perpustakaan → New → JFrame Form... , beri nama InputBuku
5. Buat tampilan seperti di bawah ini:

No.	Component	Variable Name
1	JPanel	
2	JScrollPane	
3	TextField	txtJudul
4	TextField	txtPengarang
5	TextField	txtIsbn
6	Button	btnSubmit
7	Table	tblBuku
8	JLabel	lblKeterangan

6. Klik 2x pada tombol Tambah Data, tambahkan kode berikut

```
private void BtnSubmitActionPerformed(java.awt.event.ActionEvent evt) {  
    String judul = txtJudul.getText();  
    String pengarang = txtPengarang.getText();  
    String isbn = txtIsbn.getText();  
  
    // Membuat objek Buku baru  
    Buku buku = new Buku(judul, pengarang, isbn);  
  
    // Menambah data buku ke tabel  
    DefaultTableModel data = (DefaultTableModel)  
tblBuku.getModel();  
data.addRow(new Object[]{buku.getJudul(), buku.getPengarang(),  
buku.getIsbn()});  
}
```

```
// Reset form input
txtJudul.setText("");
txtPengarang.setText("");
txtIsbn.setText("");
}
```

Penjelasan : Program di atas merupakan implementasi dari aksi tombol "Submit" pada aplikasi berbasis GUI. Ketika tombol ditekan, program mengambil teks dari tiga field input (txtJudul, txtPengarang, dan txtIsbn) untuk mendapatkan data yang dimasukkan oleh pengguna. Kemudian, sebuah objek Buku baru dibuat menggunakan data tersebut. Setelah itu, objek Buku ditambahkan ke dalam tabel (tblBuku) dengan menambahkan baris baru pada model tabel DefaultTableModel, yang mencakup judul, pengarang, dan ISBN buku. Setelah data buku berhasil ditambahkan, form input direset sehingga pengguna dapat memasukkan data buku berikutnya dengan form yang kosong.

7. Jalankan project lalu coba tambahkan data, maka akan muncul data pada tabel. Namun ketika form kosong lalu klik tambah data, maka akan ada data kosong yang masuk pada tabel. Hal ini bisa diantisipasi dengan membuat user defined exception
8. Buat sebuah class baru dengan nama ValidasiInputException

ValidasiInputException.java

```
package Perpustakaan;

/**
 * @author Tegar Ferdian Firmansyah
 */
public class ValidasiInputException extends Exception {
    public ValidasiInputException(String message) {
        super(message);
    }
}
```

Penjelasan : Program di atas merupakan implementasi dari kelas ValidasiInputException, yang merupakan turunan dari kelas Exception. Kelas ini digunakan untuk membuat pengecualian khusus (custom exception) yang dapat dilemparkan (thrown) ketika terjadi kesalahan pada validasi input dalam aplikasi. Konstruktor ValidasiInputException menerima parameter message, yang akan diteruskan ke konstruktor kelas induk Exception untuk memberikan pesan kesalahan yang relevan. Pesan ini dapat digunakan untuk memberi tahu pengguna atau pengembang mengenai masalah yang terjadi selama proses validasi input.

9. Ubah kode pada Class buku

Buku.java

```
package Perpustakaan;

/**
 * @author Tegar Ferdian Firmansyah
 */
public class Buku {
    private String judul;
    private String pengarang;
    private String isbn;

    public Buku(String judul, String pengarang, String isbn) throws
ValidasiInputException {
        if (judul.isEmpty()) {
            throw new ValidasiInputException("Judul buku tidak boleh kosong!");
        }
        if (pengarang.isEmpty()) {
            throw new ValidasiInputException("Pengarang tidak boleh kosong!");
        }
        if (!isbn.matches("\\d{13}")) {
            throw new ValidasiInputException("ISBN harus 13 digit angka!");
        }

        this.judul = judul;
        this.pengarang = pengarang;
        this.isbn = isbn;
    }
}
```

```

public String getJudul() {
    return judul;
}

public String getPengarang() {
    return pengarang;
}

public String getIsbn() {
    return isbn;
}
}

```

Penjelasan : Program di atas menambahkan validasi input pada konstruktor kelas Buku. Sebelum menginisialisasi atribut judul, pengarang, dan isbn, konstruktor memeriksa apakah judul dan pengarang kosong, atau jika isbn tidak memenuhi format yang tepat, yaitu terdiri dari 13 digit angka. Jika salah satu kondisi ini tidak terpenuhi, sebuah pengecualian `ValidasiInputException` dilemparkan dengan pesan yang sesuai, seperti "Judul buku tidak boleh kosong!", "Pengarang tidak boleh kosong!", atau "ISBN harus 13 digit angka!". Jika semua input valid, maka atribut objek Buku akan diinisialisasi dengan nilai yang diberikan. Validasi ini bertujuan untuk memastikan bahwa data buku yang dimasukkan sesuai dengan format yang diharapkan sebelum diproses lebih lanjut.

