Instructions

1. The is a closed-book exam, but you are allowed to use a single page (both sides) of notes.
2. There are 6 problems in this exam and you have 75 minutes to answer them.
3. To receive full credit, your solution must not only be correct but also show all the steps.
4. Discussing the exam contents with anyone who has not taken the exam is a violation of the academic honesty code.

Problem 1. (10 points) Consider the following j-- program:

```java
public class Mystery {
    private int f = 42;

    public static int f(int u, long v, int w) {
        int x = u * w;
        {Mystery m = new Mystery();
         long y = m.g(u, v);
         long z = (long)x * y;
         }
        {Mystery m = new Mystery();
         int y = u * w;
         long z = m.g(w, v);
        }
        return x;
    }

    public long g(int u, long v) {
        int u = u * u;
        long x = (long)f + (long)v;
        long y = x * x;
        return y;
    }
}
```

a. (5 points) Show in detail the compilation unit context for Mystery, as built by `preAnalyze()`.

b. (5 points) Show in detail the method/local contexts for `f()` and `g()`.

Problem 2. (10 points) Classify the following casts as identity, narrowing, widening, boxing, or unboxing. What instructions (if any) must generated for each?

a. (1.25 points)

```
String x = (String)(new Object());
```

b. (1.25 points)

```
char x = (char)42;
```
c. (1.25 points)

\[ \text{char } x = \text{(char)} \text{(new Character('*'))}; \]

d. (1.25 points)

\[ \text{Object } x = \text{(Object)} \text{(new String("Hello, World"))}; \]

e. (1.25 points)

\[ \text{Boolean } x = \text{(Boolean)} \text{false}; \]

f. (1.25 points)

\[ \text{String } x = \text{(String)} \text{(new String("Hello, World"))}; \]

g. (1.25 points)

\[ \text{int } x = \text{(int)} '0'; \]

h. (1.25 points)

\[ \text{char } x = \text{(char)} '0'; \]

**Problem 3.** (20 points) Suppose o is an object and f an instance integer field within, a is an array of integers, and y, i and z are integers. For each of the j-- statements below, using the following table, list the instructions generated and show how the runtime stack evolves as those instructions are executed.

<table>
<thead>
<tr>
<th>Instruction</th>
<th>x</th>
<th>a[i]</th>
<th>s.f</th>
<th>C.sf</th>
</tr>
</thead>
<tbody>
<tr>
<td>lhs = y</td>
<td>iload y'</td>
<td>aload a'</td>
<td>aload o'</td>
<td>iload y'</td>
</tr>
<tr>
<td></td>
<td>[dup]</td>
<td>iload 1'</td>
<td>[dup]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>intore x'</td>
<td>iload y'</td>
<td>iload y'</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>[dup_x2]</td>
<td>putstatic sf</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>iastore</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lhs += y</td>
<td>iload x'</td>
<td>aload a'</td>
<td>aload o'</td>
<td>getstatic sf</td>
</tr>
<tr>
<td></td>
<td>iload y'</td>
<td>iload 1'</td>
<td>dup</td>
<td>iload y'</td>
</tr>
<tr>
<td></td>
<td>[dup]</td>
<td>iaload</td>
<td>getfield f</td>
<td>iadd</td>
</tr>
<tr>
<td></td>
<td>intore x'</td>
<td>iload y'</td>
<td>iadd</td>
<td>putstatic sf</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[dup_x2]</td>
<td>iadd</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>[dup_x2]</td>
<td>putstatic sf</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>iastore</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+=lhs</td>
<td>iinc x',1</td>
<td>aload a'</td>
<td>aload o'</td>
<td>getstatic sf</td>
</tr>
<tr>
<td></td>
<td>[iload x']</td>
<td>iload i'</td>
<td>dup</td>
<td>iconst_1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dup2</td>
<td>getfield f</td>
<td>iadd</td>
</tr>
<tr>
<td></td>
<td></td>
<td>iaload</td>
<td>iadd</td>
<td>[dup]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>iadd</td>
<td>iadd</td>
<td>putstatic sf</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[dup_x2]</td>
<td>putstatic sf</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>iastore</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lhs--</td>
<td>[iload x']</td>
<td>aload a'</td>
<td>aload o'</td>
<td>getstatic sf</td>
</tr>
<tr>
<td></td>
<td>iinc x',-1</td>
<td>iload i'</td>
<td>dup</td>
<td>[dup]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dup2</td>
<td>getfield f</td>
<td>iconst_1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>iaload</td>
<td>[dup_x2]</td>
<td>isub</td>
</tr>
<tr>
<td></td>
<td></td>
<td>iadd</td>
<td>iconst_1</td>
<td>putstatic sf</td>
</tr>
<tr>
<td></td>
<td></td>
<td>iconst_1</td>
<td>isub</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>isub</td>
<td>putstatic f</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>iastore</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Problem 4. (20 points) Consider the following JVM bytecode for a j- method \( \text{int mystery(int x, int y)} \):

