Plan for Today

MiniJava compiler
- source: MiniJava
- target: MIPS

MIPS basics

Stack frame the MiniJava compiler will generate
- Calling convention is an agreement amongst programmers, procedure call convention
- Using Patt and Patel calling convention

Patt & Pattel Book Stackframe for MIPS (example)

```c
int foo(int x, int y, int *z) {
    int a;
    a = x * y - *z;
    return a;
}
void main() {
    int x;
    x = 2;
    printf("%d\n", foo(4, 5, &x));
}
```

```
.text
foo:
### prologue
addi $sp, $sp, -4  # retval space
addi $sp, $sp, -4  # push ra
sw $ra, 0($sp)     #
addi $sp, $sp, -4  # push fp
sw $fp, 0($sp)     #
addi $sp, $sp, -4  # set up new fp
addi $sp, $sp, -4  # local a
...
sw $t0, 12($fp)    # return result
addi $sp, $sp, 4   # dealloc
lw $fp, 0($sp)     # pop fp
addi $sp, $sp, 4   #
addi $ra, 0($fp)    # pop ra
addi $sp, $sp, 4   #
jr $ra
```

```
.text
main:
addi $fp, $sp, -4
addi $sp, $sp, -4  # x = 2
li $t0, 2
sw $t0, 0($fp)
# push &x
addi $sp, $sp, -4
sw $fp, 0($fp)
# push 5
li $t0, 5
addi $sp, $sp, -4
sw $t0, 0($sp)
# push 4
li $t0, 4
addi $sp, $sp, -4
sw $t0, 0($sp)
jal _foo
# grab retval
lw $t0, 0($sp)
# pop retval & params
addi $sp, $sp, 16  # print $t0
...
# HALT MARS
li $v0, 10
syscall
```

```
```
```
Patt and Patel book calling convention (for MIPS)

**Calling convention (contract between caller and callee)**

- caller should push parameters right to left onto the stack
- upon callee entry, the stack pointer $sp should be pointing at the first parameter
- upon callee exit, the stack pointer $sp should be pointing at the return value, which should be followed by the first parameter
- $sp must be divisible by 4 (for MIPS)
- $sp should always be pointing at the top entry on the stack

**Standardizing the stack frame implementation for this course**

- $ra and $fp should be stored on top of the return value slot
- locals should be stored on top of $ra and $fp
- $fp should be made to point at the first local variable, so that the address for the first local is $fp+0, the address for the second local is $fp+4, ...
- The offsets for the incoming parameters will differ based on whether there is a return value. If there is a return value, then the first parameter will be at $fp-16, the second at $fp-20, etc. If there is no return value, then the first parameter will be at $fp-12, the second at $fp-16, etc.

Another example: where does each variable go?

```java
class A {
    public static void main(String[] a){
        System.out.println(
            new B().foo(true,7,new B(),new int[3]);
        )
    }
}
class B {
    int [] x;
    boolean mBool;

    public int foo(boolean p1, int p2, B b, int [] y) {
        boolean v1; int i; int j; return 0;
    }
    public B bar() {
        B b;
        b = new B();
        return b;
    }
    public boolean baz() {
        return mBool;
    }
}
```