LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
MODUL 5
LATIHAN EXCEPTION

Oleh :

Raditya Putra Anugra Pratama

2311103143

S1SI-07-C

PROGRAM STUDI SISTEM INFORMASI

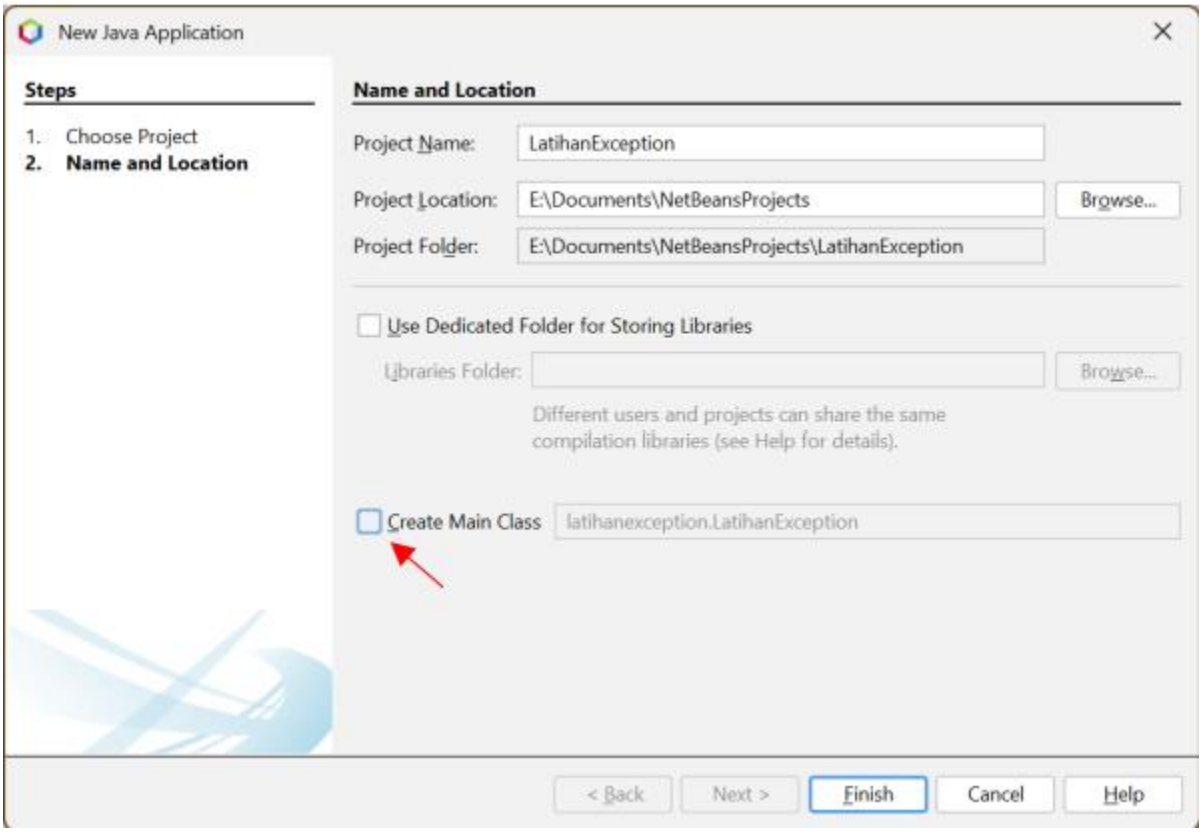
FAKULTAS REKAYASA INDUSTRI

UNIVERSITAS TELKOM PURWOKERTO

2024

A. 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

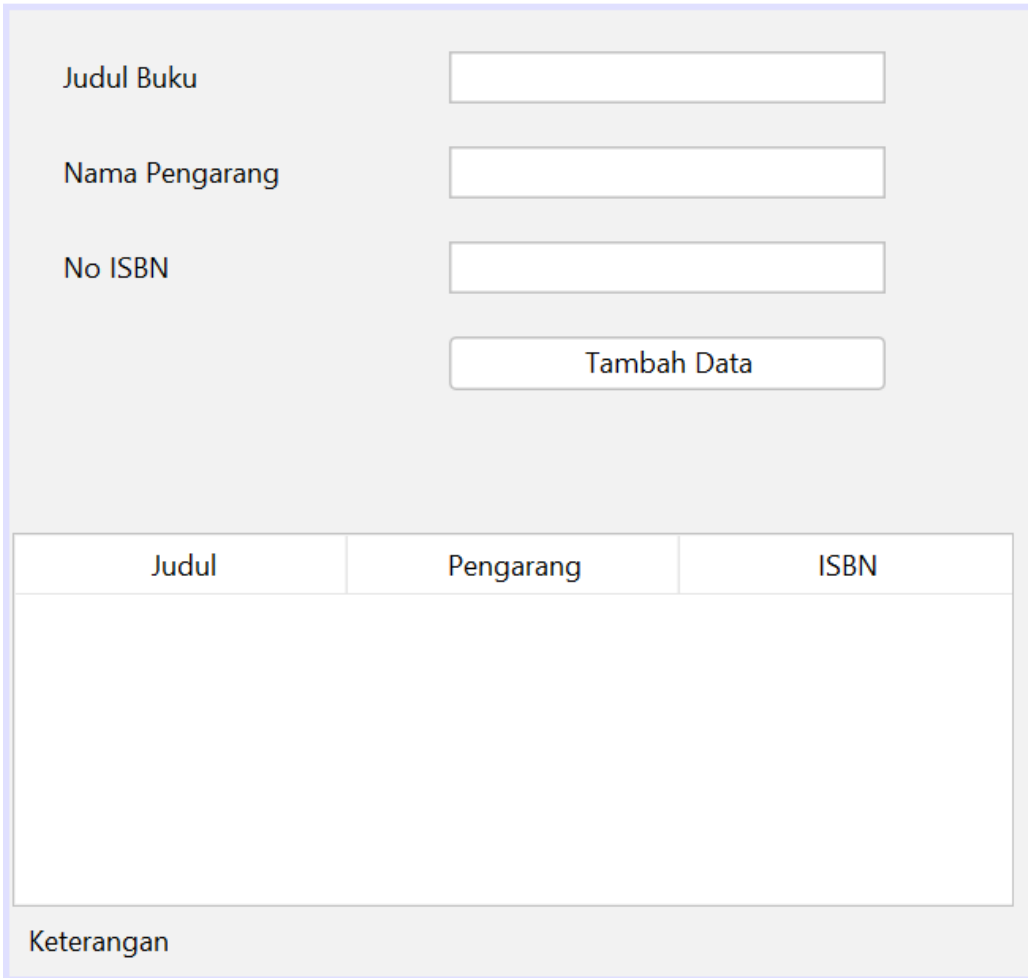
Buku.java

```
package Perpustakaan;


/**
 *
 * @author Raditya Putra Anugra Pratama
 */
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 : Kode di atas adalah kelas Buku dalam paket Perpustakaan yang menggambarkan sebuah entitas buku dalam sistem perpustakaan. Kelas ini memiliki tiga atribut privat, yaitu judul, pengarang, dan isbn, yang masing-masing menyimpan informasi tentang judul buku, pengarang buku, dan nomor ISBN (International Standard Book Number) buku tersebut. Konstruktor kelas ini digunakan untuk menginisialisasi ketiga atribut tersebut saat objek Buku dibuat. Selain itu, terdapat tiga metode getter (getJudul(), getPengarang(), dan getIsbn()) yang berfungsi untuk mengakses nilai dari masing-masing atribut buku secara publik. Dengan demikian, kelas ini memberikan struktur untuk menyimpan dan mengambil informasi buku dalam sebuah perpustakaan.

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:



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 : Kode di atas adalah sebuah metode event handler untuk tombol btnSubmit pada aplikasi berbasis GUI (Graphical User Interface), yang akan dijalankan ketika tombol tersebut diklik. Pertama, kode ini mengambil nilai teks dari tiga input form (textfield) yang digunakan untuk memasukkan judul buku (txtJudul), pengarang (txtPengarang), dan ISBN (txtIsbn). Setelah itu, sebuah objek Buku baru dibuat menggunakan data yang dimasukkan pada form tersebut. Kemudian, objek Buku tersebut ditambahkan ke dalam tabel (tblBuku) dengan menambahkan baris baru pada model tabel (DefaultTableModel), yang menampilkan informasi judul, pengarang, dan ISBN buku. Setelah data berhasil ditambahkan ke dalam tabel, input form akan di-reset (dikosongkan) untuk memungkinkan pengguna memasukkan data buku berikutnya. Secara keseluruhan, kode ini mengelola proses memasukkan data buku ke dalam sistem dan menampilkan hasilnya pada tabel.

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

   ValidInputException.java

