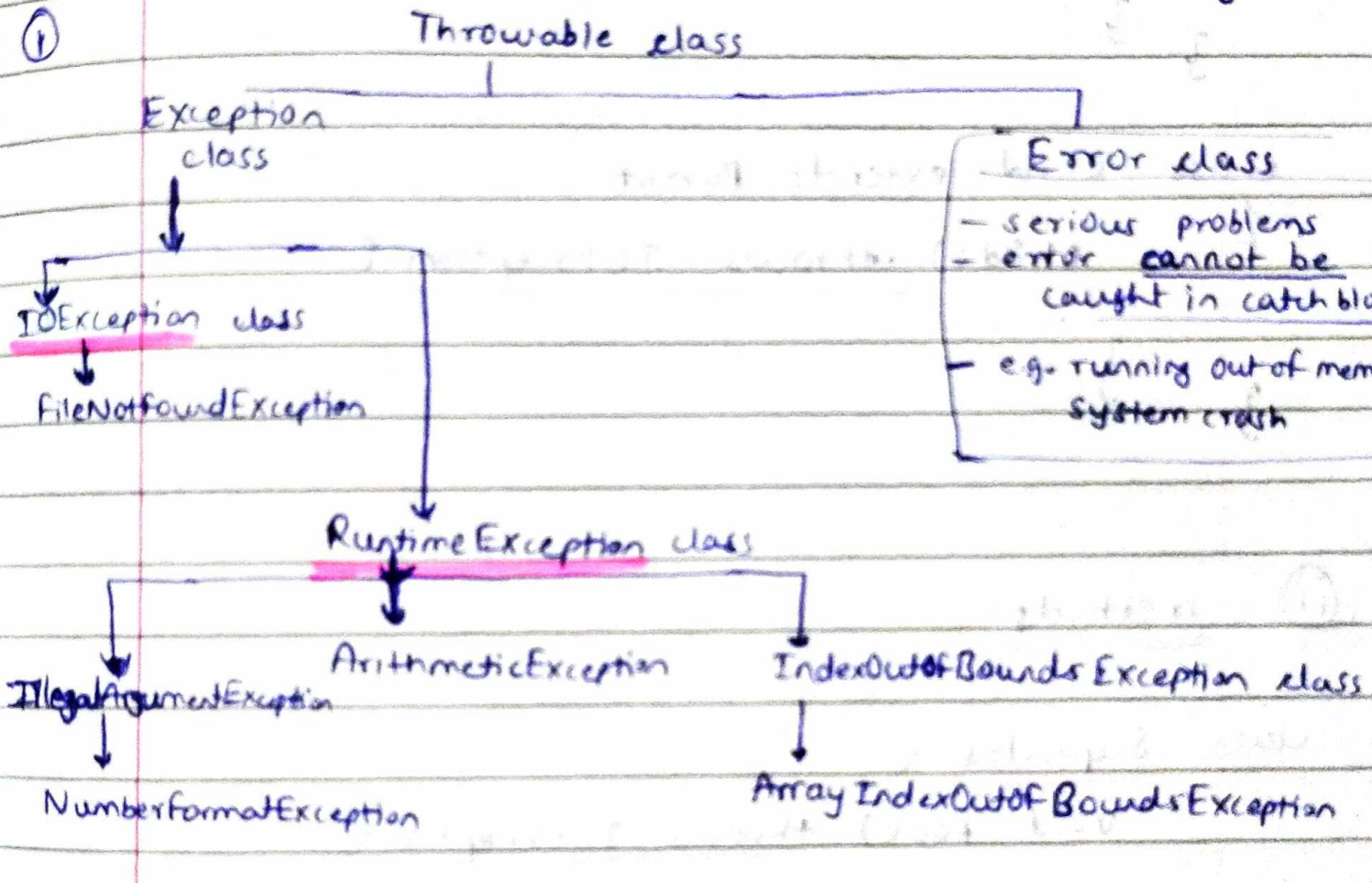
(My Doubts' clearance → exception handling)

(1)



There are many other exception classes.
I've shown only few.

(2)

finally{} cannot exist without try block

(3)

```
try {
  ---  } non-problematic lines
  ---
  ---  ← problematic statement
}
finally {
} return x;
```

even though there was exception caused in try, there is return statement in finally which will execute

④ Overriding ~~in case of~~ with throws

(i) constructors of parent & child classes

(ii) methods in parent & child classes

(i) Constructor

```
class Parent {
    Parent() throws FileNotFoundException {
    }
}

class Child extends Parent
{
    Child() throws IOException {
    }
}
```

(ii) methods

```
class Superclass {
    void foo() throws IOException {
    }
}

class Subclass extends Superclass {
    @Override
    void foo() throws FileNotFoundException {
    }
}
```

⑤ throw with unchecked exception

(i) with try-catch (ii) without try-catch

(i) throw - unchecked exception - with try-catch

```
try {
    if(y==0) {
        throw new ArithmeticException("div. by 0");
    }
}
catch(ArithmeticException e) {
    System.out.println(e.getMessage());
}
```

(ii) throw - unchecked exception - w/o try-catch

Without try-catch

```
if (y==0)
    throw new ArithmeticException("div. by 0");
```

(to print this message, try-catch is required)

⑥ throw with checked exception

(i) without try-catch

```
p.s.v.m(String[] args) throws IOException {
    throw new IOException("IIo error");
}
```

(ii) with try-catch

```
p.s.v. main(String[] args).
{
    try{
        throw new IOException("IIo error");
    }
    catch(IOException e){
        sysout.(e.getMessage());
    }
}
```

(7)

* Custom class exception as checked exception

for checked exception,

→ Custom class extends Exception class or its subclasses except RuntimeException

```
public class MyCheckedException extends Exception {
        public MyCheckedException() {
        } super();
        public MyCheckedException(String message) {
            super(message);
        }
}

public class Test{
    P.S.V.m(String[] args)
    {
        try{
            throw new MyCheckedException("this is checked excpn");
        }
        catch(MyCheckedException e)
        {
            Sysout.(e.getMessage());
        }
    }
}
```

this can be written w/o try-catch also

---

(8) custom class exception as unchecked exception

→ for unchecked exception,

custom class extends RuntimeException class or one of its subclasses

Syntax same as above