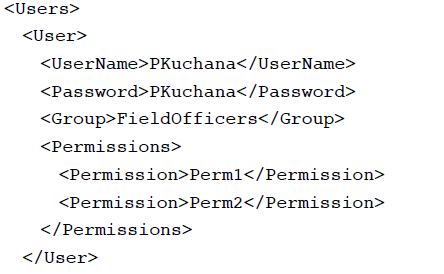
1. Let us consider an online job site that receives XML data files from different employers with current openings in their organizations. When the number of vacancies is small, employers can enter details online. When the number of vacancies is large, employers upload details in the form of an XML file. Once the XML file is received, it needs to be parsed and processed. Let us assume the XML file to have the following details:
   1. Job title
   2. Minimum qualifications
   3. Medical insurance
   4. Dental insurance
   5. Vision care
   6. Minimum number of hours of work
   7. Paid vacation
   8. Employer name
   9. Employer address In general,

Details from (c) through (i) are all considered being the same for all jobs posted by a given employer. Apply the required pattern to design the process of parsing the input XML file and creating different JOB objects

1. A computer user in a typical organization is associated with a user account. A user account can be part of one or more groups. Permissions on different resources (such as servers, printers, etc.) are defined at the group level. Users get all the permissions defined for all groups that their accounts are part of. Let us consider an organization with three different user groups — Administrators, FieldOfficers and SalesResps. Further assume that the organization is in the process of migrating user accounts from one server to a different server environment. As part of this process, all user accounts are first exported to an XML file as follows



It is to be noted that permissions for all accounts in a given group are the same. This can be considered as the intrinsic data. The user name and the password details vary from user to user and should be treated as extrinsic data. User accounts in the new server environment are created using the

Exported XML file. Make any necessary assumptions and design an application using the appropriate pattern to parse the XML file to create different user account objects.

1. Design an application that reads and writes different types of data (plain text, binary, etc.) to and from different destinations such as a file, a URL or a database. Apply the required pattern in designing the data read/write abstraction.
2. Design a code formatting application using the appropriate pattern. In general, programs can be written in any computer language (e.g., Java, VB, etc.) and a given program can be formatted in different ways such as simple text formatting, HTML formatting, color formatting and others. In effect, the interface for code formatting can be implemented in many different ways. Apply the required pattern to separate the interface from its implementations
3. Inside a house there are appliances that we can turn on or off, such as the Floor lamp, TV, and Vacuum cleaner. There are different ways to turn the appliance ON or OFF, such as using the on/off switch, the pull switch, or using a remote control. The concept of turning the appliance on or off is the Abstraction part and the user only needs to know the Abstraction part.

The second part is the Implementation, is the part that turns the appliance ON or OFF when the floor lamp, TV, or Vacuum cleaner receives the signal to turn on or off. It uses its internal implementation to perform the action. The user should not be bothered about, how the appliance is turned on or off.

1. Refactor using appropriate Pattern