```
package Perpustakaan;


/**
 *
 * @author Raditya Putra Anugra Pratama
 */
public class ValidasiInputException extends Exception{
    public ValidasiInputException(String message) {
        super(message);
    }
}
```

Penjelasan : Kode di atas mendefinisikan kelas ValidasiInputException yang merupakan turunan (subclass) dari kelas Exception. Kelas ini digunakan untuk membuat pengecualian (exception) kustom yang dapat digunakan dalam aplikasi untuk menangani kesalahan atau masalah yang berkaitan dengan validasi input pengguna. Konstruktor ValidasiInputException menerima sebuah parameter berupa pesan (string) yang menjelaskan kesalahan atau alasan mengapa pengecualian ini terjadi, dan kemudian meneruskan pesan tersebut ke konstruktor superclass Exception menggunakan super(message). Dengan demikian, kelas ini memungkinkan aplikasi untuk memberikan pesan kesalahan yang lebih spesifik terkait dengan validasi input yang tidak sesuai, serta memberikan cara untuk menangani kesalahan tersebut dengan lebih terstruktur.

9. Ubah kode pada Class buku

Buku.java

package Perpustakaan;

```
/**
 *
 * @author Raditya Putra Anugra Pratama
 */
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 : Pada kode yang ditambahkan, konstruktor kelas Buku telah dimodifikasi untuk memasukkan validasi input. Sebelum menginisialisasi atribut judul, pengarang, dan isbn, dilakukan pengecekan terhadap nilai-nilai yang dimasukkan. Jika judul kosong, maka akan dilemparkan pengecualian (ValidasiInputException) dengan pesan "Judul buku tidak boleh kosong!". Begitu juga jika pengarang kosong, maka pengecualian dengan pesan "Pengarang tidak boleh kosong!" akan dilemparkan. Untuk ISBN, dilakukan pengecekan apakah ISBN yang dimasukkan terdiri dari tepat 13 digit angka menggunakan regex (\\d{13}), jika tidak, maka pengecualian dengan pesan "ISBN harus 13 digit angka!" akan dilemparkan. Jika semua validasi berhasil, atribut judul, pengarang, dan isbn akan diinisialisasi dengan nilai yang valid. Pengecekan ini memastikan bahwa data buku yang dimasukkan memenuhi syarat dan mencegah kesalahan input pada objek Buku.

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 : Pada kode yang ditambahkan, blok try-catch digunakan untuk menangani pengecualian yang mungkin terjadi saat pengguna memasukkan data buku. Di dalam blok try, data yang dimasukkan oleh pengguna pada form (judul, pengarang, dan ISBN) akan digunakan untuk membuat objek Buku baru. Jika data valid, objek buku akan ditambahkan ke tabel dan form input akan direset. Setelah itu, label lblKeterangan akan menampilkan pesan "Buku berhasil ditambahkan!" dengan warna hijau untuk memberi tahu pengguna bahwa data telah berhasil ditambahkan. Namun, jika terjadi pengecualian ValidasiInputException (misalnya jika ada input yang kosong atau ISBN tidak valid), maka blok catch akan menangkap pengecualian tersebut. Di sini, pesan kesalahan dari pengecualian (e.getMessage()) akan ditampilkan pada label lblKeterangan dengan warna merah, memberi tahu pengguna tentang kesalahan input yang terjadi. Dengan penanganan

ini, aplikasi dapat memberi umpan balik yang jelas dan informatif kepada pengguna mengenai keberhasilan atau kegagalan input data buku.

