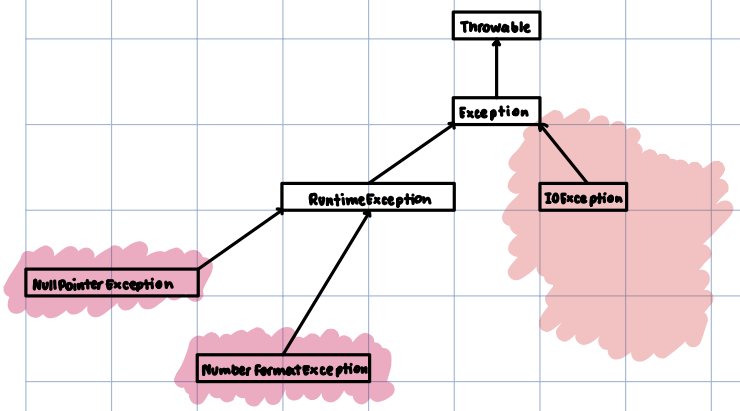
Eccezioni in Java

le eccezioni sono oggetti



Checked Exception: è necessario fornire informazioni aggiuntive al compilatore per controllare l'effettiva gestione dell'eccezione

Unchecked Exception: del tutto analoghe a C++

Unchecked Exception predefinite

- **NullPointerException**: sollevata quando si prova ad accedere ad un oggetto tramite la sua reference ma questa è null

```
Razionale a;
a.stampa (System.out);
```

- **NumberFormatException**: generata da Integer.parseInt, Boolean.parseBoolean,
- **InputMismatchException**:

```
Scanner sc = new Scanner (System.in);
int i = sc.nextInt();
```

Checked Exceptions

Tutte le eccezioni che sono sottoclasse di Exception ma non di RuntimeException sono dette controllate

```
public class Fallimento extends Exception { }
```

Se un metodo solleva un'eccezione controllata, quel metodo e tutti i metodi che lo richiamano devono specificarlo nella signature **oppure** deve catturare e gestire l'ecce.

```
public static void m () throws Fallimento {
    throw new Fallimento(); }
```

```
public class Fallimento extends Exception {
}
```

```
public static void m1() throws Fallimento {
    m();}
```

perché m() lancia un'eccezione controllata.

Checked vs Unchecked Exceptions

- **Checked:**
  - rappresenta una situazione anomala prevedibile anche se un metodo, una libreria, una API viene usata in maniera corretta.

- **Unchecked:**
  - rappresenta un errore nella logica del programma
  - non prevedibile dal compilatore
  - es: NullPointerException, ArrayIndexOutOfBoundException...