

Devi Aryaning Tyas (E31200973)

Laporan Praktikum ke – 5

2020

Politeknik Negeri Jember

Jurusan Teknologi Informasi

Program Studi Manejemen Informatika

Semester 1

Oleh :

Exception Handling

**BAB 1**

**DASAR TEORI**

* 1. **Pengertian Exception Handling**

Exception Handling adalah cara menangkap kesalahan saat program dijalankan pada java netbeans agar jalannya program tidak keluar begitu saja dari alur yang ditentukan. Beberapa kata kunci yang digunakan diantaranya adalah try, catch, finally, throw, dan throws. Urutan jalannya program diatur dalam method main dan exception handling akan membaca pengecualian sesuai dengan urutan tersebut. Untuk menangkap kesalahan saat program dijalankan, maka harus mengetahui bagian mana yang mungkin mengalami kesalahan dan memasukkannya dalam blok try sehingga saat kesalahan terjadi dapat ditangkap oleh blok catch.

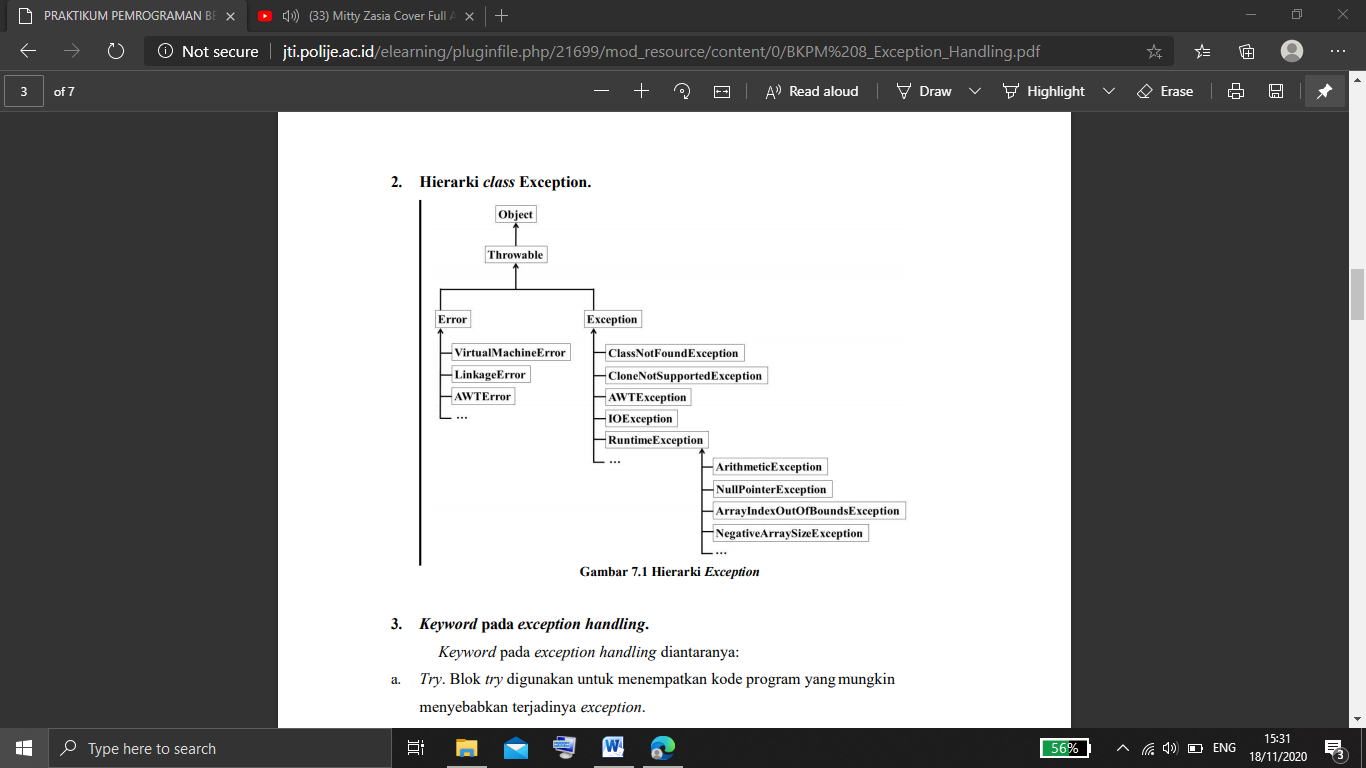
* 1. **Konsep Exception Handling**

Kategori error dalam pemrograman diantaranya:

a. Compilation error, yaitu error yang terjadi saat program dikompilasi.

b. Runtime error, yaitu error yang terjadi saat program dieksekusi/dijalankan. c. Logic error, yaitu error yang terjadi saat program tidak berjalan sesuai dengan harapan.

Exception (eksepsi atau pengecualian) adalah kondisi yang menyebabkan program menjadi hang (tergantung) atau quit (kaluar) dari alur normal yang telah ditentukan pada saat program dijalankan. Exception dipicu oleh runtime error, yaitu error atau kesalahan yang terjadi saat program dieksekusi oleh interpreter. Sedangkan exception handling (penanganan pengecualian) merupakan mekanisme untuk menangkap exception tersebut.

* 1.  **Hierarki class Exception.**
  2. **Keyword pada Exception Handling**

Keyword pada exception handling diantaranya:

a. Try. Blok try digunakan untuk menempatkan kode program yangmungkin menyebabkan terjadinya exception.

b. Catch. Blok catch digunakan untuk menangkap kesalahan yang terjadi pada blok try.

c. Finally. Blok finally akan selalu dijalankan, tidak peduli apakah terjadi exception atau tidak.

d. Throw. Klausa throw digunakan untuk melemparkan exception yang terjadi.

e. Throws. Klausa throws digunakan untuk mendeklarasikan method yang mungkin akan mengalami exception.