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

B. UNGUIDED

Tambahkan inputan berikut pada aplikasi:

1. Tahun Terbit: Validasi untuk angka empat digit.

Buku.java

```
package Perpustakaan;


/**
 *
 * @author Raditya Putra Anugra Pratama
 */
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 : Pada kode yang ditambahkan, terdapat atribut baru tahunTerbit pada kelas Buku, yang menyimpan informasi tentang tahun terbit buku. Konstruktor kelas Buku juga telah dimodifikasi untuk menerima parameter tahunTerbit dan menambahkan validasi input untuk memastikan data yang dimasukkan sesuai. Validasi pertama memeriksa apakah tahunTerbit terdiri dari tepat 4 digit angka menggunakan regex (\\d{4}). Jika validasi gagal, maka akan dilemparkan pengecualian ValidasiInputException dengan pesan "Tahun Terbit harus 4 digit angka!". Setelah validasi berhasil, atribut judul, pengarang, isbn, dan tahunTerbit diinisialisasi dengan nilai yang valid. Selain itu, ditambahkan metode getter getTahunTerbit() untuk mengakses nilai dari atribut tahunTerbit. Penambahan ini memastikan bahwa data yang dimasukkan untuk buku lebih lengkap dan tervalidasi dengan baik, termasuk informasi tahun terbitnya.

InputBuku.java

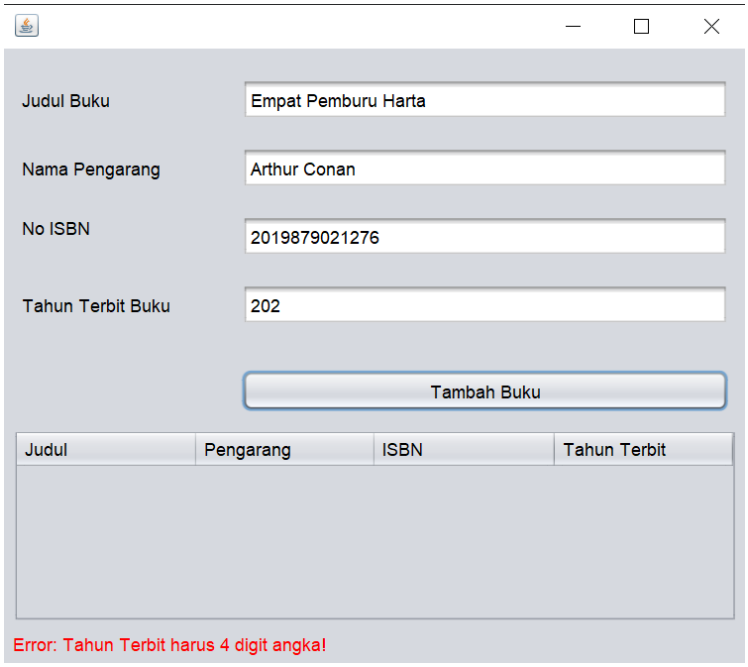
```
private void btnSubmitActionPerformed(java.awt.event.ActionEvent evt) {
        try {
```

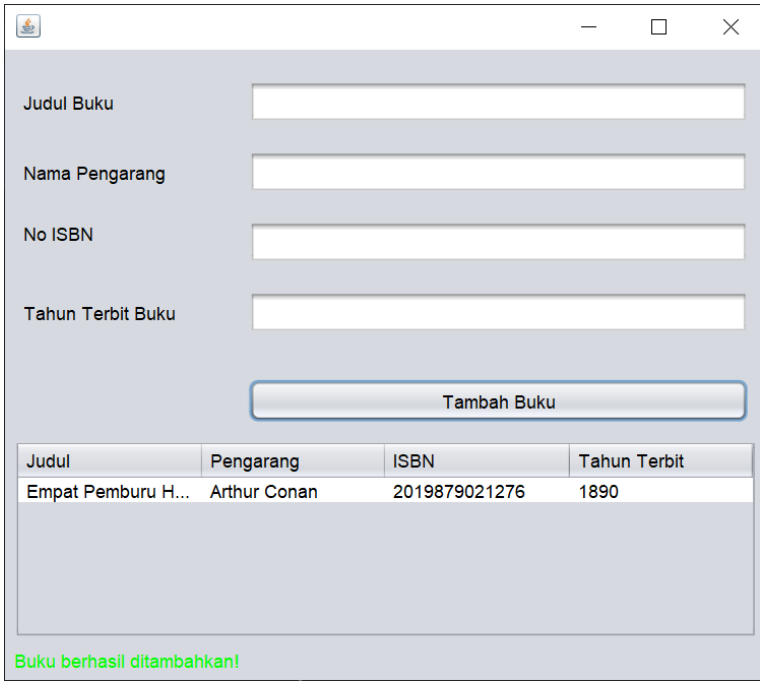
```
String judul = txtJudul.getText();
String pengarang = txtPengarang.getText();
String isbn = txtIsbn.getText();
String tahunTerbit = txtTahunTerbit.getText();

