BAB 12

Dasar Exception Handling

12.1 Tujuan

Dalam bagian ini, kita akan mempelajari teknik yang dipakai dalam Java dalam menangani kondisi yang tidak biasa dalam menjalankan operasi normal dalam program. Teknik ini dinamakan **exception handling**.

Pada akhir pembelajaran, siswa mampu untuk:

- Mendefinisikan exception
- Menangani exception menggunakan blok try-catch-finally

12.2 Apa itu Exception?

Exception adalah sebuah peristiwa yang menjalankan alur proses normal pada program. Peristiwa ini biasanya berupa kesalahan(error) dari beberapa bentuk. Ini disebabkan program kita berakhir tidak normal.

Beberapa contoh dari exception yang Anda mungkin jumpai pada latihan-latihan sebelumnya adalah: exception ArrayIndexOutOfBounds, yang terjadi jika kita mencoba mengakses elemen array yang tidak ada, atau NumberFormatException, yang terjadi ketika kita mencoba melalui parameter bukan angka dalam method Integer.parseInt.

12.3 Menangani Exception

Untuk menangani exception dalam Java, kita gunakan blok try-catch-finally. Apa yang kita lakukan dalam program kita adalah kita menempatkan pernyataan yang mungkin menghasilkan exception dalam blok ini.

Bentuk umum dari blok try-catch-finally adalah,

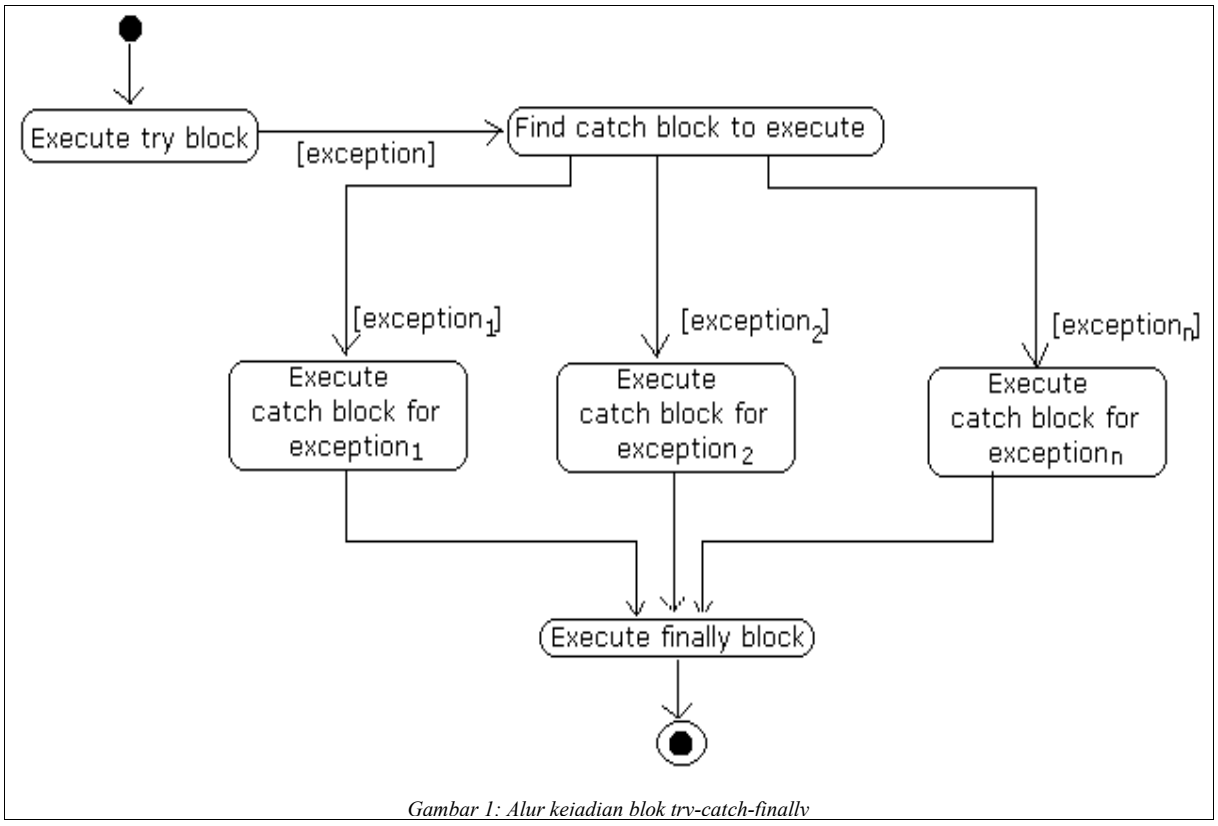
```
try{
      //tulis pernyataan yang dapat mengakibatkan exception
      //dalam blok ini
}
catch( <exceptionType1> <varName1> ){

      //tulis aksi apa dari program Anda yang dijalankan jika ada
      //exception tipe tertentu terjadi
}
. . .
```

```
catch( <exceptionType_n> <varName_n> ){
      //tulis aksi apa dari program Anda yang dijalankan jika ada
      //exception tipe tertentu terjadi
}
finally{
      //tambahkan kode terakhir di sini
}
```

