Sams Teach Yourself Object Oriented Programming in 21 Days

International Standard Book Number: 0-672-32109-2

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The author and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book.

When reviewing corrections, always check the print number of your book. Corrections are made to printed books with each subsequent printing. To determine the printing of your book, view the copyright page. The print number is right-most number on the line below the "First Printing" line. For example, the following indicates the 4th printing of a title.

First Printing: September 2001

06 05 04 03 10 9 8 7 6 5 4

Misprint	Correction
Page 61, paragraph after "Problem Statement" Use the poker card description design classes to represent the cards, the deck of cards, and the dealer.	 Use the poker card description **to** design classes to represent the cards, the deck of cards, and the dealer.
Page 121, first line of Listing 5.12 public class Stack **extends {**	 public class Stack **extends Vector {**
Page 129, last sentence of the page, which runs over to the top of page 130: So if you pass in an ExtrovertedObject, polymorphism will ensure that ExtrovertedObject's definition of speak() gets called, not the one found in **the base class**.	 So if you pass in an ExtrovertedObject, polymorphism will ensure that ExtrovertedObject's definition of speak() gets called, not the one found in **PersonalityObject**.
Page 131, first paragraph after the code at top of page: Substitutability allows you to pass any PersonalityObject to the method, and polymorphism ensures that the proper method is called on the instance. Polymorphism will call the method based on the instance's true type (PersonalityObject, OptimisticObject, IntrovertedObject, ExtrovertedObject, or PessimisticObject), no on the instance's apparent type (PersonalityObject).	 Substitutability allows you to pass any PersonalityObject to the method, and polymorphism ensures that the proper method is called on the instance. Polymorphism will call the method based on the instance's true type (PersonalityObject, OptimisticObject, IntrovertedObject, ExtrovertedObject, or PessimisticObject), no on the instance's apparent type (PersonalityObject). **Remember, polymorphism uses an automatic mechanism to pick the right code.**

Page 133, code at top of page:
 	
`}`
 	
`protected void log( String message, String level, String time ) {`
`  pw.println( level + ": " + time + ": "+ message );`
`  pw.flush();`
`}`
 	
`public void close() {`
	
	`}`
	
	`protected void log( String message, String level, String time ) {`
	`  pw.println( level + ": " + time + ": "+ message );`
	`  pw.flush();`
	`}`
	
	`public void close() {`

`pw.close();`
`}`
 	
`public class ScreenLog extends BaseLog {`
`protected void log( String message, String level, String time ) {`
`System.out.println( level + ": " + time + ": " + message );`
`}`
`}`
	`pw.close();`
	`}`
	
	`}`
	`public class ScreenLog extends BaseLog {`
	`protected void log( String message, String level, String time ) {`
	`System.out.println( level + ": " + time + ": " + message );`
	`}`
	`}`

Page 170, third paragraph up from the bottom: Conditionals are contrary to the concepts of OO. In OO you're not supposed to ask an object for its data and then do something to that data. Instead, you're supposed to ask an object to do something to its data. In the case of the `day_of_the_week()` method you probably obtain `day` from some object. You shouldn't be processing raw data. Instead, you should ask the object **for a string representation**. Conditionals force you to muddle responsibilities. Each place that uses the data will have to apply the same conditional logic.	Conditionals are contrary to the concepts of OO. In OO you're not supposed to ask an object for its data and then do something to that data. Instead, you're supposed to ask an object to do something to its data. In the case of the `day_of_the_week()` method you probably obtain `day` from some object. You shouldn't be processing raw data. Instead, you should ask the object **to print itself to the screen or to return a string representation of itself. Leave it to the object to figure out what its string representation should be**. Conditionals force you to muddle responsibilities. Each place that uses the data will have to apply the same conditional logic.
Page 239, Figures 10.5 and 10.6, right columns: ShippingInformation **ShippingCart** Order OrderDisplay	ShippingInformation **ShoppingCart** Order OrderDisplay
Page 240, Figure 10.8, right column RegisteredUser **ShippingCart** Order	RegisteredUser **ShoppingCart** Order

OrderDisplay PaymentTerminal Order	OrderDisplay PaymentTerminal Order

This errata sheet is intended to provide updated technical information. Spelling and grammar misprints are updated during the reprint process, but are not listed on this errata sheet.