Exception Handling:

→ In a normal flow of execution, if an abnormal situation occurs then jvm will terminate the program by throwing the exception class object.
→ Exception

Exception Handling

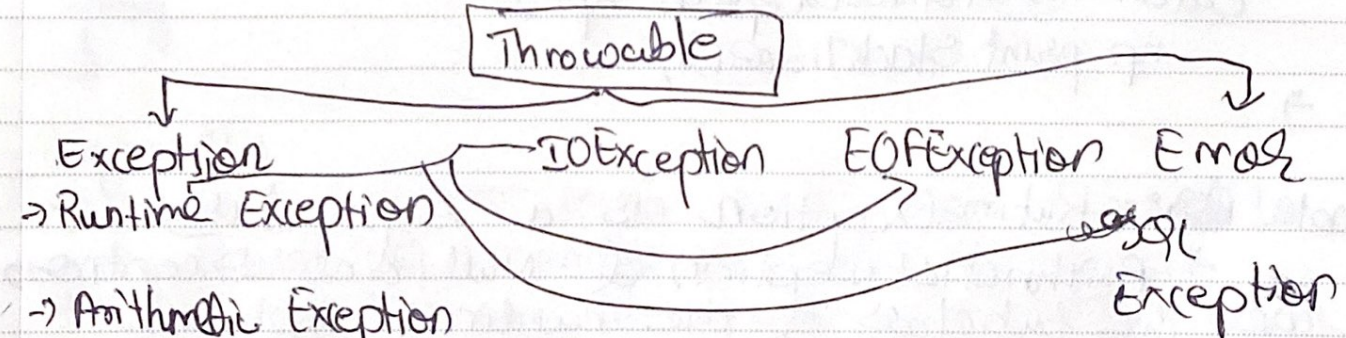
→ For exception handling we can use

```
try {
    risky code;
}
catch (ExceptionClassname reference) {
 msg related to exception.
}
```

Exception Handling means, to skip the instruction with the exception & execute the remaining part of the program correctly.
→ or to stop the inappropriate termination of program.

we have two predefined classes.

1) Class
2) Object: Object class is the super class for all the classes.



Exception is classified into 2 types:

1) Un-checked Exceptions: The exceptions which are unknown to the compiler

2) checked Exceptions
   → The exceptions which are known to the compiler then those are checked exceptions.

~~to s.toString()~~

System.out.println(s.toString());
→ toString convert object into message

System.out.println(s.getMessage());
→ give message

java.lang.ArithmeticException
→ to see the arithmetic exception

s.printStackTrace();
→ it will trace back

catch(NullPointerException np)
np.printStackTrace();
→

Note :- RutimeException is a superclass
→ ArithmeticException & NullPointerException are the subclass of the runtimeException

RutimeException is a subclass of the the exception.

→ The order of the catch block must be from subclass name to the superclass name

Format for the exception handling

1)
```
try {
  risky code;
}
catch (ExceptionClass ref){
  msg
}
```

2)
```
try {
  risky code;
  try {
    risky code;
  }
  catch (ExceptionClass ref){
    msg
  }
}
catch (Exception Class ref){
  msg
}
```

3)
```
try {
  risky code;
}
catch (ExceptionClass ref
  msg
}
catch (ExceptionClass ref){
  msg
}
```

4)
```
try {
  risky code;
}
catch (ExceptionClass
  ref) {
  msg
}
```

→ finally ~~when exception~~ is a block which is used to deallocate the memory or close the objects

→ when exception is not catch by catch then finally helps to block.

