Pick up your prelim in handback room, Gates 2016!
1. We may have made a mistake in grading or inputting grades.
2. The purpose of a test is to provide feedback about what has been taught/learned and to make adjustments for the future.
Solutions available Friday at 3PM. You may want to wait until you see solutions before asking for a regrade.
You may want to switch to S/U grade. Talk to your advisor.

27 people are taking it S/U.

1. Their mouse had a mean time between failure of a week... it would jam up irreparably, or... jam up on the table... Xerox says it can't be built for $400. I want a $10 mouse that will never fail and can be mass produced, because it's going to be the primary interface of the computer...

2. Dean Hovey came back, "I've got some good and some bad news. Good news: we've got a new project with Apple. Bad news: I told Steve we'd design a mouse for 10 bucks."

3. Year later... we... filed... and were granted a patent, on the electro-mechanical-optical mouse of today;... we ended up... [making] the mouse as invisible to people as it is today.

Steve Sachs interview on first computer with GUI: Apple Lisa (~$10K in 1982).
http://library.stanford.edu/mac/primary/interviews/sachs/trans.html

Explanation of parsing method

```java
public boolean parseE() {
    if (first token is an integer) remove it from input and return true;
    if (first token is not '(') return false else remove it from input;
    if (!parseE()) return false;
    if (first token is not '+') return false else remove it from input;
    if (!parseE()) return false;
    if (first token is not ')') return false else remove it from input;
    return true;
}
```

input: (72 + (6+2)) + 2
true

```java
public boolean parseE() {
    if (first token is an integer) remove it from input and return true;
    if (first token is not '(') return false else remove it from input;
    if (!parseE()) return false;
    if (first token is not '+') return false else remove it from input;
    if (!parseE()) return false;
    if (first token is not ')') return false else remove it from input;
    return true;
}
```

```java
public boolean parseE() {
    if (first token is an integer) remove it from input and return true;
    if (first token is not '(') return false else remove it from input;
    if (!parseE()) return false;
    if (first token is not '+') return false else remove it from input;
    if (!parseE()) return false;
    if (first token is not ')') return false else remove it from input;
    return true;
}
```

1. Provides a friendly interface between user and program
2. Allows event-driven or reactive programming: The program reacts to events such as button clicks, mouse movement, keyboard input
3. Often is multi-threaded: Different threads of execution can be going on simultaneously

We use Java's two packages for doing GUIs:
- **AWT (Abstract or Awful Window Toolkit)** —first one
- **Swing** —a newer one, which builds on AWT as much as possible

Two aspects to making a GUI:
1. Placing components (buttons, text, etc.) in it. TODAY
2. Listening/responding to events Next Lecture
Class JFrame

A JFrame object is associated with a window on your monitor. Generally, a GUI is a JFrame object with various components placed in it.

```
Class JFrame is in package javax.swing
```

Some methods in a JFrame object:
- `hide()`: hide the window.
- `show()`: show the window.
- `setVisible(boolean)`: set the visibility of the window.
- `getX()`, `getY()`: get the coordinates of the top-left point.
- `getWidth()`, `getHeight()`: get the width and height of the window.
- `setTitle(String)` and `getTitle()`: set or get the title of the window.
- `getLocation()` and `setLocation(int, int)`: get or set the location of the window.

Over 100 methods in a JFrame object!

### Placing components in a JFrame

**Layout manager**: Instance controls placement of components. **JFrame layout manager default**: BorderLayout.

```
public class C extends JFrame {
    public C() {
        Container cp = getContentPane();
        JButton jb = new JButton("Click here");
        JLabel jl = new JLabel("label 2");
        cp.add(jb, BorderLayout.EAST);
        cp.add(jl, BorderLayout.WEST);
        pack();
        setVisible(true);
    }
}
```

### Packages -- Components

Packages that contain classes that deal with GUIs:
- `java.awt`: Old package.
- `javax.swing`: New package.

- **javax.swing** has a better way of listening to buttons, text fields, etc. Components are more flexible.

- **Component**: Something that can be placed in a GUI window. These are the basic ones used in GUIs.

  **Note**: The use of subclasses to provide structure and efficiency. For example, there are two kinds of `JToggleButton`, so that class has two subclasses.

- **Component** that can contain other components:

  - **Box**
  - **Container**
  - **JComponent**
  - **JComponent**
  - **JPanel**
  - **Applet**
  - **Window**
  - **Frame**
  - **JFrame**
  - **JWindow**

  - **java.awt** is the old GUI package.
  - **javax.swing** is the new GUI package.

  When they wanted to use an old name, they put J in front of it. (e.g. Frame and JFrame)

  When constructing javax.swing, the attempt was made to rely on the old package as much as possible. So, JFrame is a subclass of Frame. But they couldn’t do this with JPanel.
import java.awt.*; import javax.swing.*;
/** Instance has labels in east/west, JPanel with four buttons in center. */
public class PanelDemo extends JFrame {
    JPanel p = new JPanel();
    /** Constructor: a frame with title "Panel demo", labels in east/west,
     * blank label in south, Panel of 4 buttons in the center */
    public PanelDemo() {
        super("Panel demo");
        p.add(new JButton("0")); p.add(new JButton("1"));
        p.add(new JButton("2")); p.add(new JButton("3"));
        Container cp = getContentPane();
        cp.add(new JLabel("east"), BorderLayout.EAST);
        cp.add(new JLabel("west"), BorderLayout.WEST);
        cp.add(new JLabel(""), BorderLayout.SOUTH);
        cp.add(p, BorderLayout.CENTER);
        pack();
    }
}

Class Box: a container
BoxLayout layout manager: Place any number of components.
They appear in the order added, taking only one row.

Simulate BoxLayout Manager in a JFrame
To simulate using a BoxLayout manager for a JFrame, create a Box and
place it as the sole component of the JFrame:
JFrame jf = new JFrame("title");
Box b = new Box(BoxLayout.X_AXIS);
Add components to b;
jf.add(b, BorderLayout.CENTER);

1. Start developing a GUI by changing an already existing one. A lot
of details. Hard to get all details right when one starts from scratch and
has little idea about the Java GUI package.

2. Showed how to place components in a GUI. Next time: how to
"listen" to things like button clicks in a GUI.