* 1. **Pengenalan class JList dan JComboBox.**

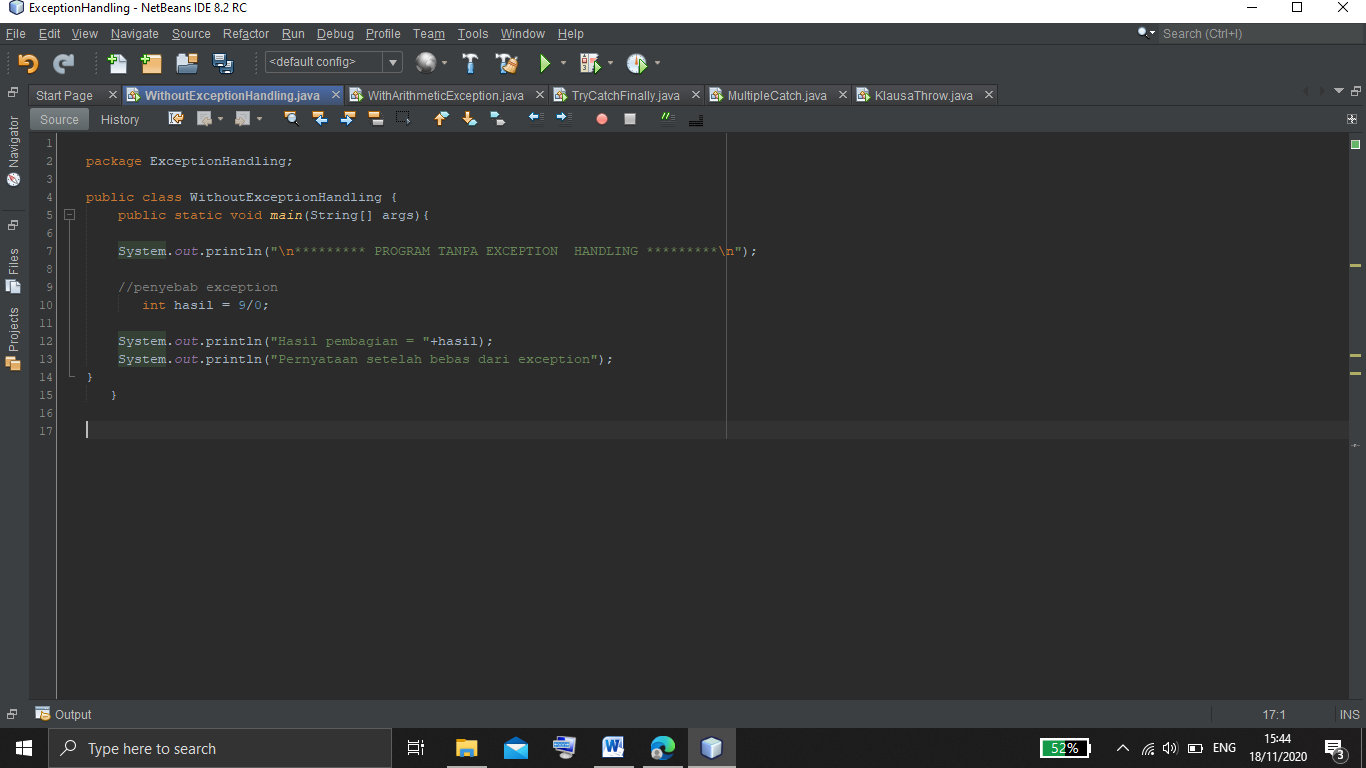
Class JList merupakan class dari package javax.swing yang digunakan untuk membuat komponen berupa list atau daftar pilihan. Dengan menggunakan list, seluruh alternatif pilihan yang tersedia dapat ditampilkan secara bersamaan. Class JComboBox merupakan class dari package javax.swing yang digunakan untuk membuat komponen berupa combo box atau kotak kombo yang berisi pilihan seperti pada list. Perbedaannya adalah dengan menggunakan combo box, seluruh alternatif pilihan yang tersedia ditampilkan seperti menu drop down. Oleh karena itu, combo box juga dikenal sebagai drop down list.

**BAB II**

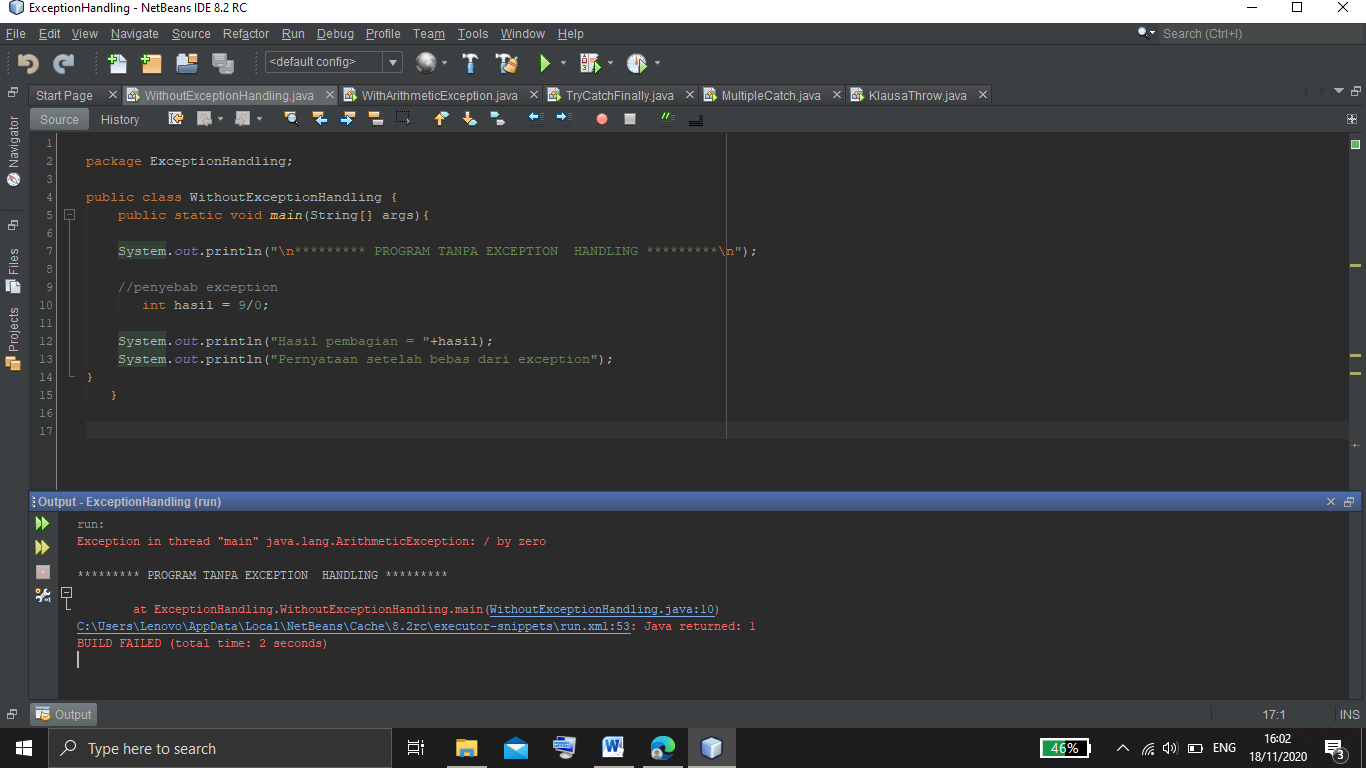
**HASIL UJI COBA PRAKTIKUM**

**2.1. Program tanpa exception handling.**

Kode Program :

