Notes

**Chapter 1**

Program- A step by step series of instructions that tells a computer exactly what to do.

Computer programming- the process of writing that set of instructions for the computer to follow in order to produce a desired result.

Programming language- a set of words, symbols, and codes that enables a programmer to communicate instructions to a computer

Programmer, also called software developer- people who design and write programs using a programming language or program development tool

Coding- another word for programming. Used because the instructions written for a computer are called computer code

Java- a high level computer programming language.

High-level languages- allow programmers to write instructions using English-like commands and words instead of cryptic numeric codes or memory addresses

Syntax- The particular set of rules or grammar that specify how the instructions are to be written

Parsimonious- a word used to describe a computer language that has a compact set of commands without numerous versions or adaptations of the same command

Object-oriented programming- an approach to programming in which the data and the code that operates on the data are packaged into a single unit called an object;

A software object represents a real object, such as a person or thing, or object, such as a transaction or an event--- ex. A mouse click

Object-oriented design-an approach to program design that identifies how objects must interact with each other in order to solve a problem.

Robust- means that programmers can use java to develop programs that do not break easily or cause unexpected behaviors; and, if a program fails, does not corrupt data

Strongly typed language- means that it (Java) checks for potential problems with different types of data

Secure- means Java’s programs are easy to protect from viruses and tampering

Portable- means a program can be run on a variety of platforms other than the one in which it was created, without requiring major rework

Platform- the underlying hardware and software for a system

Platform-independent- means that you can use Java to write and run a program on many platforms, such as a Pc running Windows or Linux, an Apple running MacOS, or a server running UNIX.

Java Software Development Kit- a programming environment that allows you to build a wide range of Java program types

Application- a program that tells a computer how to accept input from a user and how to produce output in response to those instructions

Console application- aka a console-mode application, uses a command-line interface, such as a command prompt window, to support character output

Windowed application- uses a graphical user interface for user input and program output with on-screen elements such as text boxes, buttons, menus, and toolbars to support user interface

Applet- a small program that can be downloaded and executed as part of a displayed Web page

Client-side applet- an applet which is intended for general use by people browsing the web, and is executed on the client machine

Java Web Start- a client-side helper application that functions like windows media player or realone player, which launch when an audio file is downloaded or opened

Javascript- a scripting tool, created by Netscape, used to insert code statements directly into the HTML of a web page.

HTML- a set of special codes called tags that specify how the text and other elements of a web page display

Servlets- a Java program that is hosted and run on a web server rather than launched from a browser

Hosting- involves storing web pages, programs, and other files on a computer that a user connects via a network, intranet or the Web

Web server- a computer that hosts Web pages, programs, and other files, which it delivers or serves to requesting computers

Server software- software that responds to incoming requests and serves different forms of data, as well as interfacing with applications and servlets

Server-side- programs that are hosted and run on the server and are not executed on the client-side

Java Server Pages- a server-side technology that extends the Java servlet technology to provide a way to create sophisticated solutions that run on any platform

Java Database Connectivity- a Java interface that enables Java applications to execute queries and interact with most databases

Web service- a program that receives a request for information from another program over the web and then returns data to the requesting program

Javabean, or just bean- a reusable software component developed in Java, which can be used by any application that understands the JavaBeans format.

Reusability- means they can be used over and over in many programs developed by different programmers. A bean can also store data and retrieve it later.

Application software packages- programs required for common business and personal needs

Methodology- organized plan

* program development cycle

1. analyze the requirements

2. design the solution

3. validate the design

4. implement the design

5. test the solution

6. document the solution

requirements document- a list of the functions and features that the program must provide. Includes: a statement of purpose for the requested program, the formulas the program must use, and an explanation of how the program should respond to user interaction

algorithm- in programming an algorithm is a solution

Correct- using logical constructs and valid data in an organized way so that the steps will be carried out correctly and the program will make suitable responses to invalid data, such as displaying a warning message for numbers outside a given range

