

Table of Contents

Preface	vi
Laboratory 1—Computational Java	1
Arithmetic, assignments, declarations, and the form of a basic Java program are the heart of this laboratory. In a hands-on way, you will learn how to modify programs, compile programs, and run them. The methods of the Math class are used in the creation of a complete program.	
Laboratory 2—Leap Years	19
Basic applications of if statements are illustrated. The problem of determining whether a year is a leap year motivates an examination of multiple solutions, some of which illustrate faulty logic. The laboratory session focuses on short programs using if statements. Of course, boolean operators and expressions are introduced and applied.	
Laboratory 3—Object-Oriented Java	28
Classes that model currency converters and automated teller machines illustrate the fundamentals of object-oriented programming. The objects have a visual component that allows the user to interact with them. You will modify a complete program and fill in key parts of a large program. The missing parts will require the use of information from the first two laboratories.	
Laboratory 4—Repeating Means Looping	43
The while and for loops are introduced and their equivalent forms are explored. Simple applications based on arithmetic operations will include the determination of all factors of a positive integer and the subsequent identification of perfect and prime numbers.	
Laboratory 5—String Basics and Applications	56
The basic concept of a string is introduced using the Java String class and its methods. Using material from the previous laboratory, you will determine how to create new strings from existing strings and how to determine when a string is a palindrome.	

- Laboratory 6—Zip Codes and Postal Bar Codes68**
 Focusing on the creation of small utility functions and procedures, this laboratory examines the coding and decoding of postal bar codes. You will learn how to write methods that can be used multiple times to solve problems. A large program to perform the coding and decoding provides a graphics display but depends on these small methods to work properly. You will complete these methods in the laboratory session.
- Laboratory 7—A Methods Toolbox75**
 Continuing the focus on methods and procedures, you will develop a number of static methods within a Toolbox class. These methods then can be reused later as required. This lab will prompt you to think about writing program components that will be reused.
- Laboratory 8—Collections87**
 In all programming languages, arrays are an essential data organization. In this laboratory arrays are introduced and explored with simple examples. The post-laboratory provides a brief look at the Vector class. Some simple methods involving arrays will be written and added to your Toolbox class.
- Laboratory 9—Drawing with Java.100**
 This laboratory uses methods of the Graphics object associated with a Canvas to create images. Using the principles of object-oriented programming, you will construct classes for rectangles, triangles, circles, and other special shapes. The post-laboratory extends this work to the creation of images for playing-card symbols—a heart, a spade, a diamond, and a club. The created classes all implement a simple interface illustrating the principle of common behavior among a collection of classes.
- Laboratory 10—Recursive Methods119**
 Simple principles of recursive functions and procedures are explored with examples. The recursive creation of graphics images using simple drawing is illustrated, together with the classic recursive method to find the greatest common divisor of two integers. A recursive version of the selection sort is tested. The laboratory will ask you to study and run some programs, as well as to write new programs using the recursive model presented in the pre-laboratory examples.
- Laboratory 11—Linear Algorithms128**
 Linear search, binary search, and sorting are the key topics of this laboratory. The two searches are implemented and tested, as is an iterative version of selection sort. The concept of algorithm efficiency is introduced in the pre-laboratory reading, where linear search and binary search are compared. Methods that implement these algorithms will be added to your Toolbox class.
- Laboratory 12—Sorting Out Sorts136**
 The merge sort algorithm is introduced and compared with selection sort. In the laboratory you will code a merge sort and then time the two sort algorithms on large data sets to determine if the empirical results support the claims of efficiency.
- Laboratory 13—I/O and Exceptions144**
 We have used LabPkg prior to this laboratory to avoid dealing with the problems of input from the keyboard. The focus of this laboratory is processing data from

files and from the console. Both require the introduction of exception handling, exceptions, and the try-catch mechanism. Most of the attention is centered on file processing in Java.

Laboratory 14—Java GUI Programming 157

The AWT and basic graphics classes are introduced and used in simple applications. This not only represents a set of objects that illustrate inheritance, but it provides the opportunity to bring event-driven programming front and center. You will develop a simple graphics program that visually records the positions of mouse clicks. This provides the foundation for the next laboratory.

Laboratory 15—Solitaire Games 168

Building on ideas of event-driven programming, this laboratory works through the design and creation of classes for a FreeCell solitaire game. This requires two sessions in the laboratory to complete, but results in a fully functional version of the game. It illustrates that careful design of a set of classes and their characteristics can make a large program manageable.

Laboratory 16—Klondike 183

With the addition of one class and the modification of another, we illustrate reuse of a large collection of classes by writing the Klondike solitaire game. The post-laboratory converts this game into an applet, showing how easy it is to convert an application to an applet.

Appendix A—The LabPkg Package 187

Appendix B—IDE Information 194

Index 198