Class Announcements

• Third homework is due today. Will post sample solution tomorrow morning

• Midterm 1: Friday, February 24, in class, closed book and notes
  – Please be early so we can start on time. Exam is 70 minutes.
  – No bathroom breaks
  – No smart phones or smart watches
  – Bring your RU ID - you will need to sign out when turning in your exam
  – If you have a Letter of Accommodations, arrange exam location with Disability Services

• Office hours in CoRE 305 before exam:
  – Wednesday, 10:00am - noon, Zining Fan
  – Wednesday, 1:45pm - 3:15pm, Uli Kremer (no office hours on Friday)
  – Thursday, 1:00pm - 3:00pm, Chun Lau
Example program:

```
a=3;b=5;c=3*ab;d=+c1;#d.
```
Project 1: Structure Overview

compiler

```
+----------------+         
| tinyL program  |         
+----------------+         
|                 |         
|                 |         
| compile         |         
+----------------+         
|                 |         
|                 |         
|                 |         
| tinyL.out       |         
+----------------+         
```

example: test1

example: compile test1

output always "tinyL.out"

optimizer

```
+----------------+         
| RISC machine code |         
+----------------+         
|                 |         
|                 |         
| optimize        |         
+----------------+         
|                 |         
|                 |         
| RISC machine code |         
+----------------+         
```

example: tinyL.out

example: optimize < tinyL.out

output to stdout

virtual machine

```
+----------------+         
| RISC machine code |         
+----------------+         
|                 |         
|                 |         
| sim             |         
+----------------+         
```

input and output of execution

example: tinyL.out
Project 1: Peephole Optimizations and Dead Code Elimination

Goal: To replace a sequence of instructions by another, more efficient sequence.

Peephole Optimizations - window size is 3 instructions or less.

Constant folding example:

\[
\begin{align*}
\text{loadI} & \ 5 \Rightarrow r4 \\
\text{loadI} & \ 7 \Rightarrow r5 \quad \text{can be replaced by} \quad \text{loadI} \ 12 \Rightarrow r6 \\
\text{add} \ & \ r4, \ r5 \Rightarrow r6
\end{align*}
\]

Partial evaluation example:

\[
\begin{align*}
\text{loadI} & \ 1 \Rightarrow r7 \\
\text{mult} \ & \ r6, \ r7 \Rightarrow r8 \quad \text{can be replaced by} \quad \text{i2i} \ r6 \Rightarrow r8
\end{align*}
\]

Strength reduction example:

\[
\begin{align*}
\text{loadI} & \ 4 \Rightarrow r7 \\
\text{mult} \ & \ r6, \ r7 \Rightarrow r8 \quad \text{can be replaced by} \quad \text{lshiftI} \ r6, \ 2 \Rightarrow r8
\end{align*}
\]

Dead Code Elimination

\[
\begin{align*}
\text{loadI} & \ 5 \Rightarrow r4 \\
\text{loadI} & \ 7 \Rightarrow r5 \quad \text{can be replaced by} \quad \text{loadI} \ 12 \Rightarrow r6 \\
\text{loadI} & \ 12 \Rightarrow r6
\end{align*}
\]

Replacement only valid if r4 and r5 are no longer used in the program. Note: Static single assignment code shape
**Imperative Programming Languages**

**Imperative:**
Sequence of state-changing actions.

- Manipulate an abstract machine with:
  1. Variables naming memory locations
  2. Arithmetic and logical operations
  3. Reference, evaluate, assign operations
  4. Explicit control flow statements
- Key operations: *Assignment* and “*Goto*”
- Fits the von Neumann architecture closely

---

The given bus widths are examples only!

**Von Neumann Architecture**
Next Lecture

Things to do:
First project will be posted after the first midterm exam.

Read Scott: Chap. 3.1 - 3.4; 8.3 ; ALSU Chap. 7.1 - 7.3

Next time:

- pointers and dynamic memory allocation
- Dynamic vs. static scoping
- Runtime environment
- access links and control links management