→ is a block which gets executed irrespective

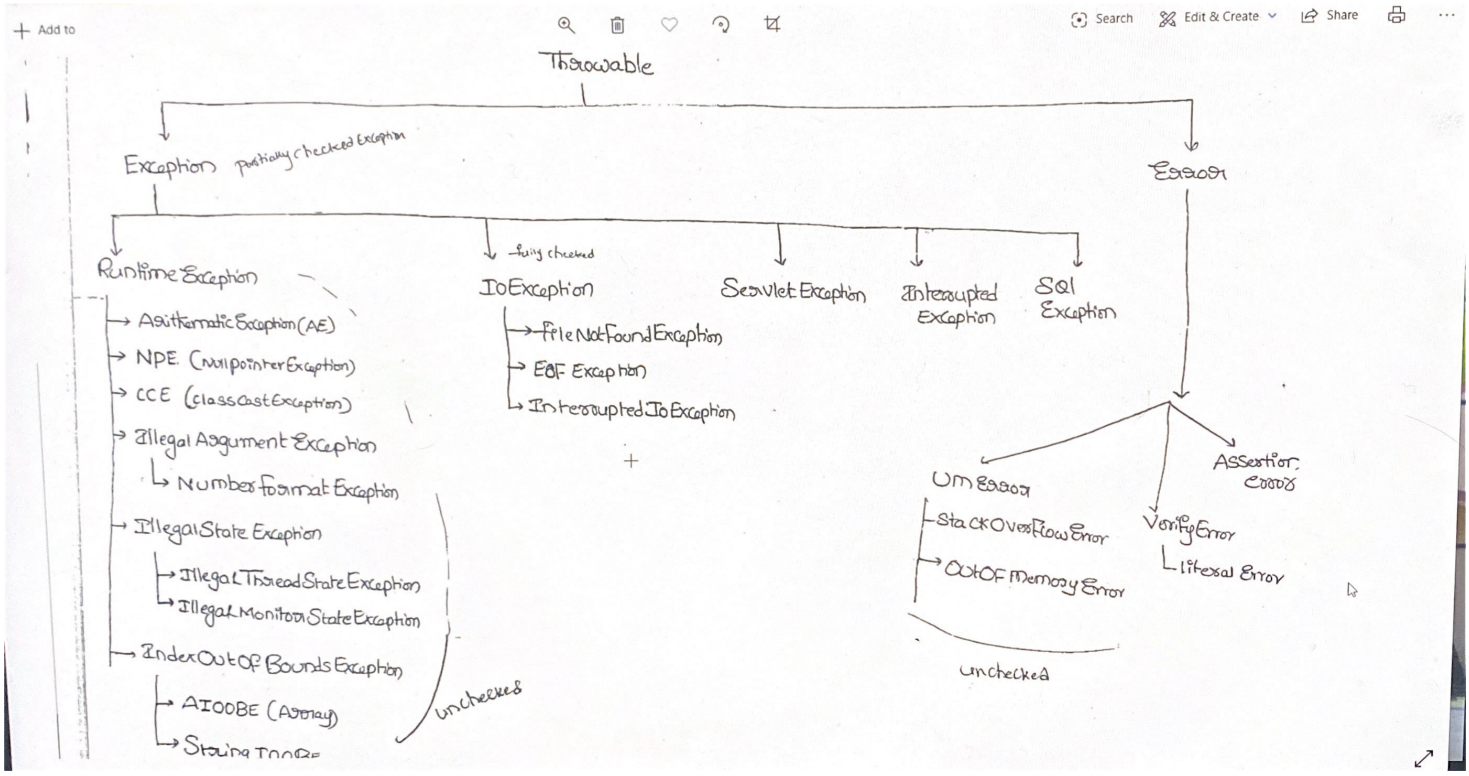
of the exception
eg:
jdbc:
Connection con;
con.close();

-> finally can be written with catch only.

-> throw is the keyword used to generate the user defined exception
-> throw Handled only checked exception

-> to handle unchecked exception we have to use try catch

-> to handle checked exceptions we can use throws and try, catch


Add to
Search
Edit & Create
Share
Throwable
Exception
partially checked Exception
Error
Runtime Exception
fully checked
IoException
ServletException
Interrupted Exception
SQL Exception
ArithmaticException (AE)
NPE (NullPointerException)
CCE (classcastException)
IllegalArgumentException
NumberFormatException
IllegalStateException
IllegalThreadStateException
IllegalMonitorStateException
IndexOutOfBoundsException
AIOOBE (Array)
FileNotFoundException
EOF Exception
InterruptedIOException
unchecked
VmError
StackOverflowError
OutOfMemoryError
VerifyError
LinkageError
AssertionError
unchecked