\[
\begin{align*}
\text{public static int mystery(int , int);} \\
0: & \text{iconst}_1 \\
1: & \text{istore}_2 \\
2: & \text{iload}_1 \\
3: & \text{iconst}_0 \\
4: & \text{if_icmple 18} \\
7: & \text{iload}_2 \\
8: & \text{iload}_0 \\
9: & \text{imul} \\
10: & \text{istore}_2 \\
11: & \text{iload}_1 \\
12: & \text{iconst}_1 \\
13: & \text{isub} \\
14: & \text{istore}_1 \\
15: & \text{goto 2} \\
18: & \text{iload}_2 \\
19: & \text{ireturn}
\end{align*}
\]

a. (10 points) Show how the runtime stack evolves as a result of the call \text{mystery}(3, 4).

b. (5 points) What value does the above call return?

c. (5 points) What does \text{mystery}(x, y) compute and return in general?

Problem 5. (20 points) The LIR instructions for the method \( \text{int mystery(int x, int y)} \) from the previous problem are listed below.

\[
\begin{align*}
\text{B0} & \\
\text{B1} & \text{0: LDC [1] [V32|I]} \\
& \text{5: MOVE $a1 [V33|I]} \\
& \text{10: MOVE [V32|I] [V34|I]} \\
\text{B2} & \text{15: LDC [0] [V35|I]} \\
& \text{20: BRANCH [LE] [V33|I] [V35|I] B4} \\
\text{B3} & \text{25: MUL [V34|I] $a0 [V36|I]} \\
& \text{30: LDC [1] [V37|I]} \\
& \text{35: SUB [V33|I] [V37|I] [V38|I]} \\
& \text{40: MOVE [V36|I] [V34|I]} \\
& \text{45: MOVE [V38|I] [V33|I]} \\
& \text{50: BRANCH B2} \\
\text{B4} & \text{55: MOVE [V34|I] $v0} \\
& \text{60: RETURN $v0}
\end{align*}
\]

a. (10 points) Compute the liveUse and liveDef sets for each basic block in the method.
b. (10 points) Compute the liveIn and liveOut sets for each basic block in the method.

Problem 6. (20 points) The liveness intervals for the registers (all except $V_{33}$ and $V_{34}$) in the LIR instructions from the previous problem are listed below.

<table>
<thead>
<tr>
<th>Register</th>
<th>Liveness Intervals</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_0$</td>
<td>[55, 60]</td>
</tr>
<tr>
<td>$a_0$</td>
<td>[0, 25] [25, 50]</td>
</tr>
<tr>
<td>$a_1$</td>
<td>[0, 5]</td>
</tr>
<tr>
<td>$V_{32}$</td>
<td>[0, 10]</td>
</tr>
<tr>
<td>$V_{33}$</td>
<td>??</td>
</tr>
<tr>
<td>$V_{34}$</td>
<td>??</td>
</tr>
<tr>
<td>$V_{35}$</td>
<td>[15, 20]</td>
</tr>
<tr>
<td>$V_{36}$</td>
<td>[25, 40]</td>
</tr>
<tr>
<td>$V_{37}$</td>
<td>[30, 35]</td>
</tr>
<tr>
<td>$V_{38}$</td>
<td>[35, 45]</td>
</tr>
</tbody>
</table>

a. (10 points) Compute the liveness intervals for $V_{33}$ and $V_{34}$.

b. (5 points) List the neighbors of the 7 vertices ($V_{32} — V_{38}$) in the interference graph $G$ for the method, in ascending order.

c. (5 points) Color the graph $G$ using 3 physical registers and draw the graph using symbols ▲, ★, and ■ for vertices, where the symbols denote the three registers.
1 a) `java.lang.String` \[\xrightarrow{\text{clashup}}\] `String` class