Exception dilemparkan selama eksekusi dari blok *try* dapat ditangkap dan ditangani dalam blok *catch*. Kode dalam blok *finally* selalu di-eksekusi.

Berikut ini adalah aspek kunci tentang sintak dari konstruksi 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. Exception harus menjadi class pelempar atau satu dari subclassesnya.



Gambar 1: Alur kejadian blok try-catch-finally

Marilah mengambil contoh kode yang mencetak argumen kedua ketika kita mencoba menjalankan kode menggunakan argumen command-line. Perkirakan, tidak ada pengecekan dalam kode Anda untuk angka dari argumen dan kita hanya mengakses argumen kedua args[1] segera, kita akan mendapatkan exception berikut.

```
Exception in thread "main"
java.lang.ArrayIndexOutOfBoundsException: 1
      at ExceptionExample.main(ExceptionExample.java:5)
```

Untuk mencegah kejadian ini, kita dapat menempatkan kode ke dalam blok try-catch. Blok finally hanya sebagai pilihan lain saja. Sebagai contoh, kita tidak akan menggunakan blok finally.

```
public class ExceptionExample
{
     public static void main( String[] args ){

      try{
            System.out.println( args[1] );
      }catch( ArrayIndexOutOfBoundsException exp ){
            System.out.println("Exception caught!");
      }
     }
      }
```

Jadi kita akan menjalankan program lagi tanpa argumen, keluarannya akan menjadi,

```
Exception caught!
```

12.4 Latihan

12.4.1 Menangkap Exception 1

Diberikan kode berikut:

```
public class TestExceptions{

            public static void main( String[] args ){
                  for( int i=0; true; i++ ){
                        System.out.println("args["+i+"]="+
                                    args[i]);
                  }
            }
      }
```

Compile dan jalankan program TestExceptions. Keluarannya akan tampak seperti ini:

```
javac TestExceptions  one  two  three
args[0]=one
args[1]=two
args[2]=three
Exception in thread "main"
      java.lang.ArrayIndexOutOfBoundsException: 3
at TestExceptions.main(1.java:4)
```

Ubah program TestExceptions untuk menangani exception, keluaran program setelah ditangkap exception-nya akan seperti ini:

```
javac TestExceptions  one  two  three
args[0]=one
args[1]=two
args[2]=three
Exception caught:
      java.lang.ArrayIndexOutOfBoundsException: 3
      Quiting...
```

12.4.2 Menangkap Exception 2

Melakukan percobaan pada beberapa program yang telah Anda tulis adalah hal yang baik sebelum menghadapi exception. Karena pada program di atas Anda tidak menangkap exception, maka eksekusi dengan mudahnya berhenti mengeksekusi program Anda. Kembali kepada program diatas dan gunakan penanganan exception.