Exceptions and Refactoring

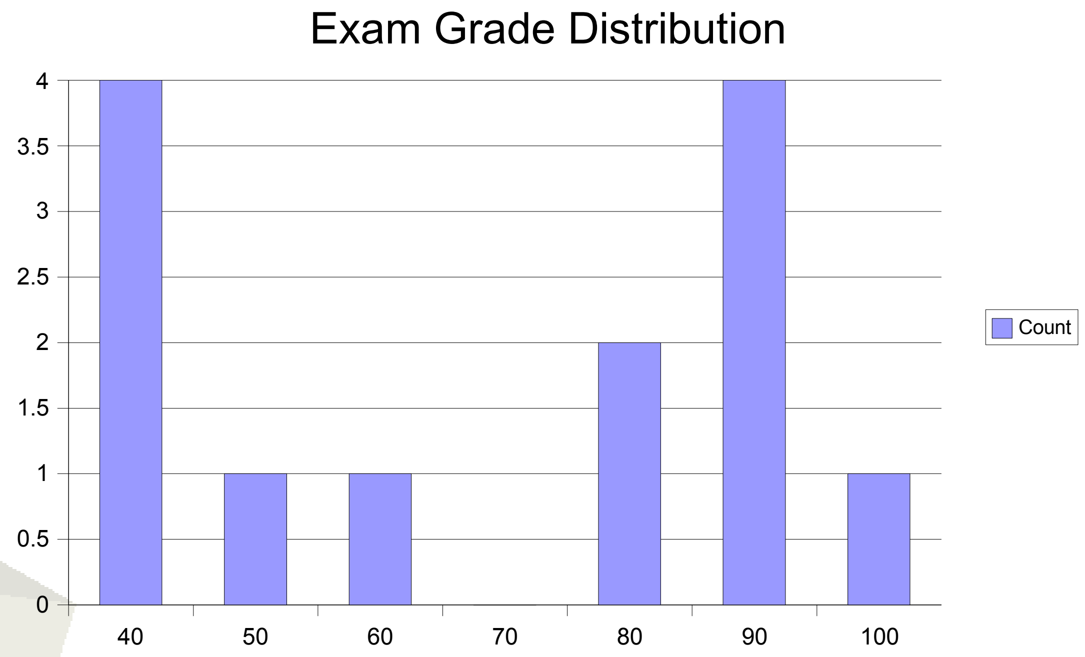
3-21-2006

Opening Discussion

- Do you have any questions about the quiz?
- What did we talk about in the class before Spring Break?
- Do you have any questions about the assignment?

Midterm Grades

- The distribution was distinctly bimodal. The median was an 81 so there won't be a curve.
- Remember that this is only 15% of your grade, but if you did poorly on it you will want to make sure you improve in the second half of the semester.



Errors in C

- If a function in C didn't do the right thing for one reason or another, how did it tell you that? What did you need to do in your code to make sure you caught those errors? Did you always do it?

Exceptions

- There are a number of problems with using return values to give you errors. The biggest is that not all functions have invalid return values. The second is that programmers are very prone to ignore them which causes the error to propagate. Using a flag has the same problem and is even more often ignored.
- For these reasons, Java includes exceptions. When an error occurs, an exception is thrown. Either something must catch the error and handle it, or the code crashes at that point and prints a stack trace. Errors are much less prone to propagate.

Exception Syntax

- There are several syntactic elements that deal with exceptions. If you raise an exception use the throw keyword. The single argument must be a subtype of Throwable.
- More often you will be calling code that can throw an exception. In that case you need to put the call in a try block and after the try block you put a catch block with the type you want to catch. There can be multiple catch blocks that catch different types.
- After the catch blocks you can put a finally block. This will always happen, whether an exception is thrown or not.

Two Types of Exceptions

- You have written lots of code that can throw exceptions without actually putting in try-catch blocks. That is because the exceptions that your code would throw are subtypes of RuntimeException. These exceptions are unchecked, which means you aren't forced to deal with them.
- Exceptions that are not RuntimeExceptions are called checked exceptions and you must either catch them or declare that your function throws them. We will see a lot more of this when we get into I/O.

Last Advantage

- Good exceptions give you all the information you need to debug your code. All exceptions have a method called printStackTrace which can be extremely helpful.
- Well written exceptions go further and have a string message that contains the information you need. For example, overflows of arrays or lists should tell you what index you asked for and how big the array or list actually was.

Coding Exceptions

- Together I want us to do some testing of exceptions in our code for parsing formulas.
- Now I want you to write some code that will do a reverse polish calculator using Scanner and one of your stacks. It should tell the user something useful when they screw something up.

Refactoring

- Many programmers, especially in smaller development groups, have moved to so-called agile development methods.
- A major aspect of these methods is that when you first write code you make something that works and does just what you need at that time. Later on you might have to come back and add functionality. Adding this functionality typically requires refactoring.
- After code has been altered many times the quality degrades. Refactoring is the process of altering the structure of code without changing the functionality.

Some Basic Refactoring

- You have already seen the refactoring menu in Eclipse because we have used it in class. We have only used the most basic option of renaming things or moving them around.
- Other refactorings include extracting a section of code into a separate method because you found you do it more than once. You can also refactor parts of a class out into their own class because you find it could be helpful in other context.
- All of these things can be done by hand, but tools like Eclipse make sure that if your code worked before you do it, it will work after you do it. Agile method programmers rely on testing to enforce this.

Drawing Program

- Let's go to out drawing program and write some extra code of it. Next class we will actually make it so that we can draw with it. For that to be useful we need to set up some more of the GUI and add some classes to it first.

Minute Essay

- Why are many exceptions in Java unchecked?
- The design for assignment #5 is due Thursday.