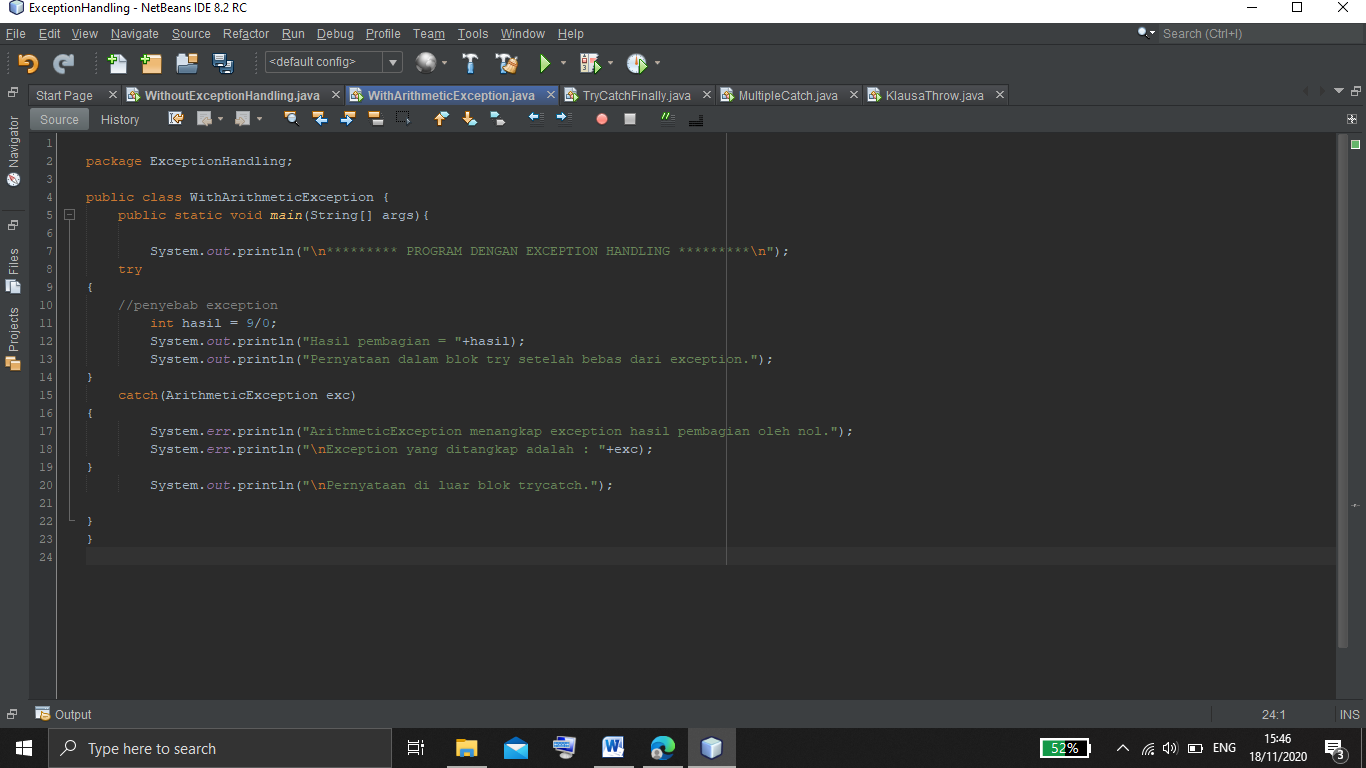


Output :

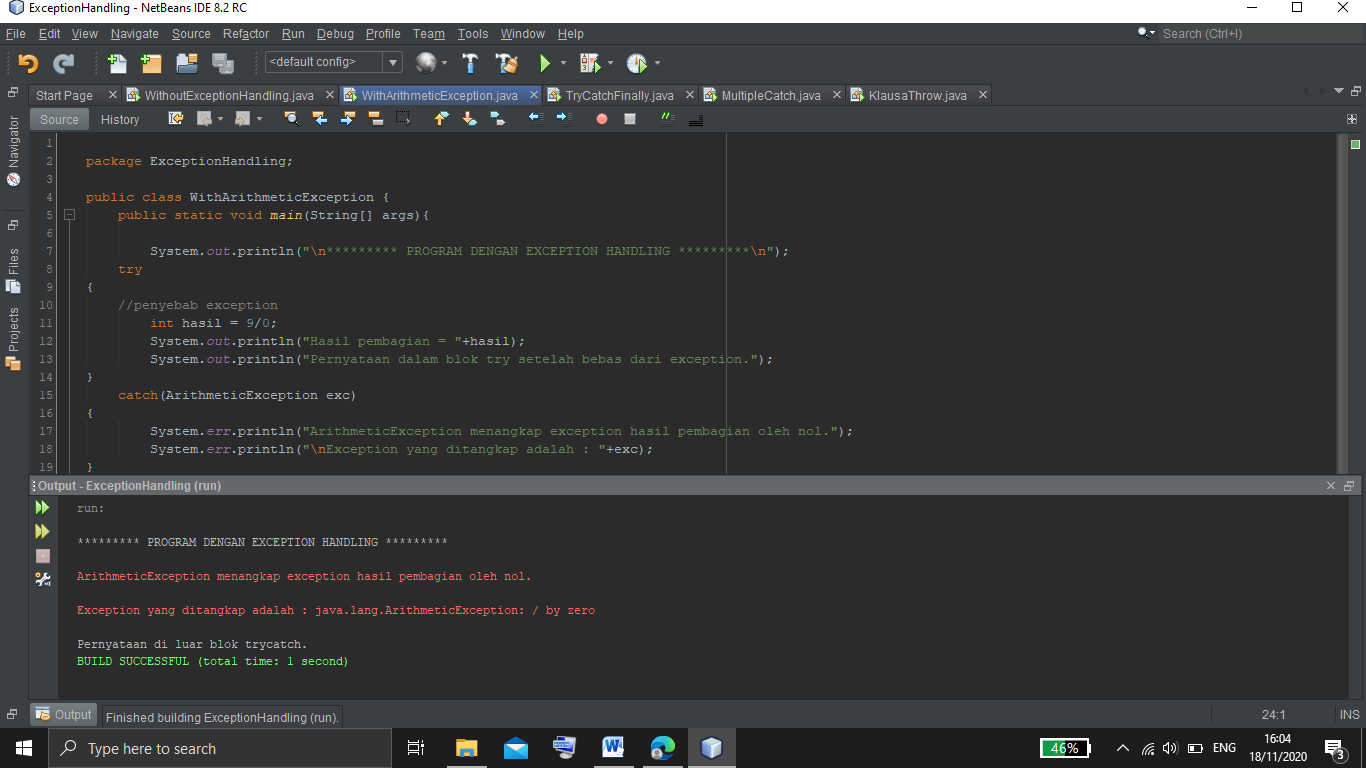


**2.2. Menangkap exception dengan blok try-catch.**

Kode Program :

