Data access objects

MVC - another view class

Ownership – an object “owns” another object

Inheritance

Data access objects

\*grab data that will be stored in our model objects

A text file

A database

A network connection

\*we will say that DAO are model classes (data)

MVC – model view controller

Model – data

View – how to access and see data

Controller – main class (coordinates interaction of model and view)

Example – input employee data from a text file

Ownership – class has data members that are themselves objects of classes

The owned object has to be created before you access it

To initialize an inner object(the owned object)

1. Pass in one that has already been created
2. Create it in the constructor for the outer (i.e. owning) class -composition

Class SteeringWheel{

}

Class Car{

Private SteeringWheel steeringWheel;

Public void turn( ){

steeringWheel.turn( );

} public Car(){

steeringWheel = new SteeringWheel(); //composition

}

Public class DrivingApp{

Psv main(string[] args){

Car c = new Car();

c.setSteeringWheel(new SteeringWheel()); //aggregation

a subclass inherits all ints and functions the super class has

public class SuperClass{

private int x;

private string s;}

public int getX(){ }

public void setX(int x){

public class SubClass extends SuperClass{

//has built into it without redclaring x and s

Public void someFunc(){

setX(7)

getX()

public class SubClass extends SuperClass{

@Overrride

Public String toString(){

}

@Override

Public void doSomething(){

Super.doSomething()

}

Public SubClass(){

}

Public SubClass(int x, String s){

Super(x,s)🡨call the super class’s constructor to initialize inherited variables

}

An abstract function has no body

It announces that subclasses will implement that functionality

When you add an abstract function to the class, you have to make the class abstract

ArrayList<Person> persons = ArrayList<Person>();

Persons.add(new Employee());

Persons.add(new Customer());

A more general variable (i.e. superclass type) can refer to subclass objects

For(person p : persons){

System.out.println(p);

Polymorphism: the ability to access specific subclass functionality from a superclass variable

Virtual method table (VMT)

toString() contact()

Employee 1 2

Customer 3 4

Supplier 5 6

An abstract function is one that has no body. Its purpose is to announce that subclass must implement it. If a class has an abstract function in it, it must be labeled abstract.

Public abstract class Name

Can’t create an object of an abstract class.

Any subclass of an abstract class must implement all its abstract functions

Implements-style inheritance involves creating a class that implements as Interface

An Interface a data type(just like a class is) but it only has public abstract functions inside it

A class implements an interface by overriding and providing a body for all the interface’s abstract functions.

A class can only ever extend one other class, but you can implement multiple interfaces