10. Klik 2x pada tombol Tambah Data, ubah kode berikut

```

private void BtnSubmitActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        String judul = txtJudul.getText();
        String pengarang = txtPengarang.getText();
        String isbn = txtIsbn.getText();

        // Membuat objek Buku baru
        Buku buku = new Buku(judul, pengarang, isbn);

        // Menambah data buku ke tabel
    }
}

```

```

DefaultTableModel data = (DefaultTableModel)
tblBuku.getModel();
data.addRow(new Object[]{buku.getJudul(),
buku.getPengarang(), buku.getIsbn()});

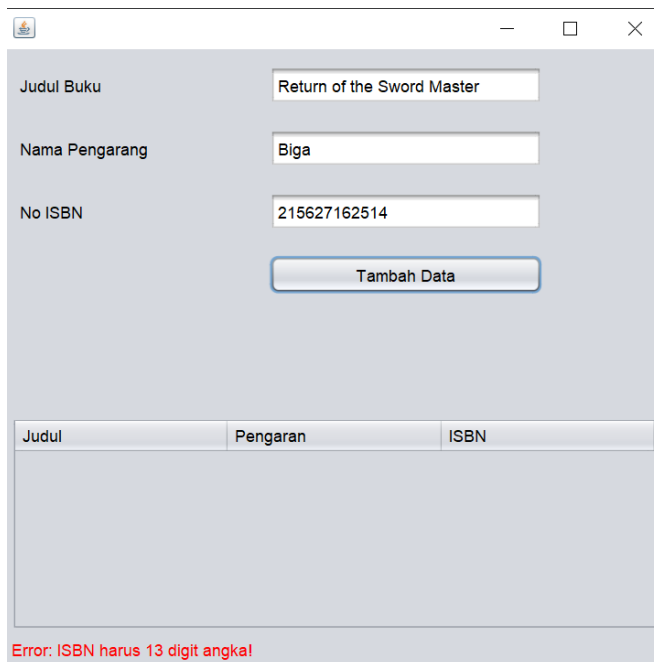
// Reset form input
txtJudul.setText("");
txtPengarang.setText("");
txtIsbn.setText("");
lblKeterangan.setForeground(Color.GREEN);
lblKeterangan.setText("Buku berhasil ditambahkan!");

} catch (ValidasiInputException e) {
    lblKeterangan.setForeground(Color.RED);
    lblKeterangan.setText("Error: " + e.getMessage());
}
}

```

Penjelasan : Program di atas menambahkan penanganan pengecualian (exception handling) untuk menangani kesalahan yang mungkin terjadi saat membuat objek Buku. Ketika tombol "Submit" ditekan, program mencoba untuk membuat objek Buku dengan data input dari form. Jika terjadi kesalahan validasi (misalnya, judul atau pengarang kosong, atau ISBN tidak valid), maka pengecualian `ValidasiInputException` akan ditangkap oleh blok `catch`. Pada kasus ini, pesan kesalahan akan ditampilkan pada label `lblKeterangan` dengan warna merah, memberikan umpan balik kepada pengguna. Jika tidak ada kesalahan, objek Buku berhasil ditambahkan ke tabel dan form input direset, serta label `lblKeterangan` menampilkan pesan sukses dengan warna hijau, yaitu "Buku berhasil ditambahkan!".