Efficient- refers to the program’s ability to deliver a result quickly enough to be useful and in a space small enough to fit the environment

Storyboard- hand-drawn sketch of how a window or applet will look and where the user interface elements will be placed in the window

Class- represents the common structure and behavior shared by the same type of objects

Class diagram- illustrates the name, attributes, and methods of a class of objects

Attributes- properties used to define characteristics such as appearance

Methods- instructions that the class uses to manipulate values, generate outputs, or perform actions

Flowchart- a graphical representation of the logic used to develop algorithms

Control structure- z portion of a program that allows the programmer to specify that code will be executed only if a condition is met

Psuedocode- expresses computer actions using keywords and depicts logical groupings or structure using indentation

Usability- a measure of a user’s ability to interact with a program in a reasonable and intuitive manner

Validate- means that both the programmer and the user must check the program design

Comments- notes within the code that explain the purpose of the code

unit testing- testing for the correctness during the programming process

testing- to verify that the program meets the requirements from the user’s point of view

test plan- consists of a collection of test cases

test cases- individual scenarios that include input data and expected output data, and are designed to ensure that the program solves the particular problem indicated in the program requirements

integration testing- testing done to ensure that all programs and components interact correctly

boundary value- values that cause a certain rule to become effective

documentation- the requirements documents, program design documents, user interface documents and documentation of the code

aggregation- a term used to describe the concept of an object being composed of other objects

generalization hierarchy- an object-oriented design tool used to show the relationship among classes

instance- a specific use of a class, a unique object

operation- aka service, is an activity that reads or manipulates the data of an object

message- activates the code to perform one of the operations

* two parts- the name of the object to which the message is being sent and the name of the operation that will be performed

trigger- or impetus, causes the message to be sent… may come from another object or external user

event- the entire process of a trigger sending a message that causes an operation to occur

event diagram- shows relationships among events and operations visually and to help plan how they will program events

event-driven- programs that, at the conclusion of the operation, will do nothing until another trigger causes an event to occur

Unified Modeling Language- provides a standardized model for object-oriented design to depict or describe concepts graphically

Encapsulation- the capability of an object to have data (properties) and functionality (methods) available to the user, without the user having to understand the implementation within the object

Transparent- an action that takes place without any visible effect other than the desired output

Information hiding- the process of making the implementation and programming details transparent to the user

Inheritance- means a programmer can use a class, along with its functions, to create a subclass, which saves time and coding

Subclassing- a subclass has at least one attribute or method that is different than differs from its superclass, but it inherits its functions and data of the superclass

Polymorphism- allows an instruction to be given to an object using a generalized rather than a specific, detailed command

Rapid application Development (RAD)- refers to the use of prebuilt objects to make program development much faster

Class providers- focus on creating classes and

Class users- leverage their knowledge of business processes to assemble applications using OOP (object oriented programming)

4 Benefits of object-oriented programming- Reusability, stability, Easier design, and faster design

Java 2 Standard Edition version 5.0 (J2SE)- contains Java SDK and Java JRE

Java 2 Runtime Environment (JRE)- Contains: Java Virtual Machine (JVM)

-Java API’s and Class Libraries

- Java Applet Viewer

- Other Tools

Java SDK contains: -Java Compiler

* Java Debugger
* Other Tools

Compiler- a program that converts a programmer’s code into machine-readable instructions

Bytecode- object code converted from source code by a Java compiler

interpreter- a program that executes the machine language instructions (from bytecode) to produce results, or answers

Java virtual machine (JVM)

-Execution of Java programs is initiated with the command java.exe

Application program interface (API)- a standard set of interfaces and classes functionally grouped into packages. Collections of classes, sometimes called libraries, which contain portable Java bytecode files

Class packages- aka class libraries- predefined packages of classes

Java applet viewer- a mini-browser designed to display Java applets without any of the version or enabling problems associated with common proprietary browsers such as MS Internet Explorer or Netscape Navigator

