Collection:

1. Create an abstract class Medicine to represent a drug manufactured by a pharmaceutical company with attributes price and expiry date and a method getDetails() to print them.
   - Also include an abstract function displayLabel() in the Medicine class.

   - Derive **Tablet, Syrup and Ointment** classes from the Medicine class. Override the displayLabel() function in each of these classes to print additional information suitable to the type of medicine. For example, in case of tablets, it could be "store in a cool dry place", in case of ointments it could be "for external use only" etc.
   - Create a class **TestMedicine** with the main method that declares an array of Medicine references
   - Check the polymorphic behavior of the displayLabel() method.

2. WAP to store details for employees of ABC corp.
Emp info to be stored is : id, name, ph-no, emailed, dept, designation, date of joining. Employee class is abstract with 2 methods:
   - displayEmpDetails() which is concrete implementation
   - calulateSalary() which is abstract

Employees are of three types: SalariedEmp(basic-salary) , SalesEmp (monthly-sales) and ContractEmp (hrs, rate-per-hour).

   - Calculate salary of SalariedEmp as follows:
     DA = 20% of Basic_Pay
     HRA = 10% of Basic_Pay
     Salary = Basic_Pay + DA + HRA
   - Calculate salary of contractEmp as hours worked * rate per hour
   - Calculate salary of Sales Emp as 10% of monthly-sales

Create an array of different types of employees. Iterate thru the array and display employeedetails as well as salary for each employee.

3. Create a class Account (account-id, name, type, balance, creation-date)
   Create an AccountManager class that has the following menu:
   1. Add new Account
   2. Close account
   3. Withdraw amount
   4. Deposit amount
   5. Check balance
   6. Exit

Based on selected menu option, use AccountDao class to perform the relevant operations.

Create a AccountDaoIntf interface that has methods mentioned below.
AccountDao class will implement this interface and has following:

   - Arraylist<Account> accountlist;

Methods (populate a few records in this class using constructor):

   - addAccount(Account a)
   - closeAccount(Account a)

DO NOT COPY - confidential

- debitAccount(Account a, double amount) – while debiting from account, check if min balance of 5000 is maintained, else raise InsufficiantFundsException
- creditAccount(Account a, double amt)
- showBalance(int acct-id)

4. Design an application for tracking the customer details.

Create a java class Customer (custId, custName, contactNo, address, dob)

Write a java class CustomerDAO with a Linkedlist of customers
Perform following operations on Customer class. Write separate method for each functionality.

- void insert(Customer c) : the record of a new Customer.
- void update(id, newContactNo) : the contact number of a customer based on customer id.
- void delete(id) : the customer record by taking the customer id.
- void displayAll() : retrieve all customer details.
- void displayCustomerById(int id)

Write a Java class CustCRUDManager with below menu based application to do following operations –

- Insert new Customer details.
- Update the customer details.
- Delete the customer record.
- Display all customer details.
- Exit from the menu and terminate the program.

DO NOT COPY - confidential