11. Jalankan project, coba tambahkan data baru. Ketika form kosong maka akan muncul peringatan pada bagian bawah aplikasi



Judul Buku: Return of the Sword Master

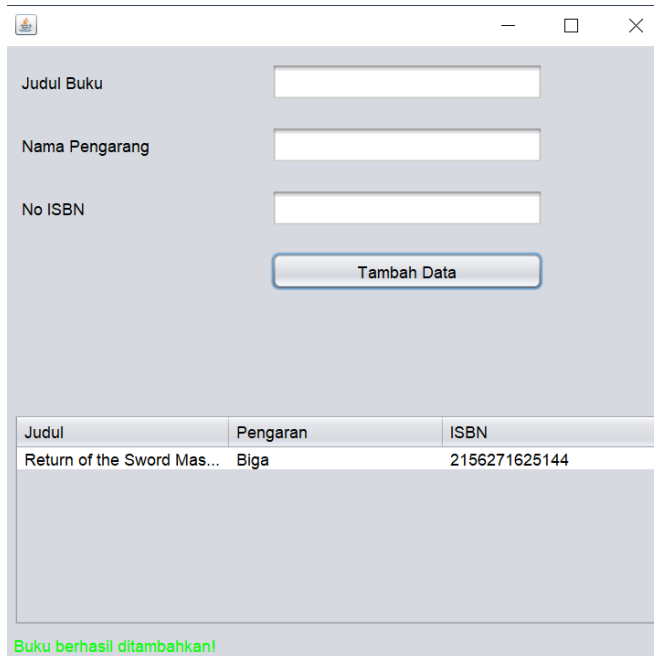
Nama Pengarang: Biga

No ISBN: 215627162514

Tambah Data

Judul	Pengaran	ISBN
-------	----------	------

Error: ISBN harus 13 digit angka!



Judul Buku:

Nama Pengarang:

No ISBN:

Tambah Data

Judul	Pengaran	ISBN
Return of the Sword Mas...	Biga	2156271625144

Buku berhasil ditambahkan!

C. UNGUIDED

Tambahkan inputan berikut pada aplikasi:

1. Tahun Terbit: Validasi untuk angka empat digit.

Buku.java

```
package Perpustakaan;

/**
 * @author Tegar Ferdian Firmansyah
 */

public class Buku {

    private String judul;

    private String pengarang;

    private String isbn;

    private String tahunTerbit;


    public Buku(String judul, String pengarang, String isbn, String tahunTerbit) throws
ValidasiInputException {

        if (judul.isEmpty()) {

            throw new ValidasiInputException("Judul buku tidak boleh kosong!");

        }

        if (pengarang.isEmpty()) {

            throw new ValidasiInputException("Pengarang tidak boleh kosong!");

        }

        if (!isbn.matches("\\d{13}")) {
```

```
        throw new ValidasiInputException("ISBN harus 13 digit angka!");
    }

    if (!tahunTerbit.matches("\\d{4}")) {
        throw new ValidasiInputException("Tahun Terbit harus 4 digit angka!");
    }

    this.judul = judul;

    this.pengarang = pengarang;

    this.isbn = isbn;

    this.tahunTerbit = tahunTerbit;

}

public String getJudul() {
    return judul;
}

public String getPengarang() {
    return pengarang;
}

public String getIsbn() {
    return isbn;
```

```

    }

    public String getTahunTerbit() {

        return tahunTerbit;

    }

}

```

Penjelasan : Program di atas menambahkan atribut baru yaitu tahunTerbit pada kelas Buku, yang digunakan untuk menyimpan informasi tentang tahun terbit buku. Selain itu, konstruktor kelas Buku juga diperbarui untuk menerima parameter tahunTerbit dan menambahkan validasi input untuk memastikan bahwa tahun terbit yang dimasukkan berupa angka dengan format 4 digit. Jika tahun terbit tidak sesuai format ini, pengecualian `ValidasiInputException` akan dilemparkan dengan pesan "Tahun Terbit harus 4 digit angka!". Setelah validasi berhasil, atribut tahunTerbit akan diinisialisasi bersama dengan atribut lainnya (judul, pengarang, dan isbn). Getter baru `getTahunTerbit()` juga ditambahkan untuk memungkinkan akses ke nilai tahun terbit. Validasi input ini bertujuan untuk memastikan bahwa data yang dimasukkan sesuai dengan format yang diharapkan.