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

// 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 : Pada kode yang ditambahkan, input untuk tahunTerbit juga diambil dari form melalui txtTahunTerbit.getText() dan dimasukkan ke dalam objek Buku saat objek tersebut dibuat. Kemudian, data buku yang mencakup judul, pengarang, isbn, dan tahunTerbit ditambahkan ke dalam tabel menggunakan model tabel DefaultTableModel. Setiap kali data baru ditambahkan, kolom untuk tahun terbit juga ditampilkan pada tabel. Setelah itu, form input untuk judul, pengarang, isbn, dan tahunTerbit di-reset (dikosongkan), memungkinkan pengguna untuk memasukkan data baru. Pada bagian feedback, label lblKeterangan menampilkan pesan "Buku berhasil ditambahkan!" dengan warna hijau jika data berhasil ditambahkan, atau menampilkan pesan kesalahan jika ada validasi yang gagal, dengan warna merah. Kode ini memastikan data yang lengkap (termasuk tahun terbit) dapat ditambahkan dan memberikan umpan balik kepada pengguna terkait status input mereka.

Output





2. Harga Buku: Validasi untuk angka positif.

Buku.java

```
package Perpustakaan;


/**
 *
 * @author Raditya Putra Anugra Pratama
 */
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); // Ubah ISBN menjadi angka
        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 : Pada kode yang ditambahkan, terdapat atribut baru harga pada kelas Buku yang digunakan untuk menyimpan informasi harga buku. Selain itu, validasi input juga ditambahkan untuk memastikan bahwa nilai harga yang dimasukkan valid. Pertama, kode mencoba untuk mengonversi input harga menjadi angka dengan menggunakan Long.parseLong(harga). Jika konversi berhasil, dilakukan pengecekan apakah harga tersebut merupakan angka positif. Jika harga kurang dari 0, maka pengecualian ValidasiInputException akan dilemparkan dengan pesan "Harga tidak boleh