

# Java<sup>TM</sup> 2

## Exam Notes<sup>TM</sup>

PROGRAMMER'S  
EXAM



**Philip Heller**

- Reinforce Your Knowledge
- Review All Key Topics
- Test Effectively

**Perfect Companion to the  
Sybex Study Guide**



# Contents

## Introduction

xvii

## Chapter 1 Declarations and Access Control

1

- Write code that declares, constructs, and initializes arrays of any base type using any of the permitted forms both for declaration and for initialization. 2
- Declare classes, inner classes, methods, instance variables, static variables, and automatic (method local) variables, making appropriate use of all permitted modifiers (such as *public*, *final*, *static*, *abstract*, and so forth). State the significance of each of these modifiers, both singly and in combination, and state the effect of package relationships on declared items qualified by these modifiers. 7
- For a given class, determine if a default constructor will be created and, if so, state the prototype of that constructor. 24
- State the legal return types for any method, given the declarations of all related methods in this or parent classes. 26

## Chapter 2 Flow Control and Exception Handling

29

- Write code using *if* and *switch* statements and identify legal argument types for these statements. 30
- Write code using all forms of loops including labeled and unlabeled use of *break* and *continue* and state the values taken by loop control variables during and after loop execution. 33

- Write code that makes proper use of exceptions and exception handling clauses (*try*, *catch*, *finally*) and declare methods and overriding methods that throw exceptions. 41

### **Chapter 3 Garbage Collection 49**

- State the behavior that is guaranteed by the garbage collection system and write code that explicitly makes objects eligible for collection. 50

### **Chapter 4 Language Fundamentals 57**

- Identify correctly constructed source files, package declarations, import statements, class declarations (of all forms including inner classes), interface declarations and implementations (for *java.lang.Runnable* or other interface described in the text), method declarations (including the *main* method that is used to start execution of a class), variable declarations and identifiers. 58
- State the correspondence between index values in the argument array passed to a main method and command line arguments. 70
- Identify all Java programming language keywords and correctly constructed identifiers. 73
- State the effect of using a variable or array element of any kind when no explicit assignment has been made to it. 76
- State the range of all primitive data types and declare literal values for *String* and all primitive types using all permitted formats, bases, and representations. 79

**Chapter 5 Operators and Assignments****85**

- Determine the result of applying any operator, including assignment operators, *instanceof*, and casts, to operands of any type class, scope or accessibility, or any combination of these. 86
- Determine the result of applying *boolean equals(Object)* method to objects of any combination of the classes *java.lang.String*, *java.lang.Boolean*, and *java.lang.Object*. 101
- In an expression involving the operators *&*, *|*, *&&*, *||*, and variables of known values, state which operands are evaluated and the value of the expression. 104
- Determine the effect upon objects and primitive values of passing variables into methods and performing assignments or other modifying operations in that method. 104

**Chapter 6 Overloading, Overriding, Runtime Type, and Object Orientation****109**

- State the benefits of encapsulation in object-oriented design and write code that implements tightly encapsulated classes and the relationships “is a” and “has a”. 110
- Write code to invoke overridden or overloaded methods and parental or overloaded constructors; and describe the effect of invoking these methods. 114
- Write code to construct instances of any concrete class including normal top level classes, inner classes, static inner classes, and anonymous inner classes. 124

<b>Chapter 7</b>	<b>Threads</b>	<b>133</b>
	<ul style="list-style-type: none"><li>• Write code to define, instantiate, and start new threads using both <i>java.lang.Thread</i> and <i>java.lang.Runnable</i>. 134</li><li>• Recognize conditions that might prevent a thread from executing. 138</li><li>• Write code using <i>synchronized</i>, <i>wait</i>, <i>notify</i>, and <i>notifyAll</i> to protect against concurrent access problems and to communicate between threads. Define the interaction between threads and between threads and object locks when executing <i>synchronized</i>, <i>wait</i>, <i>notify</i>, or <i>notifyAll</i>. 143</li></ul>	
<b>Chapter 8</b>	<b>The <i>java.awt</i> Package</b>	<b>153</b>
	<ul style="list-style-type: none"><li>• Write code using <i>Component</i>, <i>Container</i>, and <i>Layout Manager</i> classes of the <i>java.awt</i> package to present a GUI with the specified appearance and resize behavior, and distinguish the responsibilities of layout managers from those of containers. 154</li><li>• Write code to implement listener classes and methods, and in listener methods, extract information from the event to determine the affected component, mouse position, nature, and time of the event. State the event classname for any specified event listener in the <i>java.awt.event</i> package. 164</li></ul>	
<b>Chapter 9</b>	<b>The <i>java.lang</i> Package</b>	<b>171</b>
	<ul style="list-style-type: none"><li>• Write code using the following methods of the <i>java.lang.Math</i> class: <i>abs()</i>, <i>ceil()</i>, <i>floor()</i>, <i>max()</i>, <i>min()</i>, <i>random()</i>, <i>round()</i>, <i>sin()</i>, <i>cos()</i>, <i>tan()</i>, <i>sqr()</i>. 172</li></ul>	

- Describe the significance of the immutability of String objects. 179

**Chapter 10 The *java.util* Package 185**

- Make appropriate selection of collection classes/interfaces to suit specified behavior requirements. 186

**Chapter 11 The *java.io* Package 199**

- Write code that uses objects of the *File* class to navigate a file system. 200
- Write code that uses objects of the classes *InputStreamReader* and *OutputStreamWriter* to translate between Unicode and either platform default or ISO 8859-1 character encodings, and distinguish between conditions under which platform default encoding conversion should be used and conditions under which a specific conversion should be used. 203
- Select valid constructor arguments for *FilterInputStream* and *FilterOutputStream* subclasses from a list of classes in the *java.io.package*. 207
- Write appropriate code to read, write and update files using *FileInputStream*, *FileOutputStream*, and *RandomAccessFile* objects. 209
- Describe the permanent effects on the file system of constructing and using *FileInputStream*, *FileOutputStream*, and *RandomAccessFile* objects. 213

<b>Appendix A</b>	<b>The Certification Initiative for Enterprise Development</b>	<b>217</b>
	• The Structure of the Initiative	219
	• The Second-Level Exams	219
	• The Third-Level Exams	230
<b>Glossary</b>		<b>237</b>
<i>Index</i>		<b>246</b>