InputBuku.java

```

private void btnSubmitActionPerformed(java.awt.event.ActionEvent evt) {

    try {

        String judul = txtJudul.getText();

        String pengarang = txtPengarang.getText();

        String isbn = txtIsbn.getText();

        String tahunTerbit = txtttahunTerbit.getText();

        // Membuat objek Buku baru

        Buku buku = new Buku(judul, pengarang, isbn, tahunTerbit, harga, kategori);

        // Menambah data buku ke tabel
    }
}

```

```

        DefaultTableModel data = (DefaultTableModel)
tblBuku.getModel();

        data.addRow(new Object[]{buku.getJudul(),buku.getPengarang(), buku.getIsbn(),
buku.getTahunTerbit()});

        // Reset form input
        txtJudul.setText("");
        txtPengarang.setText("");
        txtIsbn.setText("");
        txttahunTerbit.setText("");

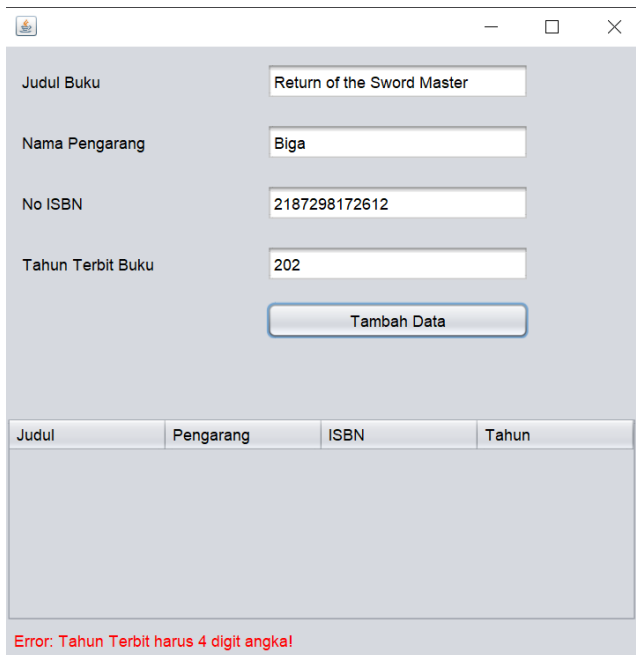
        lblKeterangan.setForeground(Color.GREEN);
        lblKeterangan.setText("Buku berhasil ditambahkan!");
    } catch (ValidasiInputException e) {
        lblKeterangan.setForeground(Color.RED);
        lblKeterangan.setText("Error: " + e.getMessage());
    }
}

```

Penjelasan : Program di atas adalah implementasi dari metode `btnSubmitActionPerformed` yang menangani proses penambahan data buku dengan validasi input. Ketika tombol "Submit" ditekan, program akan mencoba untuk mengambil data dari berbagai input, yaitu judul, pengarang, ISBN, tahun terbit, harga, dan kategori buku yang dipilih dari dropdown (`cbKategori`). Kemudian, objek Buku baru dibuat menggunakan data tersebut. Data buku ini kemudian ditambahkan ke dalam tabel `tblBuku` dalam format yang sesuai, yaitu judul, pengarang, ISBN, tahun terbit, harga, dan kategori. Setelah data ditambahkan, form input akan direset agar siap digunakan untuk memasukkan data baru. Label `lblKeterangan` akan menampilkan pesan keberhasilan dengan warna hijau. Namun, jika terjadi kesalahan validasi input (seperti menggunakan pengecualian `ValidasiInputException`), pesan kesalahan akan ditampilkan di label dengan warna merah. Dengan demikian, program ini memungkinkan

pengguna untuk menambahkan data buku secara dinamis dan memberikan umpan balik yang jelas jika terjadi kesalahan.

Output



Judul Buku: Return of the Sword Master

Nama Pengarang: Biga

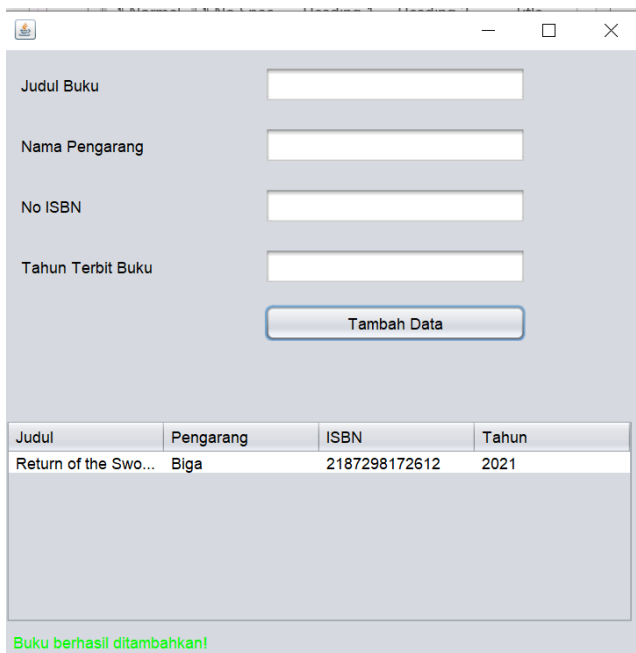
No ISBN: 2187298172612

Tahun Terbit Buku: 202

Tambah Data

Judul	Pengarang	ISBN	Tahun
-------	-----------	------	-------

Error: Tahun Terbit harus 4 digit angka!



