This is an open-book, open-notes, open-laptop, closed-network exam. Please do not post anything on Piazza until the solutions are put up on the class website.

You have 1 hour 20 minutes to work on this exam. The examination has 5 problems worth 70 points. Please budget your time accordingly. Write your answers in the space provided on the exam. If you use additional scratch paper, please turn that in as well.

Your Name: ________________________ SUNet ID: ________________________

The following is a statement of the Stanford University Honor Code:

1. The Honor Code is an undertaking of the students, individually and collectively:

   (a) that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;

   (b) that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.

2. The faculty on its part manifests its confidence in the honor of its students by refraining from proctoring examinations and from taking unusual and unreasonable precautions to prevent the forms of dishonesty mentioned above. The faculty will also avoid, as far as practicable, academic procedures that create temptations to violate the Honor Code.

3. While the faculty alone has the right and obligation to set academic requirements, the students and faculty will work together to establish optimal conditions for honorable academic work.

Signature: ________________________

<table>
<thead>
<tr>
<th>Problem</th>
<th>#1</th>
<th>#2</th>
<th>#3</th>
<th>#4</th>
<th>#5</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Score</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>15</td>
<td>10</td>
<td>10</td>
<td>15</td>
<td>20</td>
<td>70</td>
</tr>
</tbody>
</table>
Problem 1. Short questions. For true or false questions, you must justify your answers in 1 to 2 sentences [15 Points].

1. True or false. If a program does not have any program point that has more than \( n \) overlapping live ranges, the corresponding interference graph must be \( n \)-colorable.

False. This can be proven with a counter-example, such as the control flow graph in problem 2 of HW3. In that case, \( n \) is 2, but the interference graph is 3-chromatic.

2. True or false. For the data-flow analysis to find dominators as discussed in class, if we initialize all the interior points to \( \emptyset \), the iterative algorithm will converge to the MFP solution.

False. Consider the following CFG. B1 dominates B2 but the analysis will not include B1 as B2’s dominator if we initiate the interior points to \( \perp \), which is the empty set.

![CFG Diagram]

3. True or false. Applying constant propagation to a program can create more opportunities for partial redundancy elimination.

True, consider the following CFG.

![CFG Diagram]

4. True or false. Suppose we perform dead-code elimination on a program, followed by partial redundancy elimination. Performing dead-code elimination again might further optimize the program. False. PRE introduces expressions to where they are anticipated. Therefore, it cannot introduce dead code.

5. For liveness analysis, what is the preferred order for an iterative data-flow algorithm to visit the basic blocks? Post order, as it is a backwards analysis.
Problem 2. Register Allocation [10 points]

Given the following control flow graph, rename the defined and used variables with names that identify their live ranges (e.g. A becomes A1):

Solution:

Draw the register interference graph for the live ranges:

Solution:

NOTE: As C1 is used in the final block, even though it is not defined/used on the left path we also accepted answers that showed C1 conflicting with A2 and thus needing three registers to color the graph.

What is the minimum number of registers needed to color the graph? Two registers.
Problem 3. Dominators [10 points]

Consider the following flow graph.

1. Draw the dominator tree for the graph.

See graph below.

2. Are there any back edges in the graph? If so, list them and the corresponding natural loops. There are no back edges in the graph.
Problem 4. Partial Redundancy Elimination [15 points]

Show the result of running partial redundancy elimination. What’s the final optimized flow graph? You don’t need to show the intermediate steps.

Solution:
Problem 5. Data-flow Analysis [20 points]

Given a program that writes to files, we wish to remove unnecessary opening and closing of files. An open/close instruction of a file is unnecessary if that file is never written to. You can assume that no file is open at the beginning of the program. You do not have to warn of any errors, such as if a file is opened twice or a file is never closed.

A program can have an arbitrary number of file variables, denoted as \( f \) below. The relevant instruction set consists of the following:

\(- f = \text{open}(\ldots); \)
\(- f.\text{write}(\ldots); \)
\(- f.\text{close}(\ldots); \)

Describe your data-flow analysis by specifying the following:

- Direction of your data-flow analysis (forward/backward)
- Elements in the semi-lattice and their meaning
- Meet operator or lattice diagram
- Top and bottom elements if they exist
- Transfer function
- Boundary condition initialization
- Interior points initialization

You may design one or more analyses. Please fill out the above information for each analysis you create.

Make sure you specify how you remove the unnecessary open/close instructions from the program.

Write your answer on the next page.
Write your data-flow analysis answer here:

**DFA1:**

Direction: Backward

Elements: set of files that has been written to and have not seen an open

Meet operator: $\cup$

Top: $\emptyset$

Bottom: Universal set

Transfer function:

- $f[b] = \text{In}[b] - \{f\}$ if $b$ is $f = \text{close()}$
- $f[b] = \text{In}[b] \cup \{f\}$ if $b$ is $f = \text{write()}$
- $f[b] = \text{In}[b] - \{f\}$ if $b$ is $f = \text{open()}$

Boundary: $\text{In}[\text{Exit}] = \emptyset$

Interior point initialization: $\emptyset$

**DFA2:**

Direction: Forward

Elements: set of files that has been written to and have not seen a close

Meet operator: $\cup$

Top: $\emptyset$

Bottom: Universal set

Transfer function:

- $f[b] = \text{In}[b] - \{f\}$ if $b$ is $f = \text{close()}$
- $f[b] = \text{In}[b] \cup \{f\}$ if $b$ is $f = \text{write()}$
- $f[b] = \text{In}[b] - \{f\}$ if $b$ is $f = \text{open()}$

Boundary: $\text{Out}[\text{Entry}] = \emptyset$

Interior point initialization: $\emptyset$

For each basic block $B = f.\text{open()}$, if DFA1.out[$B$] does not contain $f$, remove $B$; For each basic block $B = f.\text{close()}$, if DFA2.in[$B$] does not contain $f$, remove $B$. 