angka negatif!". Jika input tidak dapat dikonversi menjadi angka (misalnya, jika pengguna memasukkan teks atau simbol), maka akan dilemparkan pengecualian dengan pesan "Input harus berupa angka!". Dengan penambahan ini, program tidak hanya memvalidasi input teks, tetapi juga memastikan bahwa harga buku yang dimasukkan berupa angka positif yang valid. Kemudian, atribut harga diinisialisasi dan dapat diakses melalui metode getter getHarga().

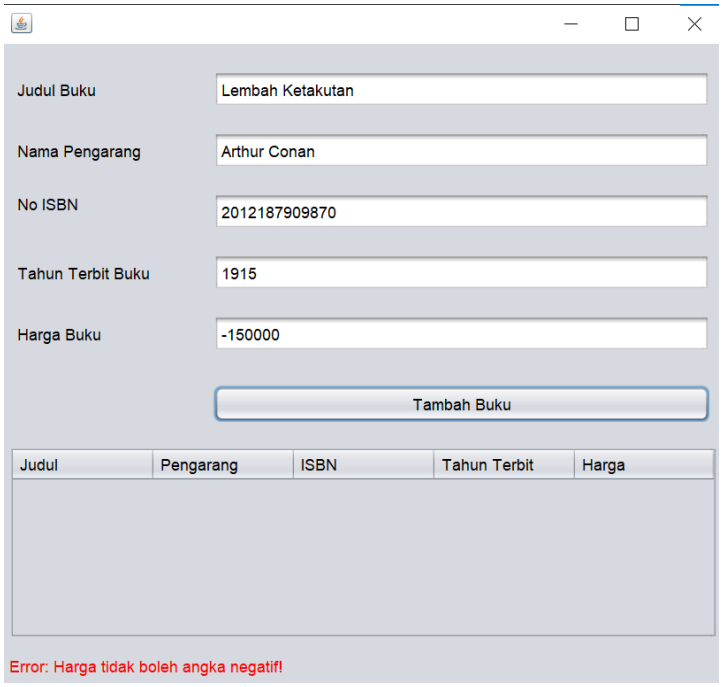
InputBuku.java

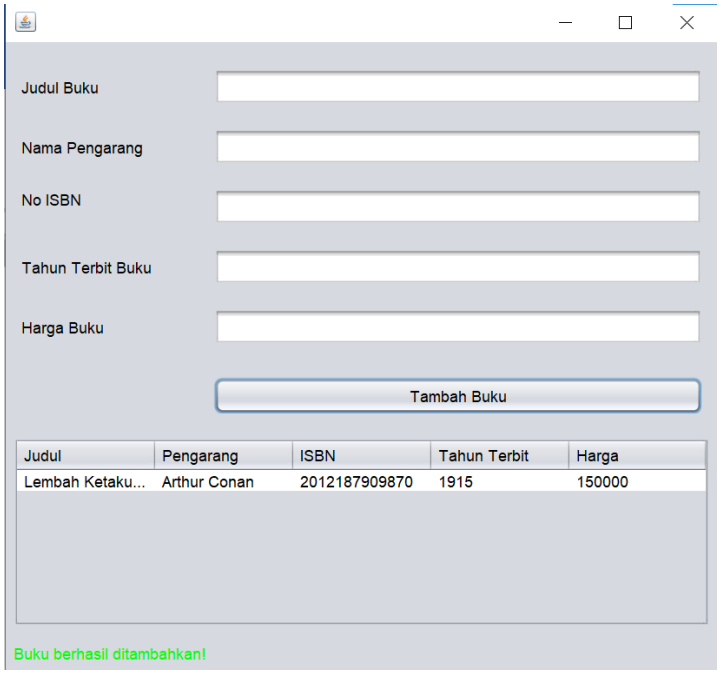
```
private void btnSubmitActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        String judul = txtJudul.getText();
        String pengarang = txtPengarang.getText();
        String isbn = txtIsbn.getText();
        String tahunTerbit = txtTahunTerbit.getText();
        String harga = txtHarga.getText();

        // 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 : Pada kode yang ditambahkan, input untuk harga juga diambil dari form melalui txtHarga.getText() dan dimasukkan ke dalam objek Buku saat objek tersebut dibuat. Selanjutnya, data buku yang mencakup judul, pengarang, isbn, tahunTerbit, dan harga ditambahkan ke dalam tabel dengan menambahkan baris baru pada model tabel DefaultTableModel. Setelah data berhasil ditambahkan, form input untuk semua kolom (judul, pengarang, isbn, tahun terbit, harga) direset, memungkinkan pengguna untuk memasukkan data baru. Pada bagian umpan balik, label lblKeterangan menampilkan pesan "Buku berhasil ditambahkan!" dengan warna hijau jika data berhasil ditambahkan. Jika terjadi kesalahan pada input, pengecualian ValidasiInputException akan ditangani dan menampilkan pesan kesalahan dengan warna merah pada label lblKeterangan, memberikan informasi yang jelas kepada pengguna mengenai kesalahan yang terjadi pada input, seperti kesalahan format harga atau data lainnya.

