# 8. Error Handling

## Lesson 3:
## Handling exceptions

# 8.5. Handling Exceptions

## 8.5.1. Try Blocks

- The first step in constructing an exception handler is to enclose the code that might throw an exception within a try block.

```
try
{
   code
}
catch and finally blocks . . .
```

- If an exception occurs within the try block, that exception is handled by an exception handler associated with it. To associate an exception handler with a try block, you must put a catch block after it.

# 8.5. Handling Exceptions

## 8.5.2. Catch Blocks

- You associate exception handlers with a try block by providing one or more catch blocks directly after the try block.

- Each catch block is an exception handler and handles the type of exception indicated by its argument. The argument type, ExceptionType, declares the type of exception that the handler can handle and must be the name of a class that inherits from the Throwable class. The handler can refer to the exception with name.

- The catch block contains code that is executed if and when the exception handler is invoked. The runtime system invokes the exception handler when the handler is the first one in the call stack whose ExceptionType matches the type of the exception thrown. The system considers it a match if the thrown object can legally be assigned to the exception handler's argument.

```
try
{

}
catch (ExceptionType name)
{

}
catch (ExceptionType name)
{

}
```

UCSC

BIT

# 8.5. Handling Exceptions
## 8.5.3. Finally Block

- The finally block always executes when the try block exits.
- This ensures that the finally block is executed even if an unexpected exception occurs. But finally is useful for more than just exception handling — it allows the programmer to avoid having cleanup code accidentally bypassed by a return, continue, or break.
- Putting cleanup code in a finally block is always a good practice, even when no exceptions are anticipated.
- The finally block is optional.

UCSC

BIT

# 8.5. Handling Exceptions

## 8.5.4. Try-Catch Mechanism

```
try
{
… normal program code
}
catch(Exception e)
{
… exception handling code
}
finally
{
…optional
}
```

# 8.5. Handling Exceptions

## 8.5.5. Passing the Exception

- In any method that might throw an exception, you may declare the method as “throws” that exception, and thus avoid handling the exception yourself

```
public void myMethod() throws IOException
{
   … normal code with some I/O
}
```