* appletviewer.exe is used to launch the Java applet viewer

value-added text editor- assists programmers by color-coding key elements in the Java code and inserting automatic line numbers… ie textpad, Jgrasp, or Jcreator

Integrated Development Environment (IDE)- assists programmers by displaying toolbars, menus, windows, and dialog boxes that are designed to facilitate coding and debugging…ie Sun Microsystems’ ONE Studio, JBuilder, Visual Age for Java, and Simplicity

**Chapter 2**

User interface- the way a user enters data and instructions into a computer and receives feedback from the computer

Splash screen- a screen that is displayed before the main program is displayed

Prototype- a functional working model of a proposed system, created to make sure it meets users’ needs

System date- the current date and time generated by the operating system of a computer

Default- a preset featureof the command prompt and applet viewer windows

TextPad- a value-added text editor (VATE) used to create many different kinds of text-based files, one of which is a Java source code file

Coding window- the area where you can enter and edit lines of Java code

Selector Window- displays a list of open TextPad files

Clip Library window- displays a list of special codes and tags used by some scripting tools and languages such as HTML

Comments- notes within the code that explain the purpose of the code

Block comment- begins with a forward slash followed by an asterisk (/\*) and ends with the symbols reversed, an asterisk followed by a forward slash (\*/)

Comment header- a comment that identifies the file and its purpose at the top of every class or file that contains code

Doc comment- a documentation comment, begins with a forward slash followed by two asterisks (/\*\*) and ends with an asterisk followed by a forward slash (\*/)

Line comment- or single line comment, is a comment that spans only a single line or a part of a line. Line comments begin with 2 forward slashes (//) which cause the rest of the line to be ignored during compilation and execution. Line comments have no ending symbol.

/\* Block comments \*/

/\*\*doc comments \*/

// line comments

tab characters- entered by pressing the *Tab* key and are used to indent items on each line

class header- identifies how the code will be accessed and specifies the class name

public- access modifier- aka a scope identifier, specifies the circumstances in which the class can be accessed

keywords- or reserved words,

case-sensitive- means that the Java compiler considers uppercase and lowercase as two different characters

All code entered after the class header is considered to be the body of the class and must be enclosed in braces { }

header- notifies the Java compiler of the method’s attributes, the type of data it will generate (in any), the name of the method, and any other information or parameters the method needs to perform a service or operation

main( ) method- the starting point during execution

Method modifier- used to set properties for the method

Static- a modifier that means this method is unique and can be invoked without creating a subclass or instance

Return value- the result or answer of a method

Void- a keyword used describe a method that does not return data

Parameter- a piece of data received by the method to help the method perform its operation

Argument- data sent to a method

Identifier- any word you choose to name an item in a Java program

Variable- a location in computer memory that can ws as the code executes

Data type- a word that describes the type or category of data the method uses

String[]- data type that indicates a series or string of characters

Body- lines of executable code. The body is enclosed in pairs of braces

Class definition- defines the instance and class variables and methods available for use in the class, as well as other information, such as the immediate superclass. You can use the Java API to view the hierarchy of individual classes and to look up specific methods used by each class

System class- contains several useful class variables and methods, such as those involving standard input, standard output, and other utility methods

Extends- or inherits

Out- refers to the object representing the default displays

println() method- returns its value to the System.out device

literal- means the data inside the quotes will be displayed exactly as entered in the code

debugging- the process of fixing errors

system error- occurs when a system command is not set up properly, software is installed incorrectly, or the location of stored files has changed

syntax error- an error caused by code statements that violate one or more syntax rules of the java programming language

Command results window- Where TextPad lists syntax errors in a command results window

Semantic error- an error that changes the meaning of the code

Logic error- occurs when a program does not behave as intended due to poor design or incorrect implementation of the design

Run-time error- aka an exception, is an error that occurs when unexpected conditions arise as you run or execute the program

Edit- modify the source code

Import statement- tells the compiler where to access the classes, fields, and methods of an existing class in the package. Placed at the beginning of the Java source code, most commonly right after the opening documentation

