Slide 1

Administrivia

- Homework 2 graded. I should be able to return Quiz 3 and written problems for Homework 3 by Friday. Homework 4 may take a bit longer, but I should be able to return them the Monday after the break at least. *If you will need them earlier to start preparing for the exam, I can make a sample solution available Friday. Ask by e-mail.*
- Example from last time available under "sample programs" on course Web site.

Slide 2

Minute Essay From Last Lecture

- (Review question and my answer.)
- Most students seem to be getting why it might be a good idea to have and follow a convention. (I'm guessing other classes stress the value of code reuse and the need in many situations to work on multi-person projects?)
- Saving/restoring registers . . . I'm guessing that many of you learned from the Homework 4 programming problem how this matters for recursive procedures? but in gneral, here too having a convention helps with larger-scale programs.

Slide 3

Pseudoinstructions, One More Thing

- Some of you discovered that SPIM supports a pseudoinstrcton form of `lw` that works more like you might like for accessing array elements, e.g.

  ```
  lw  A($t0)
  ```

  where `A` is a label in the program's data segment and `$t0` contains a byte offset.
- This is admittedly convenient for referencing elements of arrays stored in the data segment, but think about whether it would work for the more typical-in-real-code situations of an array as a local variable in some procedure, or a dynamically-allocated array. So I think it's better, in this class, not to make use of this convenience.

Slide 4

Multiplication and Division — Recap/Review

- The "real" instructions for these two operations store results not in one of the general-purpose registers, but a pair of special-purpose registers `lo` and `hi`. To access, use `mflo` and `mfhi` to copy values to a general-purpose register, as in the example last time.
- `mul` computes a 64-bit result, low 32 bits in `lo` and high 32 bits in `hi`.
- `div` computes both quotient (in `lo`) and remainder (in `hi`).

Slide 5

One More Example

- Another possibly interesting procedure would be one like the program I show in CSCI 1120 to divide, with two pointer parameters to allow "returning" both quotient and remainder.
- Let's try this as an in-class "practice problem": Starter code under "sample programs". You try it, individually or in pairs, and we'll share results before quiz time. When you have something:

```
chmod go+r pgm.s
cp -p pgm.s /users/bmassint/TEMP2321/yourusername
```

Slide 6

Minute Essay

- None — quiz.
- (Best wishes for a good spring break!)