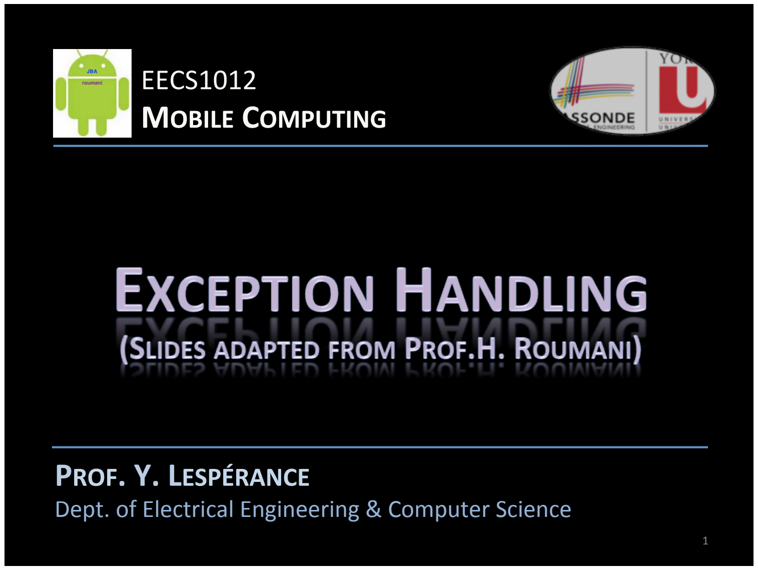
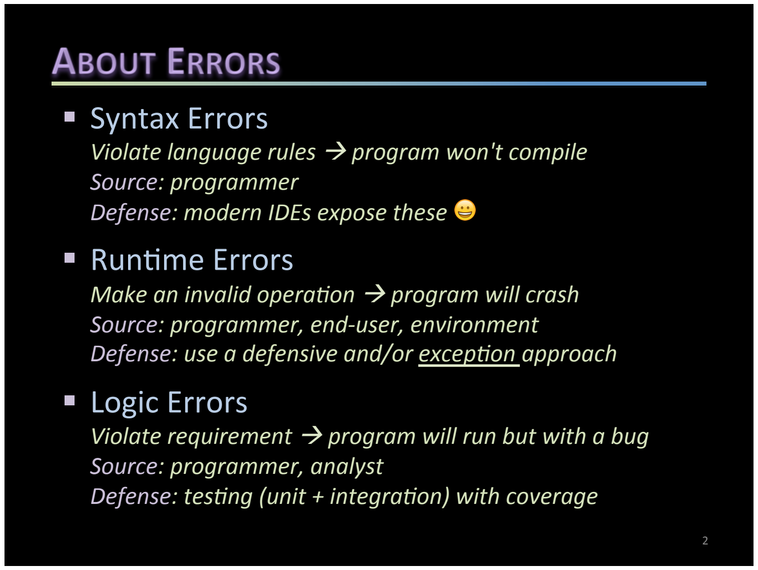


EECS1012
Mobile Computing

Exception Handling
(Slides adapted from Prof.H. Roumani)