Date class- in Java represents a specific instant in time, measured to the nearest millisecond

Declare-

Constructor- declares the type of data or object to be stored and assigns it a variable name in the computer’s memory. Identified by the =new notation

Escape characters- moves the insertion point to the right. Escape characters are non-printing control codes

Concatenate- join… done in Java by using a plus sign (+)

Hard copy, or printout- a printed version of the source code

printf() method- embeds data within a string of text rather than concatenate it. Uses a percent sign (%) followed by one of several formatting characters to display formatted data

Applet package- allows applets to inherit certain attributes and manipulate classes

Abstract Window Toolkit- a package included with the SDK to provide programs access to color, draw methods, and other GUI elements commonly used in applets

Extends command- allows a one class to inherit the attributes and methods from its superclass

init() method- loads the initial setup of the applet when execution begins

paint() methods- graphically draws texts and an image on the applet screen after the applet is initialized

Reference variable- when a method refers to an instance of an object, such as Graphics g, the g is called a reference variable

drawstring() method- draws text in the applet window

toString() method- can be used to convert currentDate to a String; the code takes the form of the variable currentDate, followed by a period, followed by toString()

Image object-

getImage() method- used to load images into an apple. The getImage()method creates and returns an Image object that represents the loaded image.

getDocumentBase() method- a second method used by the getImage()method that allows the applet to pull the image from the current folder in which your applet class is stored

drawImage() method- specifies the location where the program should draw the graphic

setBackground() method- takes a color object and its attribute, Color.cyan, to change the background color of the applet window

host- reference program

**Chapter 3:**

Data- collections of raw facts and figures, such as words, text, or numbers, which are used in reasoning or calculations (datum is the singular form of the word, data).

Dialog box- a small window that displays messages and can accept user input

Modal- a dialog box that requires the user to complete a specific actions, such as entering data, clicking the Cancel button, or choosing yes or no before returning to the program’s interface

Javax.swing- a Java package that provides a set of Java-based GUI components

throwsIOException- Java programming code that gives the program a way to acknowledge and handle potential input or output errors and still compile correctly

precision- refers to the amount of storage allocated to hold the fractional part of a number

Strongly typed language- means it enforces a set of rules about how you use the objects you create, especially when using different types of data

Primitive data type- a data type that is structured by Java to hold single data items, such as integer, character, floating point, and true or false values

Reference data type- a data type whose value is an address

Declaration Statement- A line of Java code that identifies, or declares, the data type and names the identifier or variable

Interactive- the term used with programs that allow the user to interact with the program by making choices, entering data, and viewing results

Stream- in Java, is the act of data flowing in or out of a program

Buffer- a data area shared by hardware devices or programs, where data are held until they are needed by the processor. Buffering ensures that if a user presses the BACKSPACE key to delete a character, the deleted characters are not sent when the program retrieves the characters from the input buffer

InputStreamReader (ISR)- a Java class or object that serves as an intermediary between the input buffer and the Java program

Wrap- word used by Java programmers to describe how the ISR envelops the stream from the input buffer

BufferReader class- used to store, or buffer, the input received from another object or class, such as the InputStreamReader class. Used to increase the efficiency of character input

Instantiation- the process of constructing an instance of a data type or object from a previously defined class

Instantiate- or declare an instance of, the BufferReader class.

readLine() method- reads the line of inputted text and returns a String containing the contents of the line

wrapper class- provides new ways to help primitive data types, such as ints and doubles, conform to their Object class counterparts, such as Strings

parse() method- allows programmers to convert Strings to a numeric data type

assignment operator- (=)

assignment statement- a line of code beginning with a location, followed by the assignment operator (=), followed by the new data, method, or formula

accumulator, or counter- an accumulated total

Arithmetic operators- manipulate two or more numeric values

Order of operator precedence- a predetermined order that defines the sequence in which operators are evaluated and resolved when several operations occur in an expression