Output


Judul Buku
Lembah Ketakutan
Nama Pengarang
Arthur Conan
No ISBN
2012187909870
Tahun Terbit Buku
1915
Harga Buku
-150000
Tambah Buku
Judul
Pengarang
ISBN
Tahun Terbit
Harga
Error: Harga tidak boleh angka negatif!


Judul Buku
Nama Pengarang
No ISBN
Tahun Terbit Buku
Harga Buku
Tambah Buku
Judul
Pengarang
ISBN
Tahun Terbit
Harga
Lembah Ketaku...
Arthur Conan
2012187909870
1915
150000
Buku berhasil ditambahkan!

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

Buku.java

```
package Perpustakaan;


/**
 *
 * @author Raditya Putra Anugra Pratama
 */
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); // Ubah ISBN menjadi angka
    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 : Pada kode yang ditambahkan, terdapat atribut baru kategori yang digunakan untuk menyimpan kategori buku, seperti fiksi, non-fiksi, atau genre lainnya. Selain itu, validasi input juga ditambahkan untuk memastikan bahwa kategori yang dimasukkan tidak

kosong. Jika kategori kosong atau null, maka akan dilemparkan pengecualian ValidasiInputException dengan pesan "Kategori harus dipilih!". Dengan demikian, kode ini memastikan bahwa setiap objek Buku memiliki kategori yang valid selain atribut lainnya seperti judul, pengarang, ISBN, tahun terbit, dan harga. Konstruktor Buku menerima parameter baru kategori, yang diinisialisasi bersama atribut lainnya jika semua validasi berhasil. Sebagai tambahan, ditambahkan metode getter getKategori() untuk mengakses nilai dari atribut kategori. Penambahan ini memastikan bahwa data buku yang dimasukkan tidak hanya lengkap, tetapi juga memiliki kategori yang jelas dan tervalidasi.

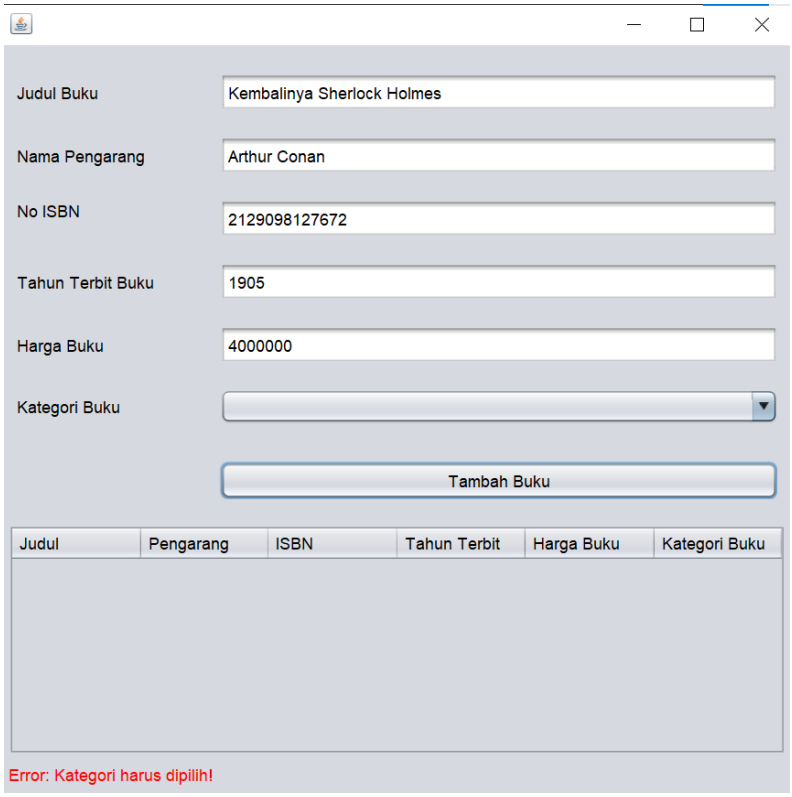
InputBuku.java

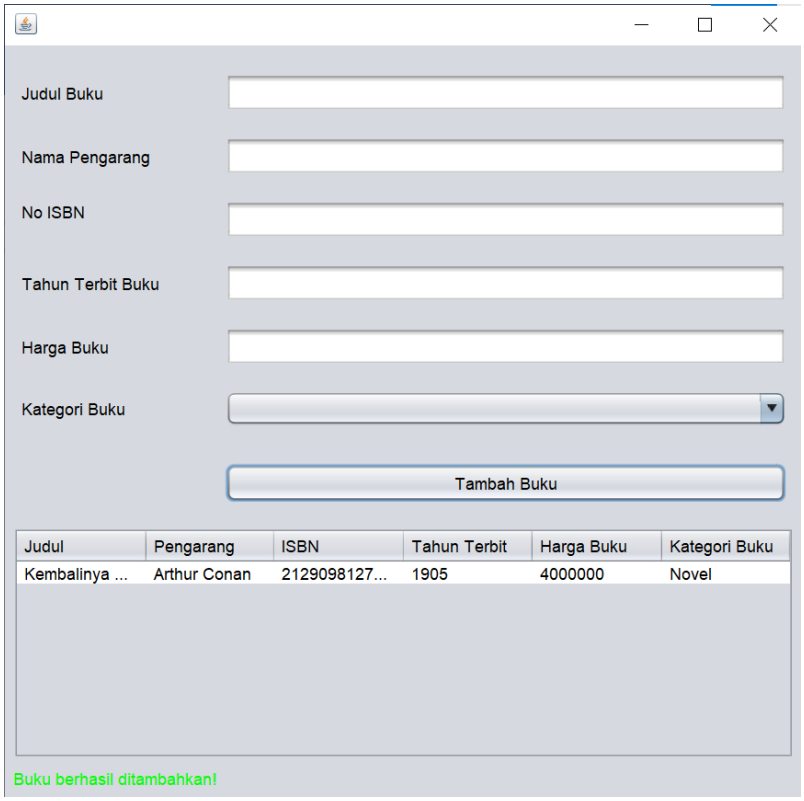
```
private void btnSubmitActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        String judul = txtJudul.getText();
