Problem 1. There are four explicit conversions namely \texttt{static\_cast}, \texttt{dynamic\_cast}, \texttt{const\_cast}, and \texttt{reinterpret\_cast} in C++.
(a) Please explain when and how each of these conversions should be used and give a simple example for each cast. [16%]
(b) Please explain the distinction between \texttt{static\_cast} and \texttt{dynamic\_cast}. [4%]

Problem 2. A class \texttt{X} is declared as follows.
\begin{verbatim}
class X {
    public:
        int x;
        int f(double);
    
};
\end{verbatim}
(a) Please describe the syntax of declaring pointer to member functions and variables using the members of \texttt{X} as examples. [10%]
(b) Please describe the binding and using of the above declarations. [10%]

Problem 3. Please draw the structure of the class diagram and the collaboration with interaction (sequence) diagram for the following design patterns: [20%]
(a) Visitor
(b) Observer

Problem 4. Describe, in your last homework, how you implement the transition of one state to another (note: you should discuss all possible transitions). Please provide related class diagram, implementations of related functions (e.g., \texttt{ChangeState()}), and explanations. [20%]

Problem 5. Encapsulating variation is the theme of many design patterns. When an aspect of a program changes frequently, these patterns define an object that encapsulates that aspect. For example, in a Strategy pattern, a Strategy object encapsulates an algorithm so that the algorithm is interchangeable and can be varied independently. Please give at least 5 other examples of design patterns that encapsulate variations. Please describe clearly the aspect that is encapsulated in each case. [20%]