Integer division- performed when both the dividend and the divisor are integers. When performing integer division, Java forces the result to be an integer because it is a primitive data type; it then drops any remainder

Modular division- aka remainder division, is used to store any truncated remainder value from integer division

Modulus operator (%)- also called the remainder operator, is entered between two integers and performs modular division. Modular division is common to many programming languages

Promotes-

Cast operation- converts data from one primitive data type to another by entering the new data type in parentheses before a literal or variable

Comparison operators- involve two values, as do arithmetic operators; however, they compare the numbers rather than perform math on them

Relational operators- they compare the relation of two values; the last two sometimes called *equality operators*

Expression- can perform a calculation, manipulate characters, call a method, or test data; expressions can be divided into two basic categories: numeric and conditional

Numeric expression- any expression that can be evaluated as a number

Conditional expression- any expression that can be evaluated as true or false

Nested- when parentheses are contained within other parentheses

Exponentiation- the process of raising a number to the power of an exponent

Math class- part of the java.lang package, contains methods for a number of useful functions, such as rounding, exponentiation, randomizing, and square roots

pow() method- used to express exponentiation as in meters squared, requires two arguments: the base number and the exponent; a comma separates the arguments

Swing components- common name for a new set of GUI components introduced by Sun Microsystems in 1997

JOptionPane- a class used to display standard dialog boxes; the class provides several methods to create and display dialog boxes that prompt users for an input value, that prompts users to confirm an action, or that displays messages

Constant- a value that Java understands to have a certain, intrinsic meaning

exit() method- used to terminate an application that displays a graphical user interface, such as a dialogue box; method accepts an integer argument that serves as code

java.awt.event package- provides interfaces and classes for dealing with different types of events triggered by AWT components

Listener interfaces- sometimes called simply an *interface*- monitors, or listens, for events during execution of an interactive program

ActionListener- The listener interface used in the applet version of the Body Mass Index Calculator; ActionListener is a listener interface that listens for any events that occur during execution of the program, such as when a user clicks a button, double-clicks an item, selects a menu item, or presses the *Enter* key

Implements- the keyword used in the class header to specify which listener interface a programmer wants to use; if more than one listener interface is implemented, those are separated by commas

Component- sometimes called a control, is a Java class with methods to construct, add, and manipulate the object

Label- an object that displays text in the applet window

TextField- an object that displays a text box in which users enter text

Button- an object that displays a command button for users to click

Naming convention- the way you use words, cases, prefixes, and underscores to name the identifiers in your program

Coding convention- rules followed by specific programmers, such as indenting related lines of code

Defined- or coded

setForeground() method- changes the foreground or text color

add() method- used to insert the previously declared objects, such as companyLabel and heightField, in the applet window

addActionListener() method- registers, or assigns, an ActionListener to receive action events from that component

event source- when an object such as the button causes an event to happen, the object is called the event source

registered- or paired

event handlers- methods in a listener interface that specify what will happen when an event is sent to the listener interface

actionPerformed()- event handler, or method, for ActionListener

getText() method- used to retrieve text from a Label, TextField, or other AWT or Swing component that uses text

setText() method- does just the opposite of the getText() method: it assigns the caption or String to an object

**Chapter 4:**

selection structure- aka an if…else structure; branches to a certain section of code

Repetition structure- repeats a certain section of code

Control structures- both the selection and repetition structure; the logic of control structures controls the order in which code statements execute

Stub- an incomplete portion of code entered to allow the developer to compile and test the program

Modularity- a characteristic of a program in which a larger program’s source code is broken down into smaller sections, or modules, of source code; in Java, these modules include methods and classes

Call- a line of code stating the name of the method, followed by any data needed by the method in the form of arguments enclosed in parentheses

Static- a method modifier that indicates that the getSales() method is unique and can be invoked without creating a subclass or instance

Return statement- indicates to the JVM that the method is finished and that execution may return to the main() method

If…else statement- used to perform selection or make a decision on whether to execute a particular piece of code based on the evaluation of a condition