Judul Buku:

Nama Pengarang:

No ISBN:

Tahun Terbit Buku:

Tambah Data

Judul	Pengarang	ISBN	Tahun
Return of the Swo...	Biga	2187298172612	2021

Buku berhasil ditambahkan!

2. Harga Buku: Validasi untuk angka positif.

Buku.java

```
package Perpustakaan;

/**
 * @author Tegar Ferdian Firmansyah
 */
public class Buku {

    private String judul;

    private String pengarang;

    private String isbn;

    private String tahunTerbit;

    private String harga;

    public Buku(String judul, String pengarang, String isbn, String tahunTerbit,
String harga) throws ValidasiInputException {
        if (judul.isEmpty()) {
            throw new ValidasiInputException("Judul buku tidak boleh kosong!");
        }

        if (pengarang.isEmpty()) {
            throw new ValidasiInputException("Pengarang tidak boleh kosong!");
        }

        if (!isbn.matches("\\d{13}")) {
            throw new ValidasiInputException("ISBN harus 13 digit angka!");
        }
    }
}
```

```
if (!tahunTerbit.matches("\\d{4}")) {  
    throw new ValidasiInputException("Tahun Terbit harus 4 digit angka!");  
}  
  
try {  
    long hargaNumber = Long.parseLong(harga);  
  
    if (hargaNumber < 0) {  
        throw new ValidasiInputException("Harga tidak boleh angka negatif!");  
    }  
} catch (NumberFormatException e) {  
    throw new ValidasiInputException("Input harus berupa angka!");  
}  
  
this.judul = judul;  
  
this.pengarang = pengarang;  
  
this.isbn = isbn;  
  
this.tahunTerbit = tahunTerbit;  
  
this.harga = harga;  
  
}  
  
public String getJudul() {  
    return judul;  
}
```



```
public String getPengarang() {  
    return pengarang;  
}  
  
public String getIsbn() {  
    return isbn;  
}  
  
public String getTahunTerbit() {  
    return tahunTerbit;  
}  
  
public String getHarga() {  
    return harga;  
}  
}
```

Penjelasan : Program di atas menambahkan atribut baru yaitu harga pada kelas Buku, yang digunakan untuk menyimpan informasi harga buku. Konstruktor kelas Buku diperbarui untuk menerima parameter harga dan menambahkan validasi input untuk memastikan bahwa harga yang dimasukkan adalah angka yang valid. Jika input harga tidak bisa diubah menjadi angka (menggunakan `Long.parseLong`), maka pengecualian `ValidasiInputException` akan dilemparkan dengan pesan "Input harus berupa angka!". Selain itu, jika harga yang dimasukkan bernilai negatif, pengecualian juga akan dilemparkan dengan pesan "Harga tidak

boleh angka negatif!". Setelah validasi berhasil, atribut harga akan diinisialisasi bersama dengan atribut lainnya. Getter baru `getHarga()` ditambahkan untuk memungkinkan akses ke nilai harga buku. Validasi ini bertujuan untuk memastikan bahwa harga buku yang dimasukkan sesuai dengan format yang diharapkan dan tidak mengandung nilai yang tidak valid.