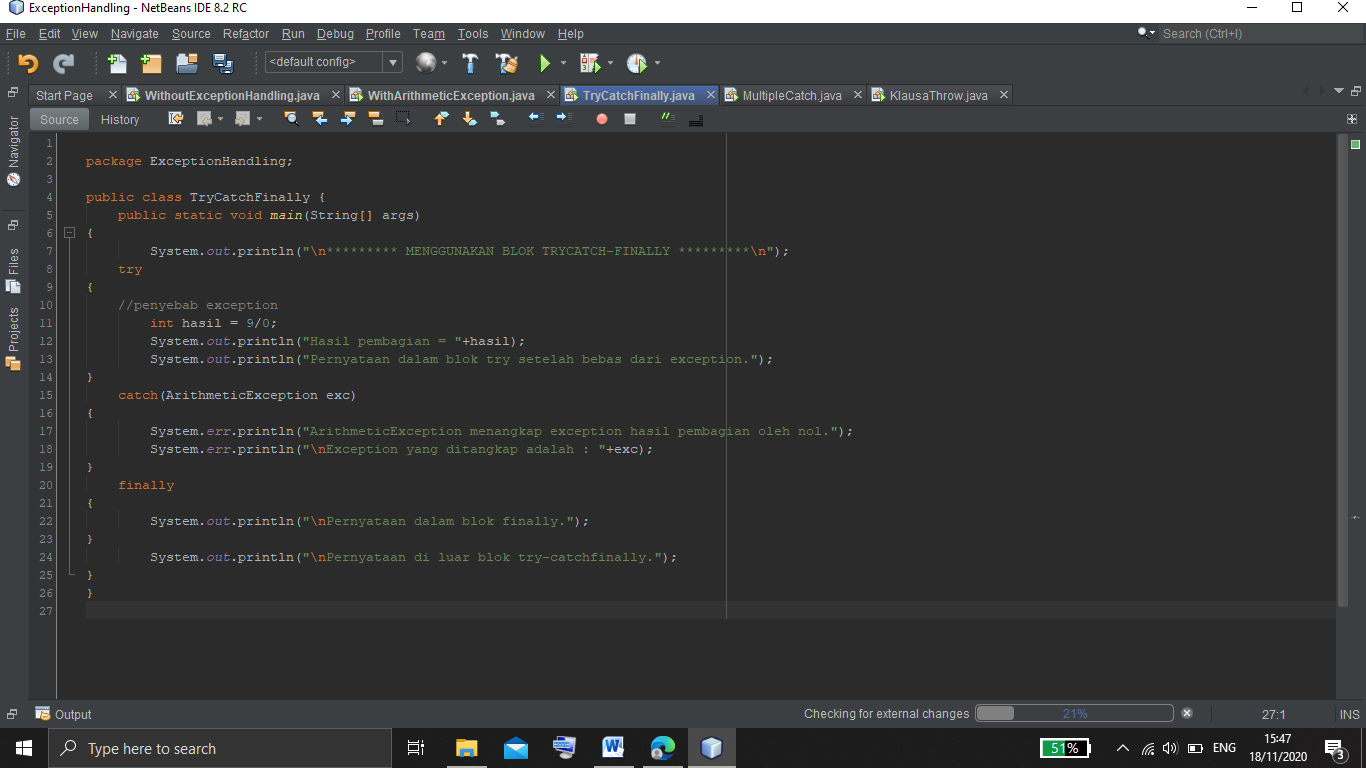


Output :

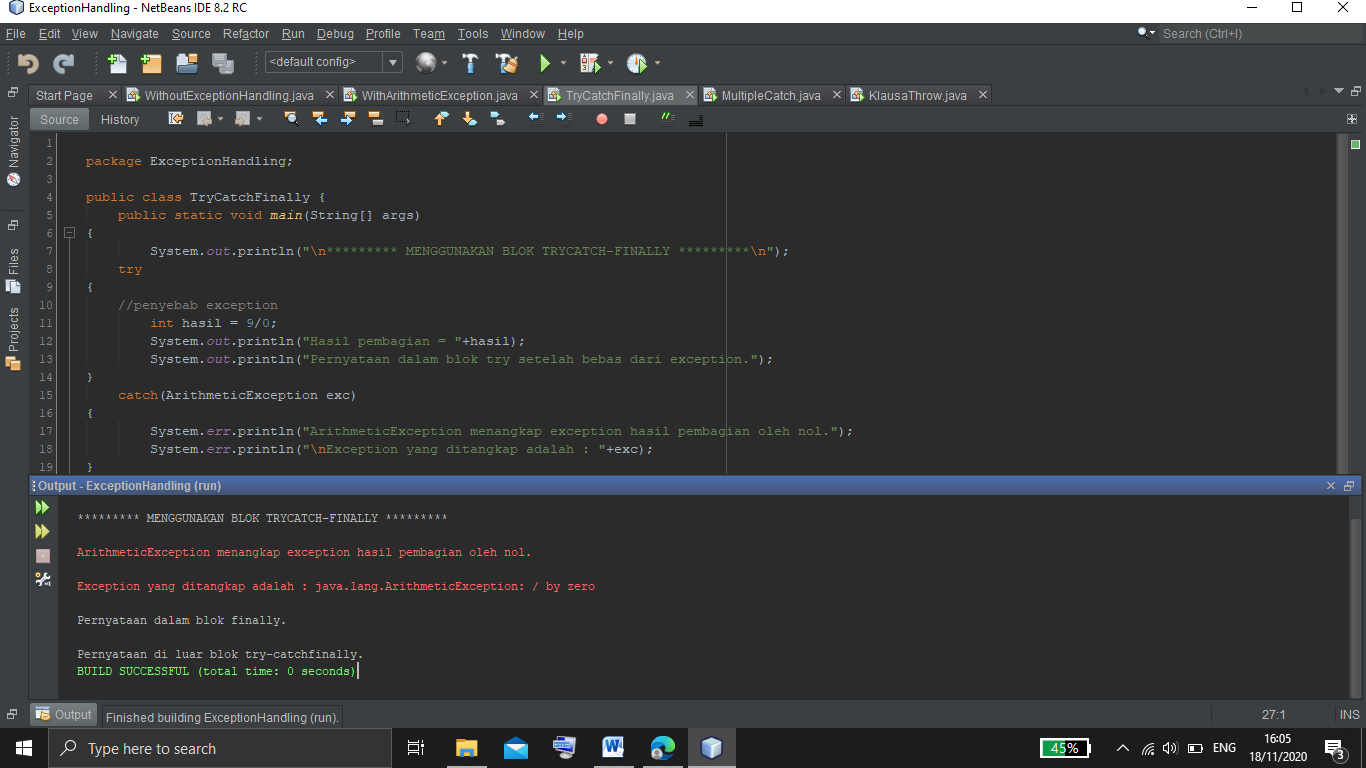


**2.3. Membuat blok try-catch-finally.**

Kode Program :

