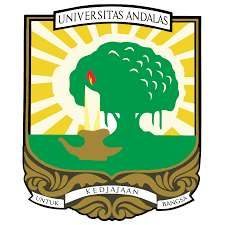
**LAPORAN PRAKTIKUM PBO**

**ABSTRACT CLASS, INTERFACE, FINAL CLASS, INNER CLASS**

****

**Disusun Oleh:**

**INDAH SYAHFITRI**

**2311532016**

**Dosen Pembimbing: Afdhal Dinilhak, S.Komp., M.Kom.**

**DEPARTEMEN INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI**

**UNIVERSITAS ANDALAS**

1. **PENDAHULUAN**

*Exception handling* adalah mekanisme yang digunakan dalam pemrograman untuk menangani kesalahan atau *exception* yang mungkin terjadi saat eksekusi program. Kesalahan ini bisa berupa berbagai macam hal, seperti kesalahan input, pembagian dengan nol, kegagalan dalam membaca file, atau kesalahan koneksi jaringan. Tanpa penanganan yang tepat, kesalahan ini bisa menyebabkan aplikasi berhenti secara tiba-tiba atau berperilaku tidak semestinya. Oleh karena itu, *exception handling* berperan penting untuk membuat aplikasi menjadi lebih stabil dan dapat mengatasi kesalahan dengan cara yang terstruktur dan terkontrol.

Di Java, error direpresentasikan dengan istilah exception dan semuanya direpresentasikan dalam bentuk class exception. Kita bisa membuat class exception sendiri atau menggunakan yang sudah disediakan oleh Java. Jika kita ingin membuat exception, maka kita harus membuat class yang extends class Throwable. Exception biasanya terjadi di method, sehingga ketika kita membuat exception di sebuah method maka method tersebut harus ditandai dengan kata kunci throws diikuti dengan class exceptionnya. Jika method tersebut bisa menimbulkan lebih dari satu jenis exception, kita bisa menambahkan lebih dari satu class exception. *Exception handling* umumnya dilakukan menggunakan blok *try-catch*, di mana kode yang berpotensi menimbulkan kesalahan diletakkan dalam blok *try*, dan jika terjadi kesalahan, blok *catch* akan menangkap kesalahan tersebut untuk ditangani.

Dengan menggunakan *exception handling*, pengembang dapat memastikan aplikasi memberikan respons yang tepat saat terjadi kesalahan, misalnya dengan menampilkan pesan kesalahan yang informatif atau mengambil langkah-langkah pemulihan untuk melanjutkan proses aplikasi. Selain itu, Java juga menyediakan berbagai jenis *exception* yang dapat digunakan untuk menangani kesalahan yang lebih spesifik, serta memungkinkan pengembang untuk membuat *exception* sendiri jika diperlukan. Konsep ini penting untuk meningkatkan kualitas dan keandalan perangkat lunak, karena aplikasi dapat menangani berbagai kondisi tak terduga tanpa menyebabkan kerusakan atau penghentian yang tidak diinginkan. Dengan demikian, *exception handling* tidak hanya berfungsi untuk menangani kesalahan, tetapi juga untuk menjaga kestabilan dan keberlanjutan aplikasi.

1. **TUJUAN PRAKTIKUM**
2. Mahasiswa mampu memahami dan mengimplementasikan konsep *exception* pada kasus login pengguna sebuah aplikasi.
3. **LANGKAH-LANGKAH**
4. Pada project laundryapps, pindahkan project ke branch baru. Biasanya, jika commit, maka akan tersimpan di master. Maka dari itu, kita bisa mengubah branch dengan klik kanan, pilih Team > Switch to > new branch. Buat branch baru untuk menyimpan project tertentu untuk seterusnya. Di sini, saya telah membuat branch baru dengan nama PBO2, jadi saya sudah menukar branch dengan branch baru.

Sebuah gambar berisi teks, cuplikan layar, software, tampilan

Deskripsi dibuat secara otomatis

Tampilan akan menjadi seperti berikut jika eclipse telah berhasil tersambung ke github dan branch telah berhasil diubah.

Sebuah gambar berisi teks, cuplikan layar, software

Deskripsi dibuat secara otomatis

1. Selanjunya, pada laundryapps telah ada package dengan nama model dan kelas bernama User. Inisialisasikan atribut yang akan digunakan pada kelas ini lalu pakai constructor seperti berikut.

Sebuah gambar berisi teks, komputer, cuplikan layar, software

Deskripsi dibuat secara otomatis

1. Buat package baru dengan nama error, kemudian buat class baru dengan nama ValidationException. Isi kelas dengan program seperti berikut.

Sebuah gambar berisi teks, cuplikan layar, Font, garis

Deskripsi dibuat secara otomatis

1. Buat package baru dengan nama util dan di dalamnya buat sebuah class baru dengan nama ValidationUtil. Class ValidationUtil memiliki static method dengan nama validate yang akan memvalidasi proses login yang jika terjadi error akan memberitahu pemanggilnya. Dalam kelas ini, dibuat percabangan if statement dengan tindakan exceptions, yakni jika terdapat salah satu atribut yang harus diisi namun dibiarkan kosong oleh pengguna, maka akan ada output berupa peringatan/tanda pada user bahwa atribut tertentu masih belum terisi sehingga tidak bisa lanjut ke tahap berikutnya.

Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Deskripsi dibuat secara otomatis

1. Buat package baru dengan nama service, kemudian di dalam package service ini buat class baru dengan nama LoginService. Class LoginService ini berfungsi untuk melakukan otentikasi username dan password yang dimasukkan pengguna dengan menggunakan try dan catch. Username dan password yang dimasukkan pengguna akan dicocokan dengan data yang ada di database, jika cocok maka login pengguna berhasil namun jika tidak cocok maka login pengguna gagal.

Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Deskripsi dibuat secara otomatis

1. Buat jFrame baru pada package ui dengan nama LoginFrame dengan desain tampilan login seperti berikut.

Sebuah gambar berisi teks, cuplikan layar, tampilan, software

Deskripsi dibuat secara otomatis

1. Klik kanan pada button login, pilih add event handler → action → actionPerformed. Selanjutnya, lakukan validasi terhadap inputan pengguna setelah tombol login diklik. Validasi dilakukan dengan memanggil method validate yang telah dibuat pada tahapan sebelumnya.  
   Sebuah gambar berisi teks, cuplikan layar, software

   Deskripsi dibuat secara otomatis

Saat memanggil method yang sebuah method yang kemungkinan bisa menyebabkan exception maka kita wajib menggunakan try-catch expression di Java. Ini bertujuan agar kita bisa menangkap exception yang terjadi, karena jika tidak ditangkap ketika terjadi exception maka program kita secara otomatis akan terhenti. Cara menggunakan try-catch expression di Java adalah di block try kita tinggal panggil method yang bisa menyebabkan exception dan di block catch kita bisa melakukan sesuatu jika terjadi exception.

1. Setelah berhasil login dan masuk ke MainFrame, buat sebuah jFrame baru dengan tampilan seperti di bawah ini.

Sebuah gambar berisi teks, cuplikan layar, tampilan, Font

Deskripsi dibuat secara otomatis

Pengguna bisa memasukan data angka-angka yang ketika tombol Simpan diklik data tersebut akan disimpan pada variabel dalam bentuk Array dan ditampilkan. Pengguna juga bisa melakukan pemeriksaan isi dari array yang telah disimpan tadi. Hasil yang diharapkan ketika pemeriksaan dilakukan adalah jika index array yang dimasukkan ada, maka akan ditampilkan nilai dari array tersebut. Namun, jika index array yang dimasukkan tidak ada maka aplikasi akan menampilkan pesan “Exception Caught”. Penjelasan kodenya yakni seperti berikut.

1. Klik tombol simpan, klik kanan lalu tekan add event handler > action dan buat kode berikut.

Sebuah gambar berisi teks, cuplikan layar, Software multimedia, software

Deskripsi dibuat secara otomatis

Program ini mengatur fungsi tombol "Simpan" untuk menyimpan data yang dimasukkan pengguna. Ketika tombol ditekan, teks dari inputField diambil dan dipecah menjadi array string menggunakan koma (,) sebagai pemisah. Setiap elemen array kemudian dikonversi menjadi bilangan bulat dan ditambahkan ke dalam ArrayList bernama data. Setelah itu, data yang tersimpan di ArrayList ditampilkan di label lblA menggunakan lblA.setText(data.toString()). Untuk memudahkan pengguna, kolom input otomatis dikosongkan dengan inputField.setText("") setelah data disimpan. Jika terjadi kesalahan, seperti input non-angka, program akan menampilkan pesan kesalahan melalui dialog menggunakan JOptionPane.showMessageDialog().

1. Klik tombol check, klik kanan lalu tekan add event handler > action dan buat kode berikut.

Sebuah gambar berisi teks, cuplikan layar, Font

Deskripsi dibuat secara otomatis

Program berikut adalah event listener untuk tombol checkButton, yang berfungsi memeriksa nilai pada indeks tertentu dalam daftar data yang telah disimpan. Ketika tombol diklik, program mencoba membaca angka yang dimasukkan pengguna pada indexField dan mengonversinya menjadi bilangan bulat menggunakan Integer.parseInt(). Nilai pada indeks tersebut kemudian diambil dari daftar data menggunakan data.get(index) dan ditampilkan pada label lblA\_1. Jika indeks yang dimasukkan berada di luar batas daftar, program akan menangkap kesalahan IndexOutOfBoundsException dan menampilkan pesan "Exception Caught: Index tidak valid." pada label. Jika input indeks tidak berupa angka, kesalahan NumberFormatException akan ditangkap, dan pesan "Input index harus berupa angka." ditampilkan. Pendekatan ini menggunakan mekanisme exception handling untuk memastikan program tetap berjalan meskipun terjadi kesalahan input.

1. **SIMPULAN**

Dalam praktikum ini, konsep *exception handling* telah berhasil diimplementasikan dalam sebuah aplikasi berbasis Java untuk pengelolaan data array. Melalui mekanisme penanganan kesalahan ini, aplikasi mampu menangani berbagai jenis input yang tidak valid, seperti indeks di luar batas array atau input yang bukan angka, tanpa menyebabkan program berhenti secara tiba-tiba. Penggunaan *try-catch block* memungkinkan aplikasi untuk memberikan respons yang sesuai kepada pengguna berupa pesan kesalahan yang jelas, sehingga meningkatkan keandalan dan kenyamanan penggunaan aplikasi.

Selain itu, praktikum ini juga mengajarkan cara memanfaatkan komponen GUI seperti JTextField, JButton, dan JLabel untuk membuat antarmuka pengguna yang interaktif. Fitur seperti menampilkan data array yang tersimpan secara dinamis dan melakukan pengecekan nilai pada indeks tertentu berhasil diterapkan menggunakan komponen-komponen tersebut. Dengan ini, logika program berinteraksi dengan elemen-elemen GUI untuk membangun aplikasi yang fungsional bisa dipahami dengan baik.