Prof. Y. Lespérance
Dept. of Electrical Engineering & Computer Science

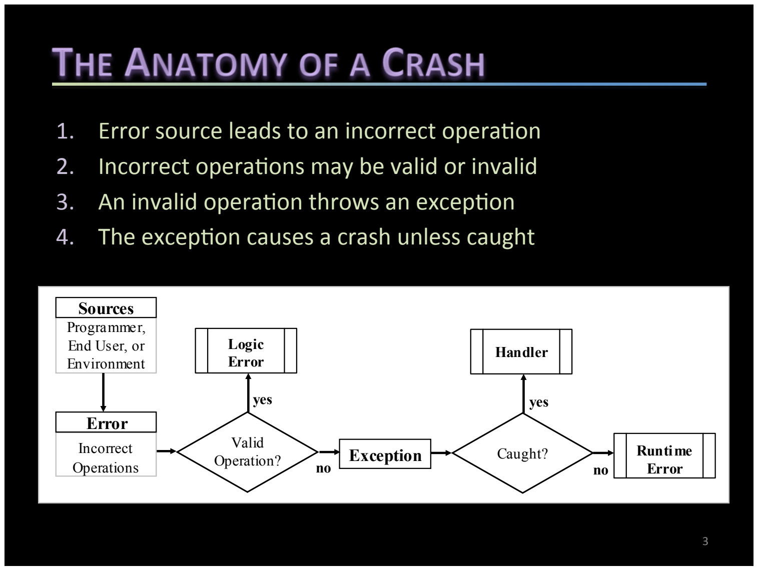
1



About Errors

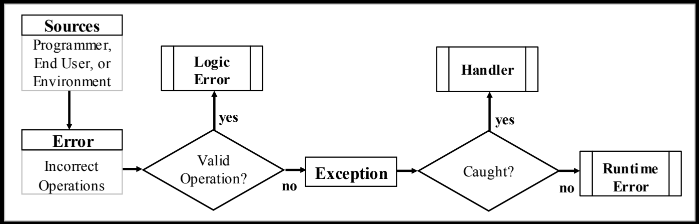
- Syntax Errors
  Violate language rules → program won't compile
  Source: programmer
  Defense: modern IDEs expose these 😀
- Runtime Errors
  Make an invalid operation → program will crash
  Source: programmer, end-user, environment
  Defense: use a defensive and/or exception approach
- Logic Errors
  Violate requirement → program will run but with a bug
  Source: programmer, analyst
  Defense: testing (unit + integration) with coverage

2

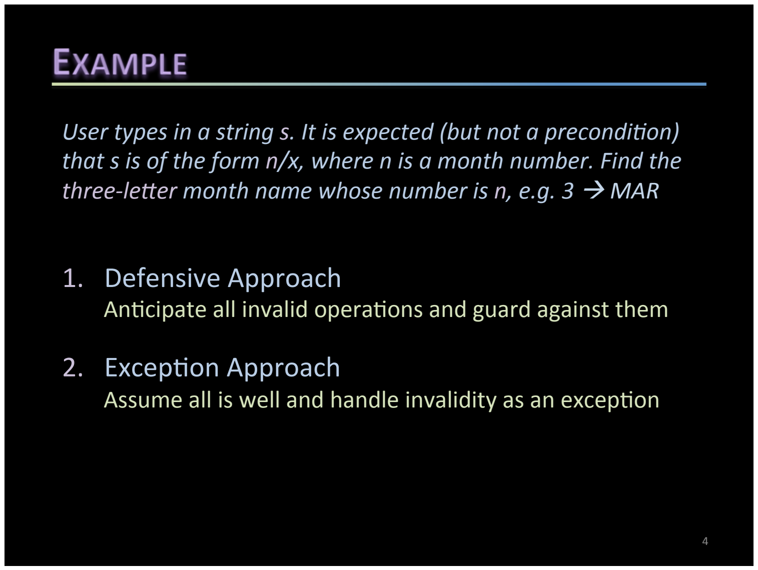


The Anatomy of a Crash

1. Error source leads to an incorrect operation
2. Incorrect operations may be valid or invalid
3. An invalid operation throws an exception
4. The exception causes a crash unless caught