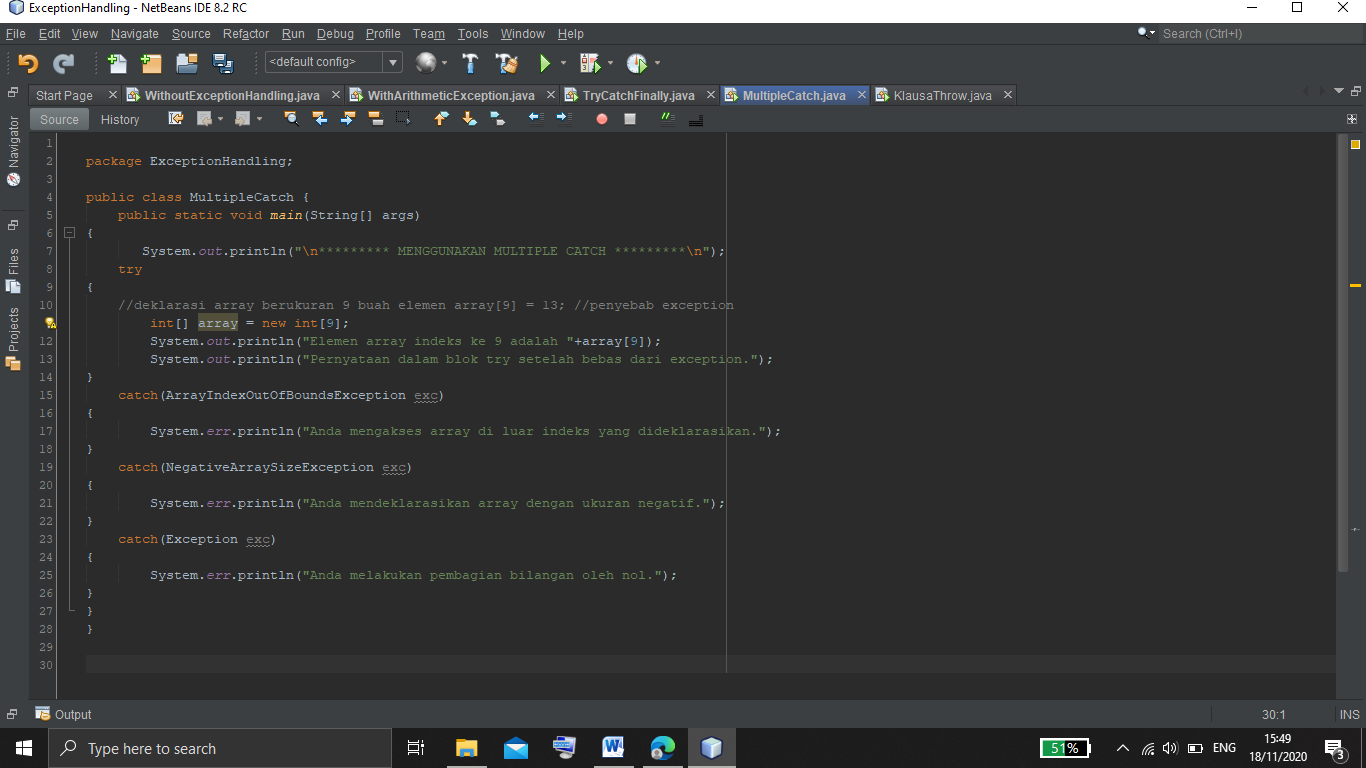


Output :

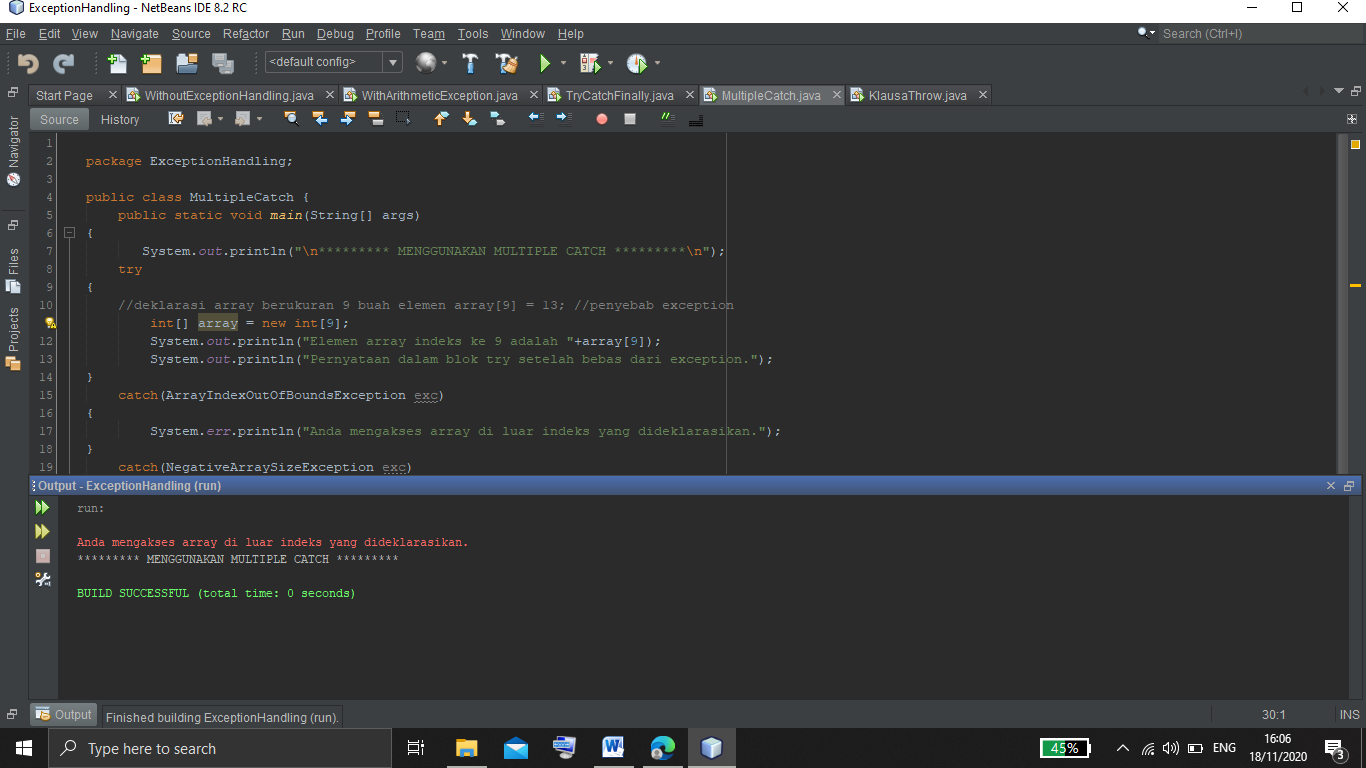


**2.4. Membuat catch secara bertingkat.**

Kode Program :

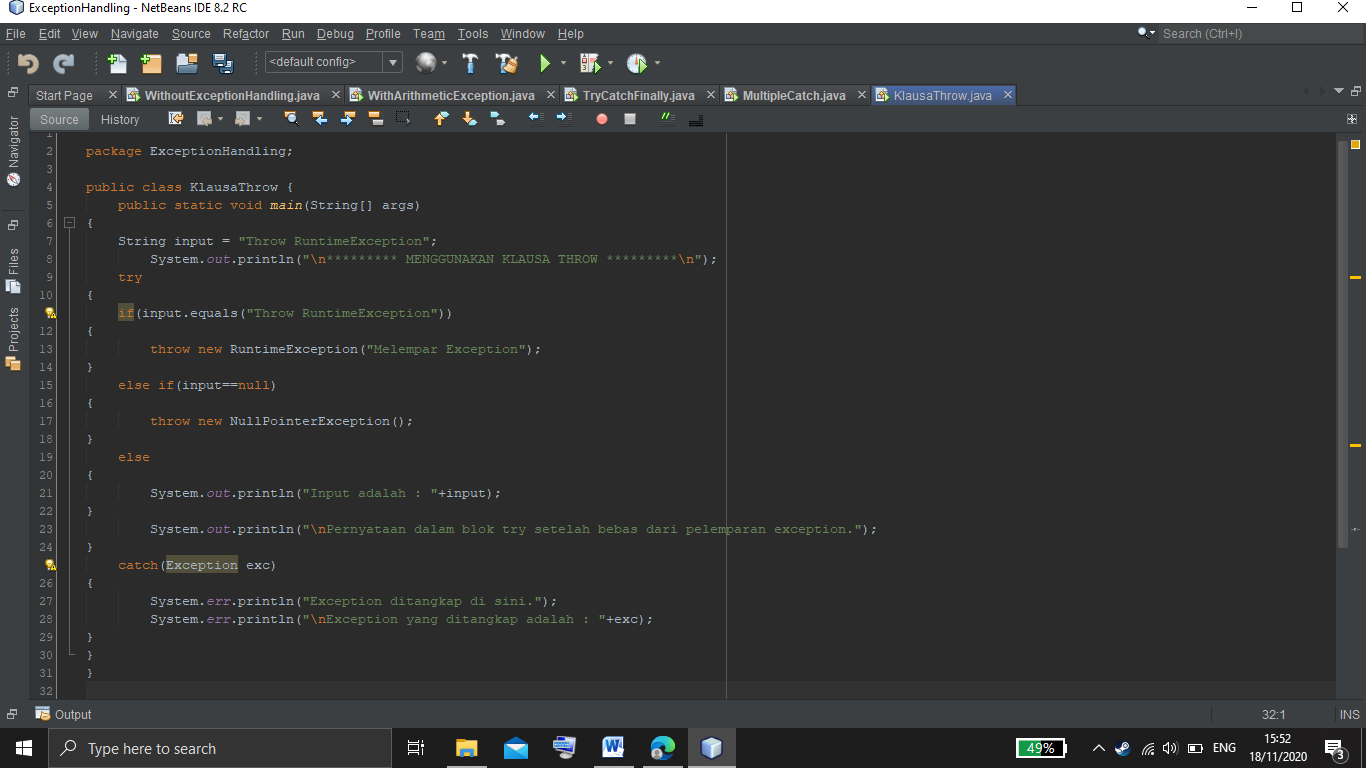


Output :