InputBuku.java

```
private void BtnSubmitActionPerformed(java.awt.event.ActionEvent evt) {  
  
    try {  
  
        String judul = txtJudul.getText();  
  
        String pengarang = txtPengarang.getText();  
  
        String isbn = txtIsbn.getText();  
  
        String tahunTerbit = txtTahunTerbit.getText();  
  
        String harga = txtHarga.getText();  
  
  
        // Membuat objek Buku baru  
  
        Buku buku = new Buku(judul, pengarang, isbn, tahunTerbit, harga);  
  
  
        // Menambah data buku ke tabel  
  
        DefaultTableModel data = (DefaultTableModel)  
tblBuku.getModel();  
  
        data.addRow(new Object[]{buku.getJudul(),  
buku.getPengarang(), buku.getIsbn(),  
buku.getTahunTerbit(), buku.getHarga()});  
    }  
}
```

```

// Reset form input

txtJudul.setText("");

txtPengarang.setText("");

txtIsbn.setText("");

txtTahunTerbit.setText("");

txtHarga.setText("");

lblKeterangan.setForeground(Color.GREEN);

lblKeterangan.setText("Buku berhasil ditambahkan!");

} catch (ValidasiInputException e) {

    lblKeterangan.setForeground(Color.RED);

    lblKeterangan.setText("Error: " + e.getMessage());

}

}

```

Penjelasan : Program di atas menambahkan penanganan input untuk atribut tahunTerbit dan harga saat tombol "Submit" ditekan. Setelah mengambil teks dari field input yang baru ditambahkan (txttahunTerbit dan txtHarga), program mencoba untuk membuat objek Buku dengan data yang dimasukkan. Jika validasi input berhasil, data buku, termasuk judul, pengarang, ISBN, tahun terbit, dan harga, ditambahkan ke dalam tabel menggunakan model DefaultTableModel. Setelah itu, form input akan direset, dan label lblKeterangan menampilkan pesan sukses "Buku berhasil ditambahkan!" dengan warna hijau. Jika terjadi kesalahan dalam input (misalnya, validasi gagal), pesan error akan ditampilkan di label dengan warna merah, menunjukkan masalah yang terjadi, seperti input yang tidak valid untuk harga atau tahun terbit.

Output

Judul Buku

Return of the Sword Master

Nama Pengarang

Biga

No ISBN

1928761524521

Tahun Terbit Buku

2021

Harga Buku

-90000

Tambah Data

Judul	Pengarang	ISBN	Tahun
-------	-----------	------	-------

Error: Harga tidak boleh angka negatif!

Judul Buku

Nama Pengarang

No ISBN

Tahun Terbit Buku

Harga Buku

Tambah Data

Judul	Pengarang	ISBN	Tahun Terbit	Harga
Return of the ...	Biga	2182716782...	2021	900000

Buku berhasil ditambahkan!

3. Kategori Buku: Validasi untuk input tidak kosong. (dalam bentuk Dropdown)

Buku.java

```
package Perpustakaan;

/**
 * @author Tegar Ferdian Firmansyah
 */
public class Buku {

    private String judul;

    private String pengarang;

    private String isbn;

    private String tahunTerbit;

    private String harga;

    private String kategori;

    public Buku(String judul, String pengarang, String isbn, String tahunTerbit,
String harga, String kategori) throws ValidasiInputException {

        if (judul.isEmpty()) {

            throw new ValidasiInputException("Judul buku tidak boleh kosong!");

        }

        if (pengarang.isEmpty()) {

            throw new ValidasiInputException("Pengarang tidak boleh kosong!");

        }

    }

}
```

```
if (!isbn.matches("\\d{13}")) {  
    throw new ValidasiInputException("ISBN harus 13 digit angka!");  
}  
  
if (!tahunTerbit.matches("\\d{4}")) {  
    throw new ValidasiInputException("Tahun Terbit harus 4 digit angka!");  
}  
  
try {  
    long hargaNumber = Long.parseLong(harga);  
    if (hargaNumber < 0) {  
        throw new ValidasiInputException("Harga tidak boleh angka negatif!");  
    }  
} catch (NumberFormatException e) {  
    throw new ValidasiInputException("Input harus berupa angka!");  
}  
  
if (kategori == null || kategori.isEmpty()) {  
    throw new ValidasiInputException("Kategori harus dipilih!");  
}  
  
this.judul = judul;  
  
this.pengarang = pengarang;  
  
this.isbn = isbn;  
  
this.tahunTerbit = tahunTerbit;  
  
this.harga = harga;
```

```
this.kategori = kategori;
```

```
}
```

```
public String getJudul() {
```

```
    return judul;
```

```
}
```

```
public String getPengarang() {
```

```
    return pengarang;
```

```
}
```

```
public String getIsbn() {
```

```
    return isbn;
```

```
}
```

```
public String getTahunTerbit() {
```

```
    return tahunTerbit;
```

```
}
```

```
public String getHarga() {
```

```
    return harga;
```

```
}
```

```

public String getKategori() {

    return kategori;

}

}