3

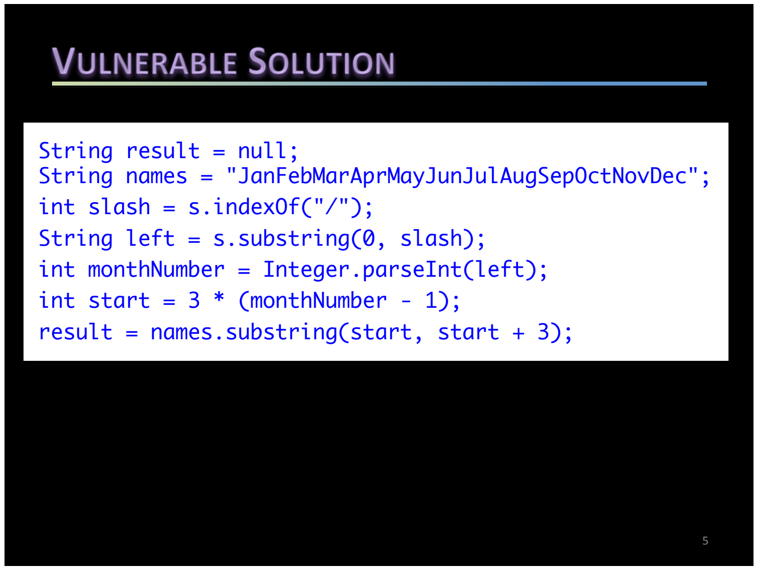


Example

User types in a string s. It is expected (but not a precondition) that s is of the form n/x, where n is a month number. Find the three-letter month name whose number is n, e.g. 3 → MAR

1. Defensive Approach
   Anticipate all invalid operations and guard against them
2. Exception Approach
   Assume all is well and handle invalidity as an exception

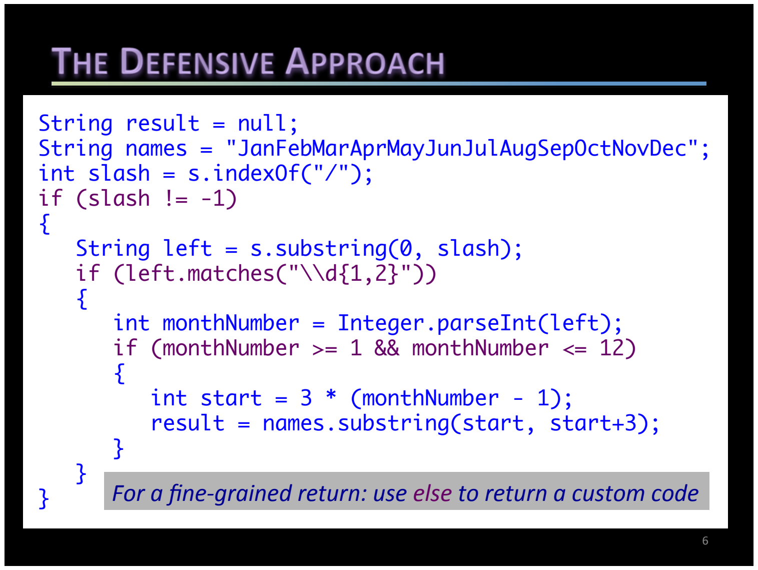
4



Vulnerable Solution

```
String result = null;
String names = "JanFebMarAprMayJunJulAugSepOctNovDec";
int slash = s.indexOf("/");
String left = s.substring(0, slash);
int monthNumber = Integer.parseInt(left);
int start = 3 * (monthNumber - 1);
result = names.substring(start, start + 3);
```