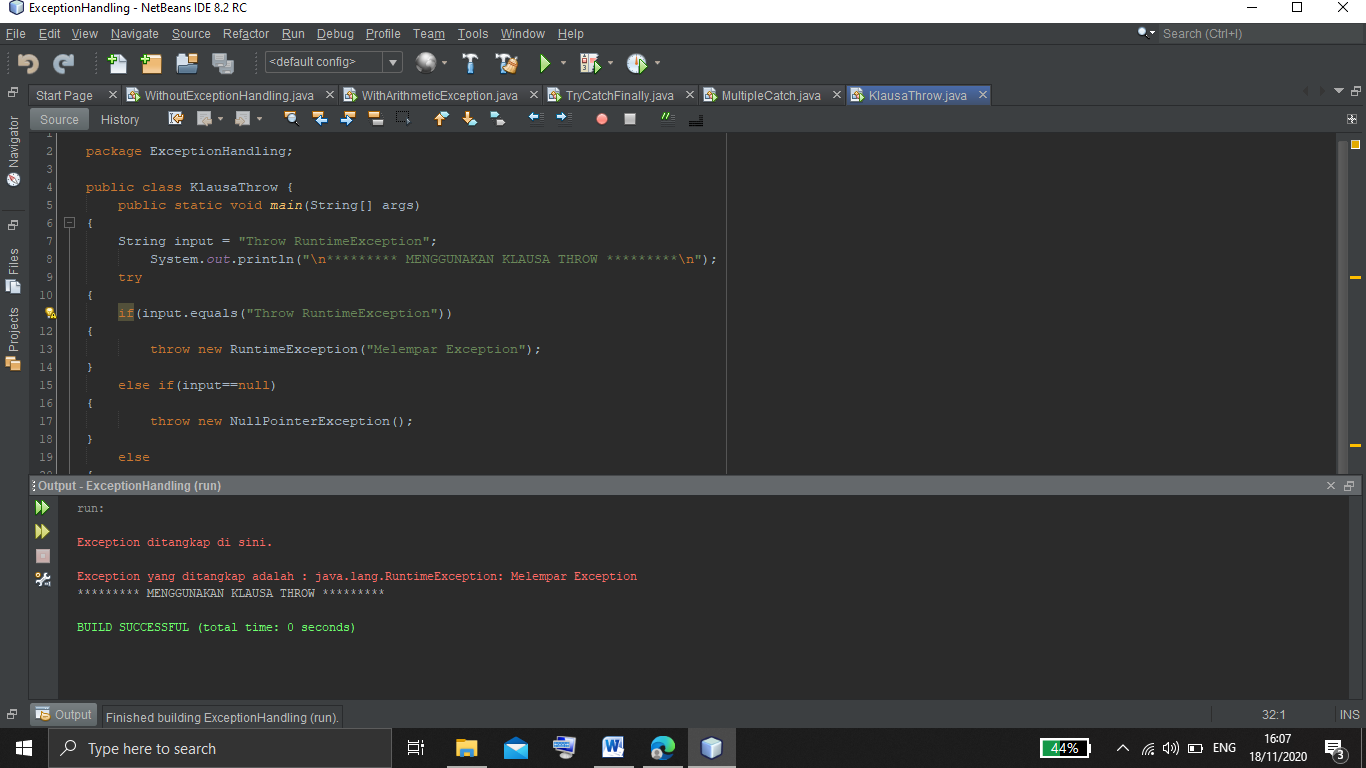


**2.5. Melemparkan exception dengan klausa throw.**

Kode Program :



Output :



**BAB III**

**ANALISA DAN KESIMPULAN**

**3.1. Program tanpa exception handling.**

Pada baris ke 7 yaitu System.out.println("\n\*\*\*\*\*\*\*\*\* PROGRAM TANPA EXCEPTION HANDLING \*\*\*\*\*\*\*\*\*\n"); fungsinya untuk mencetak saat program di dijalankan, fungsi /n diawal dan diakhir agar kalimat yang ingin kita cetak bisa tercetak pada baris baru (enter sebelum dan sesudah kalimat).

Selanjutnya kita akan melakukan hal penyebab exception seperti pada baris ke 10 yaitu int hasil = 9/0; yang artinya memberikan nilai pada int hasil = 9/0. Setelah itu kita akan mencetak sebuah output dari nilai yang diberikan tersebut seperti pada baris 12 dan 13 berikut System.out.println("Hasil pembagian = "+hasil); dan System.out.println ("Pernyataan setelah bebas dari exception");

Pada baris ke 12 seharusnya tercetak kalimat Hasil pembagian = ….. dan pada baris ke 13 akan tercetak kalimat Pernyataan setelah bebas dari exception.

Pada program yang telah kita jalankan menampilkan output seperti di bawah ini :

\*\*\*\*\*\*\*\*\* PROGRAM TANPA EXCEPTION HANDLING \*\*\*\*\*\*\*\*\*

Exception in thread "main" java.lang.ArithmeticException: / by zero at ExceptionHandling.WithoutExceptionHandling.main(WithoutExceptionHandling.java:10)

C:\Users\Lenovo\AppData\Local\NetBeans\Cache\8.2rc\executor-snippets\run.xml:53: Java returned: 1

BUILD FAILED (total time: 0 seconds)

Dari output tersebut yang bisa tercetak hanya kalimat \*\*\*\*\*\*\*\*\* PROGRAM TANPA EXCEPTION HANDLING \*\*\*\*\*\*\*\*\* karena disini tanpa exception handling yang maksutnya tanpa adanya (penanganan pengecualian) untuk menangkap error atau kesalahan yang terjadi saat program dieksekusi oleh interpreter tersebut.

Exception in thread "main" java.lang.ArithmeticException: / by zero at ExceptionHandling.WithoutExceptionHandling.main(WithoutExceptionHandling.java:10) C:\Users\Lenovo\AppData\Local\NetBeans\Cache\8.2rc\executor-snippets\run.xml:53: terjadi ketika upaya dilakukan untuk membagi dua bilangan dan bilangan dalam penyebutnya adalah nol seperti pada program diatas yang membagi angka 9 dengan angka 0 pada program Project ExceptionHandling, dalam kelas WithoutExceptionHandling di baris ke 10 dan lokasinya pada direktori C tersebut.

Jadi, kesimpulannya program tanpa exception handling ini tidak ada penanganan pengecualian untuk menangkap kesalahan yang terjadi pada saat program tersebut dijalankan. Program ini teramasuk dalam kategori runtime error.

