Exceptions

Professor: Suman Saha

Exceptions

- So far, we assumed each function will run from start to its return points, But a program also needs to handle exceptions:
  - Out of memory
  - Divide by zero
  - File not found
  - …

Workaround in C language

PennState

- Return a special value (e.g., NULL) when a real value cannot be computed
- Return an explicit ”status” to indicate if an exception happens
- Rely on the caller to pass in the exception handling code (e.g., via function pointer)

Exception handling is not enforced in some languages (error-prone)

Exception Handling

PennState

- A language feature that
  - Isolates error-checking code
  - Direct execution to a handler when appropriate

Define exceptions?

How does exception change control flow?

Define Exceptions

PennState

- Pre-defined exceptions
  - e.g., divide by zero, out-of-bound array access
- User-defined exceptions
  - Typically, defined as a special kind of class

```
public class myException extends Exception
{ private int i;
  public myException(int x) {
    this.i = x; }
  public int getI() { return i; }
}
```

Throwing Exceptions

PennState

- Pre-defined exceptions
  - Automatically generated
- User-defined exceptions
  - Generated by “throw” keyword

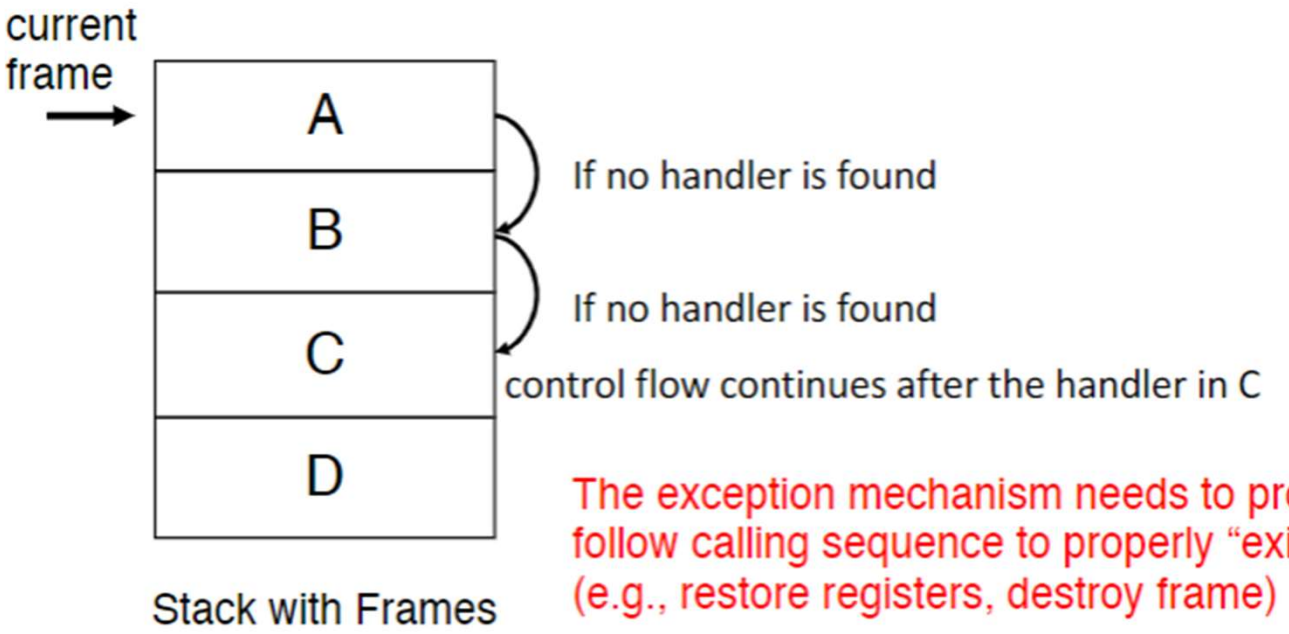
```
int foo ()
{
  ...
  throw new myException(10);
  ...
}
```

Catching Exceptions

PennState

```
try
{
   .. code that might throw exceptions ..
}
catch (Exception1 e1) {.. handling code ..}
catch (Exception2 e2) {.. handling code ..}
...
finally {.. this code always executes,
            e.g. cleanup routine ..}
```

Exception Propagation



The exception mechanism needs to properly follow calling sequence to properly "exit" functions (e.g., restore registers, destroy frame)

Exception (Example)

PennState

```
1  void foo () {
2      try {
3          throw new Exception1();
4          print ("A");
5          throw new Exception2();
6          print ("B");
7      }
8      catch(Exception1 e1) {
9          print "handler1";
10     }
11     print ("C");
12     throw new Exception2();
13 }
14 void main () {
15     try {
16         try {
17             foo();
18             print ("D");
19         }
20         catch(Exception1 e1) { print "handler2"; }
21         print ("E");
22     }
23     catch(Exception2 e2) { print "handler3"; }
24 }
```

Reading and Exercises

PennState

Reading

- Chapter: 9.4 (Michael Scott Book)