1. CS5391 Survey of Software Engineering
   {from Dr. Chen}
   Explain the following software process models and point out their pros and cons:
   (a) incremental development model
   (b) agile development model

2. CS5392 Formal Methods in Software Engineering
   {from Dr. Podorozhny}
   Given a Kripke model $M=\langle W, R, L \rangle$, where $W=\{x_1, \ldots, x_n\}$, for some $n \in \mathbb{N}$ and $n \geq 2$; relations $R = \{(x_2, x_3), (x_3, x_4), \ldots, (x_{n-1}, x_n), (x_n, x_2)\} \cup \{(x_1, x_2), (x_1, x_3), \ldots, (x_1, x_n)\}$, and the labeling function $L$ is defined as
   \[
   L(x_k) = \begin{cases} 
   \{p,q,s\}, & \text{if } k \text{ is odd and } 2 \leq k \leq n \\
   \{q,r\}, & \text{if } k \text{ is even and } 2 \leq k \leq n \\
   \{s,r\}, & \text{if } k \text{ is equal to } 1
   \end{cases} \tag{1}
   \]
   (a) Show whether $\varphi = (((s \land q) \rightarrow r) \rightarrow (q \rightarrow \square r)) \rightarrow (\diamond q \land \diamond r)$ ever holds in world $x_1$. Explain why.
   (b) Show whether $\phi = (\square q \lor ((p \rightarrow q) \land (q \rightarrow \neg r))) \rightarrow (\diamond q \land \diamond r)$ holds in all worlds $x_k$ where $k$ is odd, i.e., $k = 2 \cdot s + 1$, $s \in \mathbb{N}$. Explain why.

3. CS5393 Software Quality
   {from Dr. Yang}
   1. Consider the following program that uses binary search to check whether the input array contains a specific input value, and the 4 tests, and answer the following questions.
   t1: \{1,2,3\}, 2
   t2: \{1,2,3\}, 1
   t3: \{1,2,3\}, 3
   t4: \{1,2,3\}, 4
(a) Draw the control flow graph for the program; annotate the CFG with \(s_1\), \(s_2\), ..., and \(b_1\), \(b_2\), ...

(b) Compute the statement and branch coverage for each test.

(c) If \(s_8\) is changed during software evolution, select the set of affected tests.

(d) Compute the du-pairs covered by test \(t_3\) with respect to variables \(low\) and \(high\). Note that you will lose points if you write more than necessary.

2. Consider the method \(Min\) and its 6 mutants in Figure 1. Provide reachability conditions, infection conditions, propagation conditions, and test case values to kill mutants \(\Delta_1\), \(\Delta_2\), \(\Delta_5\).
<table>
<thead>
<tr>
<th>Original Method</th>
<th>With Embedded Mutants</th>
</tr>
</thead>
</table>
| `int Min (int A, int B)` {  
  `int minVal;`  
  `minVal = A;`  
  `if (B < A)` {  
    `minVal = B;`  
  }  
  `return (minVal);`  
} // end Min | `int Min (int A, int B)` {  
  `int minVal;`  
  `minVal = A;`  
  `if (B < A)` {  
    `minVal = B;`  
  }  
  `return (minVal);`  
} // end Min |
|                 |  
  `Δ1`  
  `Δ2`  
  `Δ3`  
  `Δ4`  
  `Δ5`  
  `Δ6`  
  `Bomb();`  
  `minVal = A;`  
  `minVal = failOnZero (B);`  
  `return (minVal);`  
} // end Min |