5

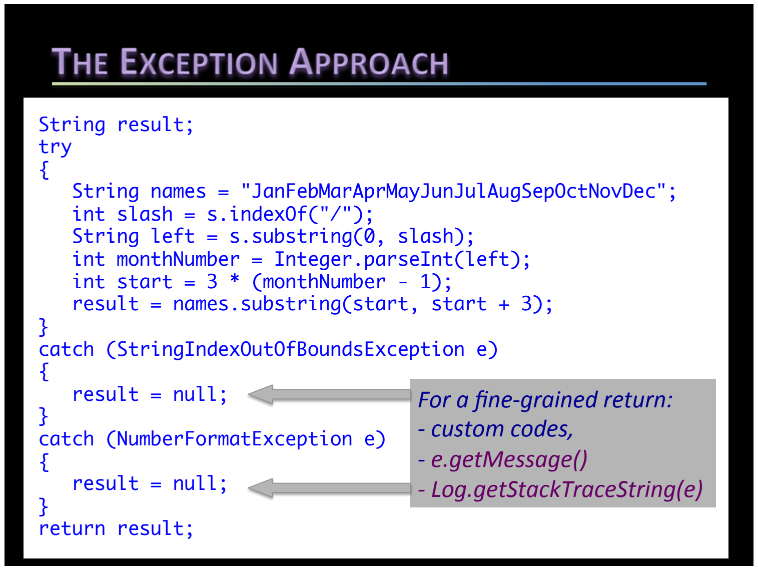


The Defensive Approach

```
String result = null;
String names = "JanFebMarAprMayJunJulAugSepOctNovDec";
int slash = s.indexOf("/");
if (slash != -1)
{
   String left = s.substring(0, slash);
   if (left.matches("\\d{1,2}"))
   {
      int monthNumber = Integer.parseInt(left);
      if (monthNumber >= 1 && monthNumber <= 12)
      {
         int start = 3 * (monthNumber - 1);
         result = names.substring(start, start+3);
      }
   }
}
```

For a fine-grained return: use else to return a custom code

6

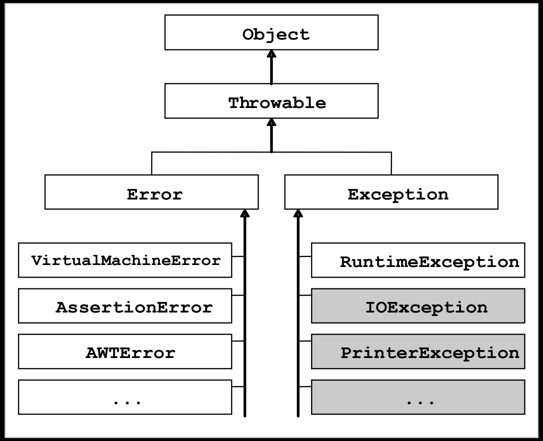


The Exception Approach

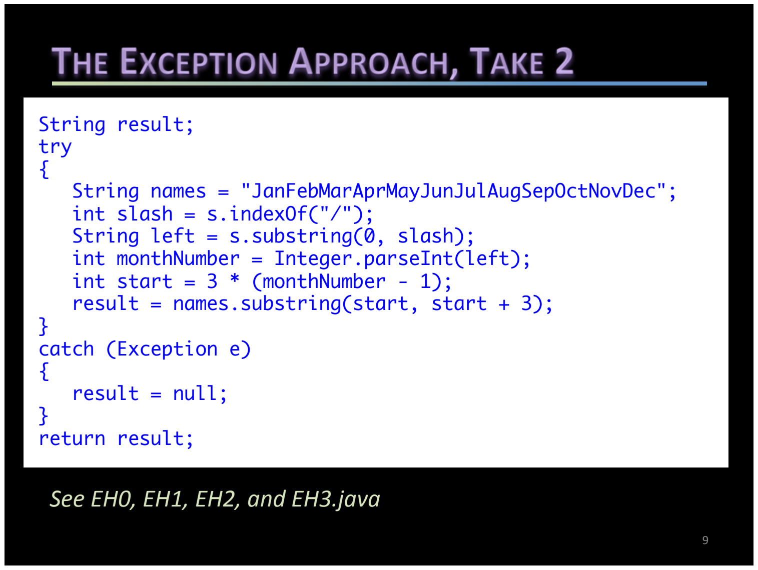
```
String result;
try
{
   String names = "JanFebMarAprMayJunJulAugSepOctNovDec";
   int slash = s.indexOf("/");
   String left = s.substring(0, slash);
   int monthNumber = Integer.parseInt(left);
   int start = 3 * (monthNumber - 1);
   result = names.substring(start, start + 3);
}
catch (StringIndexOutOfBoundsException e)
{
   result = null;
}
catch (NumberFormatException e)
{
   result = null;
}
return result;
```

For a fine-grained return:
- custom codes,
- e.getMessage()
- Log.getStackTraceString(e)

The Exception Hierarchy



8



The Exception Approach, Take 2

```
String result;
try
{
   String names = "JanFebMarAprMayJunJulAugSepOctNovDec";
   int slash = s.indexOf("/");
   String left = s.substring(0, slash);
   int monthNumber = Integer.parseInt(left);
   int start = 3 * (monthNumber - 1);
   result = names.substring(start, start + 3);
}
catch (Exception e)
{
   result = null;
}
return result;
```

See EH0, EH1, EH2, and EH3.java

9