Analisa Penggunaan try dan catch

---

1. **Blok try**:
    - Dalam blok **try**, sebuah objek **Exception** diciptakan dan dilempar menggunakan **throw new Exception("Here's my Exception");**.
    - Exception ini membawa pesan "Here's my Exception".
2. **Blok catch (Exception e)**:
    - Blok **catch** menangkap exception yang dilempar dari blok **try**.
    - Di dalam blok **catch**, berbagai metode exception digunakan untuk mendapatkan informasi tentang exception yang ditangkap:
        - **System.out.println("Caught Exception");** mencetak pesan yang menunjukkan bahwa exception telah ditangkap.
        - **System.out.println("e.getMessage(): " + e.getMessage());** mencetak pesan dari exception, yaitu "Here's my Exception".
        - **System.out.println("e.toString(): " + e.toString());** mencetak representasi string dari exception, yang mencakup nama kelas exception dan pesan error.
        - **System.out.println("e.printStackTrace():");** mencetak stack trace dari exception, yang menunjukkan urutan panggilan metode yang menyebabkan exception tersebut.
        - **e.printStackTrace();** benar-benar mencetak stack trace ke konsol.

Kesimpulan

- Program ini memperlihatkan bagaimana cara memunculkan dan menangkap exception secara eksplisit menggunakan **throw** dalam blok **try** dan menangani exception tersebut dalam blok **catch**.
- Blok **catch** memberikan berbagai cara untuk mengakses informasi tentang exception yang ditangkap:
    - **e.getMessage()** mengembalikan pesan error yang diberikan saat exception dibuat.
    - **e.toString()** mengembalikan representasi string dari exception, termasuk nama kelas dan pesan error.
    - **e.printStackTrace()** mencetak stack trace, yang sangat berguna untuk debugging.
- Program ini memberikan contoh yang baik tentang bagaimana menangani exception dengan cara yang informatif, membantu pengembang memahami dan memperbaiki masalah yang menyebabkan exception tersebut.