1.) Consider the following problem:

**PROB**

**INSTANCE:** A program II such that II takes a string as input, and outputs a string. It is guaranteed that II terminates on any input string.

**QUESTION:** Do there exist strings $I_1, I_2$ such that $\Pi(I_1) = I_2$, i.e., such that the output of $\Pi$ on the input $I_1$ is equal to $I_2$?

Prove that the problem **PROB** is semi-decidable. For this, describe a procedure that shows the semi-decidability of the problem (i.e. a semi-decision procedure for **PROB**) and argue that it is correct.

Note: we consider only strings that are built from symbols 0 and 1.  

(15 points)

2.) (a) Given the following first-order logic formula $\psi$:

$$\psi: \left[p(f(x, y), u) \land p(x, z)\right] \rightarrow p(f(y, z), u)$$

where $f/2$ is a binary function symbol and $p/2$ is a binary predicate symbol. Let $T$ be a theory which forces $p/2$ to be reflexive, symmetric, and transitive. Additionally, $T$ includes the following axiom related to $p$ and $f$:

$$\forall x_1, x_2, y_1, y_2 : \left[p(x_1, x_2) \land p(y_1, y_2)\right] \rightarrow p(f(x_1, y_1), f(x_2, y_2))$$

Give a detailed proof that $\psi$ is $T$-valid.  

(9 points)

(b) Consider the clauses $C_1, \ldots, C_5$ in dimacs format (in this order, shown in the box; recall that 0 indicates the end of a clause) which are given as input to a SAT solver. Apply CDCL to solve the CNF using the convention that if a variable is assigned as a decision, then it is assigned 'false'. Further, select variable 3 as the first decision variable that is assigned.

- Each time when a conflict occurs and after backtracking, draw the implication graph and indicate all UIPS and mark the first UIP. For each UIP, indicate the cut (i.e., a set of edges) and its asserting conflict clause. Learn the asserting conflict clause that corresponds to the first UIP.

- Is the given CNF satisfiable, unsatisfiable, or valid? Can the empty clause be derived from the given CNF during CDCL? Justify your answers to the above questions.

(6 points)

3.) Let $\pi$ be the program $x := x - y; y := x + y; x := y - x$.

(a) Specify a correctness assertion stating that this program swaps that values of the variables $x$ and $y$.  

(1 point)

(b) Prove the correctness assertion using weakest preconditions.  

(5 points)

(c) Prove the correctness assertion using strongest postconditions.  

(9 points)
4.) (a) Show that simulation is a transitive relation, i.e. given any 3 Kripke structures 

\[ K_1 = \{S_1, I_1, R_1, L_1\}, \; K_2 = \{S_2, I_2, R_2, L_2\} \text{ and } K_3 = \{S_3, I_3, R_3, L_3\} \]

over atomic predicates \(AP\), such that \(K_1 \leq K_2\) and \(K_2 \leq K_3\), show that \(K_1 \leq K_3\). (5 points)

(b) Consider the following Kripke structure \(M\):

\[\begin{array}{c}
\text{s0: \{c\}} \\
\text{s1: \{a, b, c\}} \\
\text{s2: \{b, c\}} \\
\text{s3: \{c\}} \\
\text{s4: \{b\}}
\end{array}\]

For each of the following formulae \(\varphi\),

i. check the respective box if the formula is in CTL, LTL, and/or CTL*, and

ii. list the states \(s_i\) on which the formula \(\varphi\) holds; i.e. for which states \(s_i\) do we have \(M, s_i \models \varphi\)?

<table>
<thead>
<tr>
<th>(\varphi)</th>
<th>CTL</th>
<th>LTL</th>
<th>CTL*</th>
<th>States (s_i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(G(b))</td>
<td>□</td>
<td>□</td>
<td>□</td>
<td></td>
</tr>
<tr>
<td>(F(a))</td>
<td>□</td>
<td>□</td>
<td>□</td>
<td></td>
</tr>
<tr>
<td>(X(a))</td>
<td>□</td>
<td>□</td>
<td>□</td>
<td></td>
</tr>
<tr>
<td>(A[a U c])</td>
<td>□</td>
<td>□</td>
<td>□</td>
<td></td>
</tr>
<tr>
<td>(EF(a))</td>
<td>□</td>
<td>□</td>
<td>□</td>
<td></td>
</tr>
</tbody>
</table>

(5 points)

(c) The **subset sum problem** is defined as follows: Given a set of \(N\) integers \(S = \{i_1, i_2, \ldots, i_N\}\), does \(S\) have a nonempty subset whose sum is zero?

Write a C program that implements a *guess and check* routine for the subset sum problem and instrument the program with an appropriate CBMC assertion.

You may assume the following template:

```c
int nondet_bool(); // non-deterministically returns 0 or 1

// Fixed sample input:
int N = 8; // size of the set
int values[] = { 4, -8, 15, -16, -23, 42, -11, 13 }; // elements in the set

int main() {
    // add code here:
    // 1. guess a solution
    // 2. put an assertion such that CBMC reports if there is a solution
}
```

(5 points)