        String pengarang = txtPengarang.getText();
        String isbn = txtIsbn.getText();
        String tahunTerbit = txtTahunTerbit.getText();
        String harga = txtHarga.getText();
        String kategori = (String) boxKategori.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("");
        boxKategori.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 kode yang ditambahkan, input untuk kategori diambil dari boxKategori, yang merupakan elemen dropdown (combo box). Nilai yang dipilih dari combo box (boxKategori.getSelectedItem()) akan dimasukkan ke dalam objek Buku saat objek tersebut dibuat. Kemudian, data buku yang mencakup judul, pengarang, isbn, tahunTerbit, harga, dan kategori ditambahkan ke dalam tabel menggunakan model tabel DefaultTableModel. Setelah data berhasil ditambahkan, form input akan di-reset, yaitu teks pada kolom txtJudul, txtPengarang, txtIsbn, txtTahunTerbit, dan txtHarga akan dikosongkan, dan pilihan pada boxKategori di-reset ke indeks pertama (kategori default). Umpan balik diberikan melalui label lblKeterangan, yang menampilkan pesan "Buku berhasil ditambahkan!" dengan warna hijau jika data berhasil ditambahkan, atau menampilkan pesan kesalahan jika ada masalah dalam validasi input dengan warna merah. Dengan penambahan ini, aplikasi dapat menangani input kategori buku dengan benar dan memberikan umpan balik yang jelas kepada pengguna.

Output


Judul Buku
Kembalinya Sherlock Holmes
Nama Pengarang
Arthur Conan
No ISBN
2129098127672
Tahun Terbit Buku
1905
Harga Buku
4000000
Kategori Buku
Tambah Buku
Judul
Pengarang
ISBN
Tahun Terbit
Harga Buku
Kategori Buku
Error: Kategori harus dipilih!


Judul Buku
Nama Pengarang
No ISBN
Tahun Terbit Buku
Harga Buku
Kategori Buku
Tambah Buku
Judul
Pengarang
ISBN
Tahun Terbit
Harga Buku
Kategori Buku
Kembalinya ...
Arthur Conan
2129098127...
1905
4000000
Novel
Buku berhasil ditambahkan!