**Compiler Construction**
**Professor Jeremy Siek**

**Quiz 3**
**September 27, 2010**

**Name:** _____

This exam has 4 questions, for a total of 10 points.

1. [2 points] Give the set of live variables between each statement in the following program.

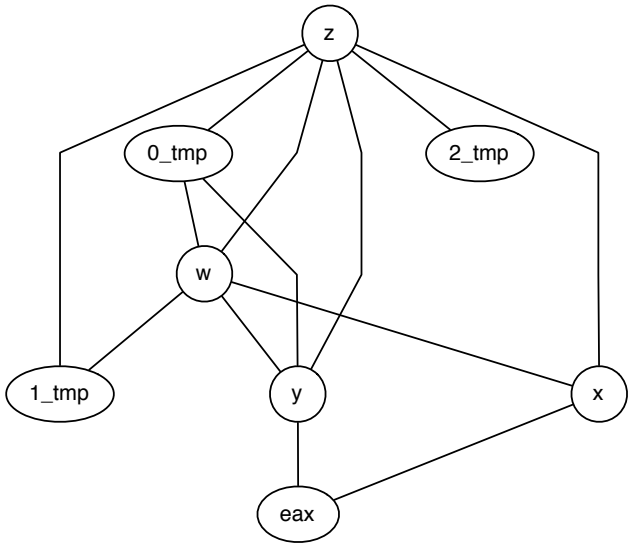
```
subl $16, %esp
call input
addl $16, %esp
movl %eax, x
movl x, y
subl $16, %esp
call input
addl $16, %esp
movl %eax, z
movl z, w
movl w, z
movl y, w
movl x, 0_tmp
addl y, 0_tmp
movl 0_tmp, 1_tmp
addl w, 1_tmp
movl 1_tmp, 2_tmp
addl z, 2_tmp
push 2_tmp
call print_int_nl
addl $16, %esp
```

**Solution:**

```
subl $16, %esp
 { }
call input
 { }
addl $16, %esp
 { }
movl %eax, x
 { x }
movl x, y
 { x, y }
subl $16, %esp
 { x, y }
call input
 { x, y }
```

```
addl $16, %esp
 { x, y }
movl %eax, z
 { x, y, z }
movl z, w
 { w, x, y }
movl w, z
 { x, y, z }
movl y, w
 { w, x, y, z }
movl x, 0_tmp
 { w, y, z, 0_tmp }
addl y, 0_tmp
 { w, z, 0_tmp }
movl 0_tmp, 1_tmp
 { w, z, 1_tmp }
addl w, 1_tmp
 { z, 1_tmp }
movl 1_tmp, 2_tmp
 { z, 2_tmp }
addl z, 2_tmp
 { 2_tmp }
push 2_tmp
call print_int_nl
addl $16, %esp
```

2. [2 points] Draw the interference graph for the program in question 1.




**Solution:**

3. [4 points] Perform one iteration of register allocation for the program from question 1 using the greedy saturation-based algorithm. You are compiling for a computer architecture that has only two general purpose registers, %eax and %ebx, but is otherwise just like the x86 (it does have %esp and %ebp). Write down the following information for each step in the algorithm: 1) how many available registers there are for each variable at the beginning of the step, 2) what variable is chosen for assignment in this step, and 3) which register or stack location is assigned to that variable.

**Solution:** The following is one of many possible solutions.

| w | x | y | z | 0_tmp | 1_tmp | 2_tmp | chosen variable | chosen color |
|---|---|---|---|---|---|---|---|---|
| 2 | 1 | 1 | 2 | 2 | 2 | 2 | x | %ebx |
| 1 | - | 1 | 1 | 2 | 2 | 2 | w | %eax |
| - | - | 1 | 0 | 1 | 1 | 2 | z | -4(%ebp) |
| - | - | 1 | - | 1 | 1 | 2 | y | %ebx |
| - | - | - | - | 0 | 1 | 2 | 0_tmp | -8(%ebp) |
| - | - | - | - | - | 1 | 2 | 1_tmp | %ebx |
| - | - | - | - | - | - | 2 | 2_tmp | %eax |

4. [2 points] Do you need to insert any extra spill code given the results of the register allocation? If so, what needs to be inserted and where?

**Solution:** The variables z and 0_tmp do not appear together in an instruction, so no spill code is necessary.