**3.2. Menangkap exception dengan blok try-catch.**

Pada baris ke 7 dalam program tersebut terdapat kode System.out.println("\n\*\*\*\*\*\*\*\*\* PROGRAM DENGAN EXCEPTION HANDLING \*\*\*\*\*\*\*\*\*\n"); yang nantinya akan menampilkan sebuah output \*\*\*\*\*\*\*\*\* PROGRAM DENGAN EXCEPTION HANDLING \*\*\*\*\*\*\*\*\* pada baris baru.

Fungsi try pada baris ke 8 untuk menempatkan kode program yang mungkin menyebabkan terjadinya exception. Di dalam blok try adalah program yang akan dijalankan dan akan ditangkap kesalahannya jika terjadi kesalahan.

Pada baris kode ke 11-13 penyebab exception, int hasil = 9/0; fungsinya untuk membuat hasil dari 9/0. Dan untuk System.out.println("Hasil pembagian = "+hasil); fungsinya untuk menampilkan hasil untuk pada output dengan mencetak kalimat Hasil pembagian = (…hasil dari 9/0...). System.out.println("Pernyataan dalam blok try setelah bebas dari exception."); akan mencetak kalimat Pernyataan dalam blok try setelah bebas dari exception pada output.

Sedangkan fungsi di block catch adalah blok yang dijalankan jika terjadi kesalahan, jadi pesan kesalahannya di dalam catch ini yang berupa statement. Fungsi catch(ArithmeticException exc) untuk menangkap kesalahan yang terjadi pada blok try. Pada baris ke 17 dan 18 fungsinya untuk mencetak kesalahan. Sedangkan pada baris ke 20 fungsinya untuk mencetak dalam outputan.

Dari program yang telah kita buat tersebut akan mengahsilkan sebuah output seperti di bawah ini :

ArithmeticException menangkap exception hasil pembagian oleh nol.

\*\*\*\*\*\*\*\*\* PROGRAM DENGAN EXCEPTION HANDLING \*\*\*\*\*\*\*\*\*

Pernyataan di luar blok trycatch.

Exception yang ditangkap adalah : java.lang.ArithmeticException: / by zero

BUILD SUCCESSFUL (total time: 2 seconds)

Maksut dari output tersebut adalah menggunakan arithmetic exception yang artinya untuk perhitungan aritmetika seperti angka 9 dibagi dengan nol. Dan tercetak sebuah kalimat \*\*\*\*\*\*\*\*\* PROGRAM DENGAN EXCEPTION HANDLING \*\*\*\*\*\*\*\*\* terlebih dahalu baru masuk try, yang telah kita buat dengan kode System.out.println. dan tercetak kalimat Pernyataan di luar blok trycatch pada source kode akhir. Di dalam program try terjadi kesalahan, yaitu angka 9 jika dibagi nol itu tidak terdefinisi. Dan block catch mencetak kesalahannya. Pada output Exception yang ditangkap adalah : java.lang.ArithmeticException: / by zero maksutnya yaitu mencetak kode System.err.println dan menghasilkan exc pada java.lang.ArithmeticException: / by zero (membagi sebuah angka dengan nol).

Jadi, kesimpulannya pada program ini, fitur try catch berfungsi untuk menangkap kesalahan dalam membuat sebuah program, dengan menggunakan fungsi ini kita bisa menangkap kesalahan tersebut. Kesalahannya bisa berupa penulisan kode, kesalahan saat menjalankan method, dan lain-lain.

**3.3. Membuat blok try-catch-finally.**

Pada program ini hampir sama dengan program sebelumnya yang telah kita buat, tetapi jika pada program ini kita membuat blok try-catch-finally. Pada baris ke 7 terdapat kode System.out.println("\n\*\*\*\*\*\*\*\*\* MENGGUNAKAN BLOK TRYCATCH-FINALLY \*\*\*\*\*\*\*\*\*\n"); yang artinya akan menampilkan sebuah output \*\*\*\*\*\*\*\*\* MENGGUNAKAN BLOK TRYCATCH-FINALLY \*\*\*\*\*\*\*\*\* pada baris baru. Dan fungsi try pada baris ke 8 untuk program yang akan dijalankan dan akan ditangkap eksepsinya jika terjadi kesalahan.

Pada baris kode ke 11-13 penyebab exception, int hasil = 9/0; fungsinya untuk membuat hasil dari 9/0. Dan untuk System.out.println("Hasil pembagian = "+hasil); fungsinya untuk menampilkan hasil untuk pada output dengan mencetak kalimat Hasil pembagian = (…hasil dari 9/0...). System.out.println("Pernyataan dalam blok try setelah bebas dari exception."); akan mencetak kalimat Pernyataan dalam blok try setelah bebas dari exception pada output.

Sedangkan fungsi di block catch adalah blok yang dijalankan jika terjadi kesalahan, jadi pesan kesalahannya di dalam catch ini yang berupa statement. Fungsi catch(ArithmeticException exc) untuk menangkap kesalahan yang terjadi pada blok try. Pada baris ke 17 dan 18 fungsinya untuk mencetak kesalahan.

Fungsi finally pada baris ke 20 sebenarnya hanya sebagai pelengkap. Ini bisa dibayangkan sebagai footer, yaitu akan keluar di akhir atau dijalankan di akhir. Pada baris ke 22 dan 24 akan mencetak kalimat Pernyataan dalam blok finally dan Pernyataan di luar blok try-catchfinally dengan menggunakan source kode System.out.println.

Pada program tersebut akan mencetak sebuah output seperti dibawah ini :

ArithmeticException menangkap exception hasil pembagian oleh nol.

\*\*\*\*\*\*\*\*\* MENGGUNAKAN BLOK TRYCATCH-FINALLY \*\*\*\*\*\*\*\*\*

Exception yang ditangkap adalah : java.lang.ArithmeticException: / by zero

Pernyataan dalam blok finally.

Pernyataan di luar blok try-catchfinally.

BUILD SUCCESSFUL (total time: 2 seconds)

Maksut dari output tersebut adalah menggunakan arithmetic exception yang artinya untuk perhitungan aritmetika seperti angka 9 dibagi dengan nol. Dan tercetak sebuah kalimat \*\*\*\*\*\*\*\*\* MENGGUNAKAN BLOK TRYCATCH-FINALLY \*\*\*\*\*\*\*\*\* terlebih dahulu baru masuk try. Di dalam program try terjadi kesalahan, yaitu angka 9 jika dibagi nol itu tidak terdefinisi. Dan block catch mencetak kesalahannya. Pada output Exception yang ditangkap adalah : java.lang.ArithmeticException: / by zero maksutnya yaitu mencetak kode System.err.println dan menghasilkan exc pada java.lang.ArithmeticException: / by zero (membagi sebuah angka dengan nol). Blok finally itu bisa dikatakan seperti footer, jadi akan dijalankan di bagian terakhir saja dan akan tetap dijalankan walaupun tidak terjadi kesalahan atau eksepsi sekalipun. Setelah selesai akan keluar dari Try Catch dan mencetak Pernyataan di luar blok try-catchfinally.