`java.lang.Object` \[\xrightarrow{\text{clashup}}\] `Object` class

`Mystery` \[\xrightarrow{\text{clashup}}\] `Mystery` class
- private int f;
- public static void f(int, long, int)
- public long g(int, long)

b) `f()`

\[
\begin{array}{c|c}
\text{u} & 0 \\
\text{V} & 1 \\
\text{W} & 3 \\
\text{LC} & x - 4 \\
\text{M} & 5 \\
\text{LC} & y - 6 \\
\text{z} & 8 \\
\text{M} & 5 \\
\text{LC} & y - 6 \\
\text{z} & 7 \\
\end{array}
\]

`g()`

\[
\begin{array}{c|c}
\text{this} & 0 \\
\text{MC} & u - 1 \\
\text{V} & 2 \\
\text{W} & 4 \\
\text{LC} & x - 5 \\
\text{Y} & 7 \\
\end{array}
\]

MC \Rightarrow Method Context

LC \Rightarrow Local Context
(2) a) Narrowing reference
   CHECKCAST java/lang/String (fails at runtime)

b) Narrowing primitive
   I2C

c) Unboxing
   INVOKEVIRTUAL Character charValue (java/lang/Character;)C

d) Widening reference
   No code

e) Boxing
   INVOKESTATIC Boolean valueOf (Z) java/lang/Boolean;

f) Identity
   No code

g) Widening primitive
   No code

h) Identity
   No code
(3) a) \( f = 0 \)
\[
\begin{align*}
&0 \quad | \quad 0 \\
&0 \quad | \quad 0 \\
&0 \quad | \quad 0.0 \\
&0 \quad | \quad 0.0 + 1 \\
&0 \quad | \quad 0.0 + 1 - 1 \\
\end{align*}
\]
\[
\begin{align*}
&\text{aload 0} \\
&\text{dup} \\
&\text{get field f} \\
&\text{iconst -1} \\
&\text{sub} \\
&\text{put field f} \\
\end{align*}
\]

b) \( z = ++a[i] \)
\[
\begin{align*}
&1a \\
&1a | i \\
&1a | i | a[i] \\
&1a | i | a[i] | 1 \\
&1a | i | a[i] | 1 + 1 \\
&1a | i | a[i] | 1 + 1 \\
\end{align*}
\]
\[
\begin{align*}
&\text{aload a} \\
&\text{iload i} \\
&\text{dup2} \\
&\text{inload} \\
&\text{iconst -1} \\
&\text{load} \\
&\text{dup -x2} \\
&\text{restore} \\
\end{align*}
\]

(4) a)
\[
\begin{array}{cccccc}
0 & 1 & 2 & 3 & 4 & 5 \\
3 & 4 & 5 & 2 & 1 & 0 \\
2 & 3 & 8 & 1 & 9 & 0 \\
\end{array}
\]

b) 81

c) \( x^y \)
a)  
\[B_0\]
\[\downarrow\]
\[B_1\]
\[\downarrow\]
\[B_2\] \[V_{33}\] \[\downarrow\]
\[B_3\]
\[\downarrow\]
\[B_4\] \[V_{34}\]  

b)  
\[B_0\]
\[\downarrow\]
\[B_1\] \[\$a_0, \$a_1\]  
\[\downarrow\]
\[B_2\]\[\$a_0, V_{33}, V_{34}\]  
\[\downarrow\]
\[B_3\]\[\$a_0, V_{33}, V_{34}\]  
\[\downarrow\]
\[B_4\]\[V_{34}\]  

\boxed{\text{Line In}}  
\boxed{\text{Line Out}}
a) $V_{33}$

$V_{34}$

b) $V_{32}$ — $V_{33}$

$V_{33}$ — $V_{32}$, $V_{34}$, $V_{35}$, $V_{36}$, $V_{37}$

$V_{34}$ — $V_{33}$, $V_{35}$, $V_{38}$

$V_{35}$ — $V_{33}$, $V_{34}$

$V_{36}$ — $V_{33}$, $V_{34}$, $V_{38}$

$V_{37}$ — $V_{33}$, $V_{36}$

$V_{38}$ — $V_{34}$, $V_{36}$

c) Allocation

$V_{32}$ — $\Gamma_1$

$V_{33}$ — $\Gamma_2$

$V_{34}$ — $\Gamma_3$

$V_{35}$ — $\Gamma_4$

$V_{36}$ — $\Gamma_4$

$V_{37}$ — $\Gamma_1$

$V_{38}$ — $\Gamma_1$