**import** java.util.\*;

**//Composite Pattern**

**public** **class** CompositePattern {

**public** **static** **void** main(String[] args) {

Employee employee1 = **new** Associate("Mary", "2345", "technician");

Employee employee2 = **new** Associate("Mike", "3215", "Machine Operator");

Employee employee3 = **new** Associate("Evan", "3231", "Sales");

Employee bob = **new** Manager("Bob", "0000", "CEO", **new** Employee[]{

**new** Manager("Joe", "1212", "production manager", **new** Employee[]{ employee1, employee2 }),

**new** Manager("Beth", "1234", "marketing manager", **new** Employee[]{ employee3 })} );

bob.doSurvey("Are you happy?");

}

}

**abstract** **class** Employee{

String name, employeeId;

**public** Employee(String n, String id){

name = n;

employeeId = id;

}

**public** **abstract** **void** doSurvey(String question);

}

**class** Manager **extends** Employee{

String rank;

ArrayList<Employee> workersUnder;

**public** Manager(String n, String id, String rank, Employee[] workers){

**super**(n, id);

**this**.rank = rank;

workersUnder = **new** ArrayList<Employee>();

**for**(**int** i = 0; i < workers.length; i++){

workersUnder.add(workers[i]);

}

}

@Override

**public** **void** doSurvey(String question) {

System.*out*.println("I am " + **this** + ", I vote YES!");

**for**(**int** i = 0; i < workersUnder.size(); i++){

workersUnder.get(i).doSurvey(question);

}

}

**public** String toString(){

**return** name + " " + rank;

}

}

**class** Associate **extends** Employee{

String rank;

**public** Associate(String n, String id, String rank){

**super**(n, id);

**this**.rank = rank;

}

@Override

**public** **void** doSurvey(String question) {

System.*out*.println("I am " + **this** + ", I vote NO!");

}

**public** String toString(){

**return** name + " " + rank;

}

}

// **Decorator Pattern**

**public** **class** Decorator {

**public** **static** **void** main(String[] args) {

IGymMembership basic = **new** BasicGymMembership("abc2014");

IGymMembership premium = **new** PremiumGymMembership("eds2014", basic);

premium.getMembershipDescription();

}

}

**interface** IGymMembership{

**void** getMembershipDescription();

}

**class** BasicGymMembership **implements** IGymMembership{

String memberId;

**public** BasicGymMembership(String id){ memberId = id; }

@Override

**public** **void** getMembershipDescription() {

System.*out*.println("Member: " + memberId + " has the following benefit:\n" +

"use of all gym facilities.");

}

}

**class** PremiumGymMembership **implements** IGymMembership{

String memberId;

IGymMembership basicMembership;

**public** PremiumGymMembership(String id, IGymMembership basic){

memberId = id;

basicMembership = basic;

}

@Override

**public** **void** getMembershipDescription() {

**this**.basicMembership.getMembershipDescription();

System.*out*.println("In addition, you can schedule personal trainning sessions for free.");

}

}

**//Adaptor Pattern**

**public** **class** Adapter {

**public** **static** **void** main(String[] args) {

NewSystem.AccountStanding acctStanding = **new** NewSystem().getAcctStanding(**new** Account());

System.*out*.println(acctStanding);

}

}

**class** Account{

String custName = "Joe";

**double** balance = 100.00;

String passCode = "1203";

String acctId = "234213";

}

**class** NewSystem{

**public** AccountStanding getAcctStanding(Account acct){

AccountStanding standing = **new** AccountStanding();

standing.status = verifyAcct(acct);

standing.currentBalance = LegacySystem.*getBalance*(acct.custName, Integer.*parseInt*(acct.acctId), acct.passCode);

**return** standing;

}

**public** **boolean** verifyAcct(Account acct){ **return** **true**; }

**class** AccountStanding{

**boolean** status;

**double** currentBalance;

**public** String toString(){

**return** "Good standing? " + status + " Current balance: " + currentBalance;

}

}

}

**class** LegacySystem{

**public** **static** **double** getBalance(String customerName, **int** acctId, String securityCode){

**return** 100.00;

}

}

**/\* Façade Pattern (**A facade or façade /fəˈsɑːd/ is generally one exterior side of a building, usually, but not always, the front. The word comes from the French language, literally meaning "frontage" or "face"**) \*/**

**public** **class** Facade {

**public** **static** **void** main(String[] args) {

DiplomaHandler handler = **new** DiplomaHandler();

**if**(handler.handleGraduation(**new** Student()))

System.*out*.println("Student is ready to graduate");

}

}

**class** DiplomaHandler{

Registrar registrar;

BusinessOffice business;

CommencementHandler agent;

**public** DiplomaHandler(){

registrar = **new** Registrar();

business = **new** BusinessOffice();

agent = **new** CommencementHandler();

}

**public** **boolean** handleGraduation(Student s){

**if**(registrar.hasAppliedForGraduation(s)){

**if**(registrar.auditTranscript(s) && business.auditStudentDebts(s)){

agent.orderDiploma();

**return** **true**;

}

}

**return** **false**;

}

}

**class** Student{

String name;

}

**class** Registrar{

**public** **boolean** hasAppliedForGraduation(Student s){ **return** **true**; }

**public** **boolean** auditTranscript(Student s){ **return** **true**; }

}

**class** BusinessOffice{

**public** **boolean** auditStudentDebts(Student s){ **return** **true**; }

}

**class** CommencementHandler{

**public** **boolean** orderDiploma(){ **return** **true**; }

}