Jadi kesimpulannya dalam membuat blok try catch finally, notasi blok bersifat perintah. Setiap blok try terdapat satu atau lebih blok catch, tetapi hanya satu blok finally. Blok catch dan blok finally harus selalu muncul dalam konjungsi dengan blok try, dan diatas urutan. Blok try harus diikuti oleh paling sedikit satu blok catch atau satu blok finally, atau keduanya. Setiap blok catch mendefinisikan sebuah penanganan exception. Header dari blok catch harus membawa satu argumen, dimana exception pada blok tersebut akan ditangani

**3.4. Membuat catch secara bertingkat.**

Dalam membuat multiple catch kita menangkap beberapa pengecualian dalam satu blok, mengurangi duplikasi kode dan meningkatkan efisiensi. Pada baris ke 7 terdapat kode System.out.println("\n\*\*\*\*\*\*\*\*\* MENGGUNAKAN MULTIPLE CATCH \*\*\*\*\*\*\*\*\*\n"); yang artinya akan menampilkan sebuah output \*\*\*\*\*\*\*\*\* MENGGUNAKAN MULTIPLE CATCH \*\*\*\*\*\*\*\*\* pada baris baru. Dan fungsi try pada baris ke 8 untuk program yang akan dijalankan dan akan ditangkap eksepsinya jika terjadi kesalahan.

Pada baris ke 11 int[ ] array = new int[9]; artinya mendeklarasikan array bilangan bulat baru dengan array terikat 0 hingga 8 dan menetapkan nilai ke indeks 9. Pada baris ke 12 dan 13 akan menampilkan sebuah output berupa kalimat Elemen array indeks ke 9 adalah…. dan Pernyataan dalam blok try setelah bebas dari exceptionm dengan menggunakan kode System.out.println.

Pada baris ke 15 catch(ArrayIndexOutOfBoundsException exc) berfungsi untuk pengecualian yang tidak dicentang, yang berarti bahwa umumnya menandakan kesalahan pemrograman, dibandingkan dengan kondisi di luar kontrol program. Pengecualian ini harus dicegah, dari pada ditangani. Pada baris ke 17 berfugsi untuk mencetak kesalahan dengan kode System.err.println.

Pada baris ke 19 catch(NegativeArraySizeException exc) berfungsi untuk menunjukkan bahwa program berusaha membuat array dengan ukuran negatif. Pada baris ke 21 berfugsi untuk mencetak kesalahan dengan kode System.err.println.

Pada baris ke 23 catch(Exception exc) berfungsi untuk mencetak kesalahan pengecualian pada exception exc. Pada baris ke 25 berfugsi untuk mencetak kesalahan dengan kode System.err.println.

Pada output program ini akan tercetak seperti dibawah ini :

\*\*\*\*\*\*\*\*\* MENGGUNAKAN MULTIPLE CATCH \*\*\*\*\*\*\*\*\*

Anda mengakses array di luar indeks yang dideklarasikan.

BUILD SUCCESSFUL (total time: 2 seconds)

Maksut dari output tersebut adalah akan tercetak kalimat \*\*\*\*\*\*\*\*\* MENGGUNAKAN MULTIPLE CATCH \*\*\*\*\*\*\*\*\* terlebih dahulu baru masuk try. Di dalam program try terjadi kesalahan, yaitu tidak dapat mendeklarasikan array bilangan bulat baru dengan array terikat 0 hingga 8 dan menetapkan nilai ke indeks 9. Selanjutnya akan mencetak dari catch pertama kali yaitu kalimat Anda mengakses array di luar indeks yang dideklarasikan pada catch(ArrayIndexOutOfBoundsException exc).

Jadi kesimpulannya untuk membuat multiple catch adalah untuk menangani beberapa pengecualian dalam satu blok tangkapan. Jika kita menangkap beberapa pengecualian dan memiliki kode serupa, maka menggunakan fitur ini akan mengurangi duplikasi kode.

**3.5. Melemparkan exception dengan klausa throw.**

Pada klausa throw ini digunakan untuk melemparkan exception yang terjadi. Pada baris ke 7 kode String input = "Throw RuntimeException"; berfungsi untuk mengidentfikasikan “Throw RuntimeException” menggunakan variable string karena berupa karakter secara khusus. Sedangkan pada baris ke 8 kode System.out.println("\n\*\*\*\*\*\*\*\*\* MENGGUNAKAN KLAUSA THROW \*\*\*\*\*\*\*\*\*\n"); akan menampilkan output \*\*\*\*\*\*\*\*\* MENGGUNAKAN KLAUSA THROW \*\*\*\*\*\*\*\*\* pada baris baru. . Dan fungsi try pada baris ke 9 untuk program yang akan dijalankan dan akan ditangkap eksepsinya jika terjadi kesalahan.

Pada baris ke 11 kode if(input.equals("Throw RuntimeException")) berfungsi untuk melemparkan excepetion jika masukan String tidak dapat diubah, memeriksa kesetaraan string ke objek lain harus dilakukan menggunakan metode daripada operator (equals).

Pada baris ke 13 kode throw new RuntimeException("Melempar Exception"); maksutnya yaitu untuk melempar pengecualian baru yang dapat dilemparkan selama operasi normal. Pada baris ke 15 kode else if(input==null) artinya **pilihan alternative dari penyeleksian** untuk mencari kondisi yang diinginkan jika input = nol. Pada baris ke 17 kode throw new NullPointerException(); terjadi karena ada object yang dipanggil dan hasilnya kosong.

Pada baris ke 19-23 terjadi jika nilai pernyataan if tidak memenuhi persyaratan dari program yang dibuat maka akan menghasilkan output kalimat Input adalah : ……. Dan Pernyataan dalam blok try setelah bebas dari pelemparan exception.

Pada baris ke 25 fungsi catch untuk menangkap kesalahan yang terjadi pada blok try dan mencetak kesalahan pada baris ke 27 dan 28 dengan kode System.err.println.

Pada output program ini adalah :

Exception ditangkap di sini.

\*\*\*\*\*\*\*\*\* MENGGUNAKAN KLAUSA THROW \*\*\*\*\*\*\*\*\*

Exception yang ditangkap adalah : java.lang.RuntimeException: Melempar Exception

BUILD SUCCESSFUL (total time: 1 second)

Maksut dari output tersebut yaitu menjalankan kode catch yang mencetak kesalahan terlebih dahulu yaitu kalimat Exception ditangkap di sini. lalu mencetak kalimat \*\*\*\*\*\*\*\*\* MENGGUNAKAN KLAUSA THROW \*\*\*\*\*\*\*\*\* terlebih dahulu sebelum memasuki try. Setelah itu mencetak kesalahan yang terakhir yaitu Exception yang ditangkap adalah : java.lang.RuntimeException: Melempar Exception.

Jadi kesimpulan dari program klausa throw ini adalah Keyword throw digunakan untuk melempar exception atau bug yang dibuat secara manual. Misalnya saat kita membuat sebuah program dan user salah memasukan input, tetapi program tidak menandakan bahwa itu terjadi error, karena input yang dimasukan tidak berpotensi akan terjadi error. Nah disinilah peranan throw, dengan menggunakan kata kunci throw, kita dapat melempar exception pada kondisi yang telah kita tentukan,

**BAB IV**

**DAFTAR PUSTAKA**

Modul Exception Handling

<https://bernazlionk.wordpress.com/2009/03/27/try-catch/>

<https://stackoverflow.com/questions/16467325/how-to-handle-arrayindexoutofboundsexception>

<https://ridahweb.wordpress.com/2017/03/04/jawaban-1-pengertian-excption-2-jenis-kenis-excption-3-pengertian-trycatchfinallydan-throw-4-perbedaan-cacth-dan-finally-5-aturan-penggunaan-tycatchfinally/>

https://www.wildantechnoart.net/2017/11/exception-handling-throw-throws-java.html