1. **(6 points)** Consider the language $L_i$ defined by the expression $a^n b^m c^n$, $n, m \geq 0$. (i.e., any string where the number of 'a's is equal to the number of 'b' s and 'c's combined). Can a finite state machine recognize $L_i$? Give a one-sentence argument for why or why not.

2. **(6 points)** Express $L_i$ from Question 1, above, using a grammar with $S$ as the start symbol. Tell what the terminals and non-terminals for the grammar are. *Note: this can be done with a grammar with four productions, but points will not be deducted if your grammar has eight or fewer productions.*

3. Use the LR(1) machine at the bottom of the page to answer parts a–c. When specifying the parser action, give the action and the production recognized if a reduction.

   a. **(6 points)** Let the symbol stack be “$x \ y\ x$” (where the right-most symbol is the top of the stack), the state stack is 1 5 8 5, and the next symbol is $x$. What is the next parser action, and the state of the symbol and state stack after that parser action?

   b. **(6 points)** Let the symbol stack be “$x\ y\ x$” (where the right-most symbol is the top of the stack), the state stack is 1 5 8 5, but assume the next symbol is $z$. What is the next parser action, and the state of the symbol and state stack after that parser action?

   c. **(6 points)** Suppose the state stack is 1 5 6 7, and the next symbol is $x$. What is the next parser action?
4. This question pertains to liveness analysis. For each statement below, we want to calculate which variables are live coming into the statement, and which variables are live going out of the statement. Assume no variables are live at the end of this code.

   a. **(16 points) In your blue books**, fill in a table like the one below, with one row for each statement in the program. **Do not fill in the table below.** Assume there is no aliasing.

   b. **(4 points) Show the GEN and KILL sets for statement 5 if variable b may be aliased to variable c.**

<table>
<thead>
<tr>
<th>Statement</th>
<th>GEN</th>
<th>KILL</th>
<th>IN</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. `a = 8;`

2. `b = a + c;`

3. If `(b < d)` goto 7

4. `d = 8 + c;`

5. `b = b - a;`

6. goto 3

7. `a = 10;`

8. `exit`
5. Given the loop nest:

```c
for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++) {
        c[j, i] = b[i + 1, j + 3];
        b[i + 3, j] = c[j + 2, i];
    }
}
```

a. **(5 points)** Describe the dependences, if any, on the “c” array in the loop nest above. Either say “no dependence”, or, if one or more dependences exists, give the type(s) (flow or true, output or anti), the direction and the distance.

b. **(5 points)** Describe the dependences, if any, on the “b” array in the loop nest above. Either say “no dependence”, or, if one or more dependences exists, give the type(s) (flow or true, output or anti), the direction and the distance.

c. **(5 points)** Can the loops be interchanged? Why or why not?

d. **(5 points)** Can loop distribution be applied to the inner loop? Why or why not?

6. The latest Acme computers have half as many registers as their old machines. Are the following transformations more or less effective on the new machine? Explain your answer in 40 words or less.

a. **(5 points)** Loop unrolling

b. **(5 points)** Register allocation

c. **(5 points)** Common subexpression elimination

d. **(5 points)** Instruction scheduling

7. **(10 points)** For the following code, show what the code would look like after performing strength reduction.

```c
1: X = 10;
2: Y = A;
3: if (Y < A + 10) goto 9
4: R = 4*X + 7;
5: X = X + 10;
6: S = 3*Y + 10;
7: Y = Y - 1;
8: goto 3
9: exit;
```