import java.util.ArrayList;   
  
public class Main {  
  
 public static void main(String[] args) {  
  
 Dessert theCake = new Cake("Chocolate");  
 Dessert theIceCream = new IceCream("Vanilla");  
 Dessert thePie = new Pie("Cherry");  
 Dessert theCookie = new Cookie("Peanut Butter");  
  
 ArrayList<Dessert> allDesserts = new ArrayList<>();  
  
 allDesserts.add(theCake);  
 allDesserts.add(theIceCream);  
 allDesserts.add(thePie);  
 allDesserts.add(theCookie);  
  
 for(Dessert d: allDesserts)  
 System.*out*.println(d.toString());  
  
 }  
  
}

import java.util.ArrayList;  
import java.util.Scanner;  
  
public abstract class Dessert {  
  
 private String flavor;  
 private ArrayList<String> ingredients = new ArrayList<>();  
  
 protected boolean isStored;  
  
 static Scanner *scan* = new Scanner(System.*in*);  
  
 public Dessert(String f){  
 flavor = f;  
 setIngredients(this);  
  
 isStored = false;  
 }  
  
  
 public String getIngredients(){  
 String allIngredients = "";  
  
 for(String s: ingredients){  
 allIngredients += s+", ";  
 }  
  
 return allIngredients;  
 }  
  
 public String toString() {  
 String theDessert = "This dessert is "+flavor+" flavored and is made with ";  
  
 for(String s: ingredients){  
 theDessert += s+", ";  
 }  
  
 return theDessert;  
 }  
  
  
 public void setIngredients(Dessert d){  
 boolean done = false;  
 String input = "";  
  
 while(!done){  
  
 if(d instanceof Cake)  
 System.*out*.println("Enter the ingredient you want to add to this CAKE, or type 'done' to finish");  
 else if(d instanceof Pie)  
 System.*out*.println("Enter the ingredient you want to add to this PIE, or type 'done' to finish");  
 else if(d instanceof Cookie)  
 System.*out*.println("Enter the ingredient you want to add to this COOKIE, or type 'done' to finish");  
 else if(d instanceof IceCream)  
 System.*out*.println("Enter the ingredient you want to add to this ICE CREAM, or type 'done' to finish");  
  
  
 input = *scan*.nextLine();  
 if(input.equalsIgnoreCase("done"))  
 done = true;  
 else{  
 ingredients.add(input);  
 }  
 }  
 }  
  
 public abstract void create();  
  
 public abstract void prepare();  
  
 public abstract void finish();  
  
 public abstract void store(boolean b);  
}

public class IceCream extends Dessert {  
  
 private boolean isFrozen;  
  
 public IceCream(String flavor) {  
 super(flavor);  
 isFrozen = true;  
 }  
  
 public void make(){  
 System.*out*.println("Ice Cream made!");  
 }  
 public void eat(){  
 System.*out*.println("Ice Cream has been eaten!");  
 }  
  
 @Override  
 public void create(){  
 System.*out*.println("Ice Cream created!");  
 }  
  
 @Override  
 public void finish() {  
 System.*out*.println("Ice Cream Finished!");  
 }  
  
 @Override  
 public void store(boolean b){  
 isStored = true;  
 }  
  
 @Override  
 public void prepare(){  
 System.*out*.println("Ice Cream Prepared!");  
 }  
  
 public String toString(){  
 String theString = super.toString()+" is Ice Cream and ";  
 if(isFrozen)  
 theString+= "IS frozen";  
 else  
 theString += "IS NOT frozen";  
  
 return theString;  
 }  
  
}

public class Cake extends Dessert{  
  
 private String shape;  
  
 public Cake(String flavor){  
 super(flavor);  
 shape = "round";  
 }  
  
 public void bake(){  
 System.*out*.println("Cake has been BAKED!");  
 }  
 public void eat(){  
 System.*out*.println("Cake has been eaten!");  
 }  
  
 @Override  
 public void create(){  
 System.*out*.println("Cake created!");  
 }  
  
 @Override  
 public void finish() {  
 System.*out*.println("Cake Finished!");  
 }  
  
 @Override  
 public void store(boolean b){  
 isStored = true;  
 }  
  
 @Override  
 public void prepare(){  
 System.*out*.println("Cake Prepared!");  
 }  
  
 public String toString(){  
 String theString = super.toString()+" is Cake and "+shape+" shaped.";  
 return theString;  
 }  
  
}

public class Cookie extends Dessert {  
  
 private String shape;  
  
 public Cookie(String flavor){  
 super(flavor);  
 shape = "round";  
 }  
  
 public void bake(){  
 System.*out*.println("Cookie has been BAKED!");  
 }  
 public void eat(){  
 System.*out*.println("Cookie has been eaten!");  
 }  
  
 @Override  
 public void create(){  
 System.*out*.println("Cake created!");  
 }  
  
 @Override  
 public void finish() {  
 System.*out*.println("Cake Finished!");  
 }  
  
 @Override  
 public void store(boolean b){  
 isStored = true;  
 }  
  
 @Override  
 public void prepare(){  
 System.*out*.println("Cookie Prepared!");  
 }  
  
 public String toString(){  
 String theString = super.toString()+" is a Cookie and "+shape+" shaped.";  
 return theString;  
 }  
}

public class Pie extends Dessert{  
  
 private boolean isCold;  
  
 public Pie(String flavor){  
 super(flavor);  
 isCold = true;  
 }  
  
 public void bake(){  
 System.*out*.println("Pie has been BAKED!");  
 }  
 public void eat(){  
 System.*out*.println("Pie has been eaten!");  
 }  
  
 @Override  
 public void create(){  
 System.*out*.println("Cake created!");  
 }  
  
 @Override  
 public void finish() {  
 System.*out*.println("Cake Finished!");  
 }  
  
 @Override  
 public void store(boolean b){  
 isStored = true;  
 }  
  
 @Override  
 public void prepare(){  
 System.*out*.println("Pie Prepared!");  
 }  
  
 public String toString(){  
 String theString = super.toString()+" is Pie and ";  
 if(isCold)  
 theString+= "IS a cold Pie";  
 else  
 theString += "IS NOT a cold Pie";  
  
 return theString;  
 }  
  
}