```

Penjelasan : Program di atas menambahkan atribut baru yaitu kategori pada kelas Buku, yang digunakan untuk menyimpan informasi tentang kategori buku. Konstruktor kelas Buku diperbarui untuk menerima parameter kategori dan menambahkan validasi input untuk memastikan bahwa kategori tidak kosong atau null. Jika kategori tidak dipilih atau tidak diisi, pengecualian `ValidasiInputException` akan dilemparkan dengan pesan "Kategori harus dipilih!". Selain itu, validasi untuk atribut lainnya seperti judul, pengarang, isbn, tahunTerbit, dan harga tetap dipertahankan. Getter baru `getKategori()` juga ditambahkan untuk memungkinkan akses ke nilai kategori buku. Validasi ini memastikan bahwa semua data yang dimasukkan memenuhi format yang diharapkan dan sesuai dengan aturan yang ditetapkan.

InputBuku.java

```

private void BtnSubmitActionPerformed(java.awt.event.ActionEvent evt) {

    try {

        String judul = txtJudul.getText();

        String pengarang = txtPengarang.getText();

        String isbn = txtIsbn.getText();

        String tahunTerbit = txtTahunTerbit.getText();

        String harga = txtHarga.getText();

        String kategori = (String) cbxKategori.getSelectedItem();

        // Membuat objek Buku baru
    }
}

```



```
Buku buku = new Buku(judul, pengarang, isbn, tahunTerbit, harga, kategori);
```

```
// Menambah data buku ke tabel
```

```
DefaultTableModel data = (DefaultTableModel)
```

```
tblBuku.getModel();
```

```
data.addRow(new Object[]{buku.getJudul(),
```

```
buku.getPengarang(), buku.getIsbn(),
```

```
buku.getTahunTerbit(), buku.getHarga(),
```

```
buku.getKategori()});
```

```
// Reset form input
```

```
txtJudul.setText("");
```

```
txtPengarang.setText("");
```

```
txtIsbn.setText("");
```

```
txtTahunTerbit.setText("");
```

```
txtHarga.setText("");
```

```
cbxKategori.setSelectedIndex(0);
```

```
lblKeterangan.setForeground(Color.GREEN);
```

```
lblKeterangan.setText("Buku berhasil ditambahkan!");
```

```
} catch (ValidasiInputException e) {
```

```
    lblKeterangan.setForeground(Color.RED);
```

```
    lblKeterangan.setText("Error: " + e.getMessage());
```

```

    }

}

```

Penjelasan : Pada program di atas, terdapat penambahan elemen input `cbxKategori`, yaitu sebuah combo box yang digunakan untuk memilih kategori buku. Nilai kategori yang dipilih dari combo box kemudian disertakan dalam pembuatan objek Buku. Saat data buku ditambahkan ke tabel (`tblBuku`), kolom kategori juga ditambahkan pada baris yang baru. Setelah data berhasil ditambahkan, form input direset, termasuk mengembalikan pilihan combo box `cbxKategori` ke indeks pertama, dan label `lblKeterangan` menampilkan pesan keberhasilan dengan warna hijau ("Buku berhasil ditambahkan!"). Jika terjadi kesalahan validasi, exception akan ditangani, dan pesan kesalahan ditampilkan dengan warna merah di `lblKeterangan`. Dengan penambahan ini, aplikasi kini dapat menangani kategori buku dan menampilkannya pada tabel.


Output

The screenshot shows a Java Swing application window titled "Buku". Inside the window, there is a form with the following fields and values:

- Judul Buku: Return of the Sword Master
- Nama Pengarang: Biga
- No ISBN: 2187219281721
- Tahun Terbit Buku: 2021
- Harga Buku: 9000000
- Kategori Buku: (empty dropdown menu)

Below the form is a button labeled "Tambah Data". Underneath the button is a table with the following columns: Judul, Pengarang, ISBN, Tahun, Harga, and Kategori. The table is currently empty.

At the bottom of the window, there is a red error message: "Error: Kategori harus dipilih!".



—

□

×

Judul Buku

Nama Pengarang

No ISBN

Tahun Terbit Buku

Harga Buku

Kategori Buku

Tambah Data

Judul	Pengarang	ISBN	Tahun	Harga	Kategori
Return of t...	Biga	21872192...	2021	9000000	Novel

Buku berhasil ditambahkan!