Preconditions

- Functions often have requirements on their inputs

```
// Return maximum element in A[i..j]
int findMax(int[] A, int i, int j) { ... }
```

  - A is nonempty
  - A isn't null
  - i and j must be nonnegative
  - i and j must be less than A.length
  - i < j (maybe)
- These are called *preconditions*

CMSC 330: Organization of Programming Languages

Exceptions

Signaling Errors

- Style 1: Return invalid value

```
// Returns value key maps to, or null if no
// such key in map
Object get(Object key);
```

  - Disadvantages?

Dealing with Errors

- What do you do if a precondition isn't met?
- What do you do if something unexpected happens?
  - Try to open a file that doesn't exist
  - Try to write to a full disk

Problems with These Approaches

- What if all possible return values are valid?
  - E.g., `findMax` from earlier slide
  - What about errors in a constructor?
- What if client forgets to check for error?
  - No compiler support
- What if client can't handle error?
  - Needs to be dealt with at a higher level
- Poor modularity- exception handling code becomes scattered throughout program
- 1996 Ariane 5 failure classic example of this …

Signaling Errors (cont'd)

- Style 2: Return an invalid value and status

```
static int lock_rdev(mdk_rdev_t *rdev) {
  ...
  if (bdev == NULL)
    return -ENOMEM;
  ...
}

// Returns NULL if error and sets global
// variable errno
FILE *fopen(const char *path, const char *mode);
```


Why Ariane 5 failed

- SRI tried to convert a floating point number out of range to integer. Therefore it issued an error message (as a 16 bit number). This 16 bit number was interpreted as an integer by the guidance system and caused the nozzle to move accordingly.
  - The backup SRI performed according to specifications and failed for the same reason.
- Ada range checking was disabled since the SRI was supposedly processing at 80% load and the extra time needed for range checking was deemed unnecessary since the Ariane 4 software worked well.
- The ultimate cause of the problem was that the Ariane 5 has a more pronounced angle of attack and can move horizontally sooner after launch. The "bad value" was actually the appropriate horizontal speed of the vehicle.

Ariane 5 failure

Design issues: In order to save funds and ensure reliability, and since the French Ariane 4 was a successful rocket, the Inertial Reference System (SRI) from Ariane 4 was reused for the Ariane 5.

What happened?: On June 4, 1996 the Ariane 5 launch vehicle failed 39 seconds after liftoff causing the destruction of over $100 million in satellites.

Cause of failure: The SRI, which controls the attitude (direction) of the vehicle by sending aiming commands to the rocket nozzle, sent a bad command to the rocket causing the nozzle to move the rocket toward the horizontal.

The vehicle tried to switch to the backup SRI, but that failed for the same reason 72 millisec earlier.

The vehicle had to then be destroyed.

Throwing an Exception

- Create a new object of the class Exception, and throw it

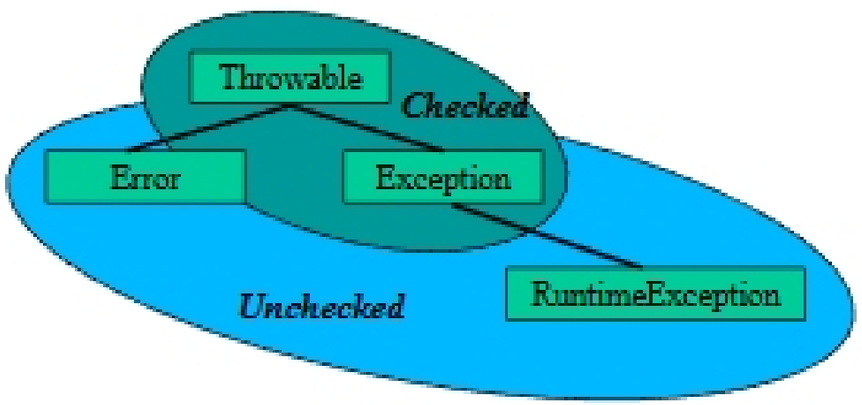
```
if (i >= 0 && i < a.length )
  return a[i];
throw new ArrayIndexOutOfBounds();
```

- Exceptions thrown are part of the return type in Java
  - When overriding method in superclass, cannot throw any more exceptions than parent's version

Better approaches: Exceptions in Java

- On an error condition, we *throw* an exception
- At some point up the call chain, the exception is *caught* and the error is handled
- Separates normal from error-handling code
- A form of non-local control-flow
  - Like goto, but structured

Exception Hierarchy



Method throws declarations

- A method declares the exceptions it might throw
  - public void openNext() throws UnknownHostException, EmptyStackException { … }
- Must declare any exception the method might throw
  - Unless it is caught in (masked by) the method
  - Includes exceptions thrown by called methods
  - Certain kinds of exceptions excluded