Condition- a Boolean expression that evaluates to true or false

Single-line if statement- used to perform a single task when the condition in the statement is true

Block if statement- used to execute more than one command if the condition in the statement is true

Nested- or completely included, within another line of code

Logical operator- used to connect two conditional expressions

AND operator (&&)- connects two expressions, x and y, so that both conditions individually must be evaluated as true for the entire expression, x && y, to be evaluated as true

OR operator (||)- connects two expressions, x and y, so that the whole expression, x || y, evaluates to true, or if they both do

NOT operator (!)- connects two expressions, x and y, so that if x evaluates to true, then the expression !x evaluates to false, and vice versa

Null- a constant that represents the presence of no data

Exception- aka run-time exception; a Java event resulting from an unusual or erroneous situation which disrupts the normal program flow of instructions

Exception handling- the general concept of planning for possible exceptions by directing the program to deal with them gracefully without terminating prematurely

Checked exception- one in which the compiler checks each method during compilation to ensure that each method has a *handler*- the code used to address any possible exceptions

Claims- when a method lets the compiler know that it may pass along an exception rather than handling it

Try statement- identifies a block of statements that potentially may throw an exception

Throw statement- transfers execution from the method that caused the exception to the handler that addresses any possible exceptions

Catch statement- consists of the keyword, catch, followed by a parameter declaration that identifies the type of exception being caught and an identifier name in parentheses

NumberFormatException- indicates an operation attempted to use a number in an illegal format

Finally statement- typically contains code to perform any cleanup that might be necessary after executing the try statement and catch statement

Validity- checking validity involves testing data to ensure that it uses the correct data type

Reasonableness- that is, that the values entered are within reason as expected input

Looping- when Java uses a repetition structure to repeat a certain section of code

While loop- a special repetition structure that Java uses for looping when the exact number of repetitions is unknown

While statement- starting with the keyword, while, followed by a condition in parentheses

Case structure- a type of selection structure that allows for more than two choices when the condition is evaluated

Switch statement- evaluates an integer expression or value and then conditionally perform statements; the switch statement evaluates its value and then, depending on the value, transfers control to the appropriate case statement

Case statement-

Break statement- forces an exit of the structure when a match is found; after the break, no more statements within the structure are evaluated, thereby reducing processing time

Formatted numeric output- includes features such as dollar signs, commas, decimal points, leading zeroes, and other formatting symbols, applied automatically to a displayed value

DecimalFormat class- formats decimal numbers

DecimalFormat() method- the argument for the DecimalFormat() method is a String called a *pattern-* which determines how the formatted number should be displayed

Format() method- assigns the formatting value to a specific value

Printf() method- used to format numeric output

Conversion characters- inserts data into a string

ItemListener- listens for when the user clicks components such as check boxes

Color() method- takes 3 arguments, each of which is a number in the range from 0 to 255 that corresponds to a specific red, green, and blue color

Traditional- displays as a small square with a caption

CheckboxGroup- used to group together several Checkbox components

Caption- label

State- true or false

addItemListener event- causes the applet to listen for clicks initiated by the user

setForeground() method- changes the color of the text used in the applet window

**Chapter 5:**

Object has 3 characteristics:

1. identity- means the object can be called and used as a single unit

2. state- refers to the various properties of the object, whose values might change

3. behavior- means the object can perform actions and can have actions performed on it

array- stores multiple data items of the same data type in a single storage location

TextArea component- appears as a rectangular box in the interface, displays a larger amount of data than a textField

Choice component- displays as a drop-down list in a graphical user interface, has methods for creating a list of data items and methods to retrieve and reset the user’s selection from the list

Frame- an AWT component that serves as a container for a collection of graphical AWT components

Panel- an AWT component that serves as an invisible container to further refine the arrangement of components within another container, such as a Frame

Constructor method- a programmer-defined method that is called when creating an object to ensure that all associated variables are initialized properly

External class- a class or method that is not a driver class

Index number- aka, a subscript, is assigned to each element of the array, allowing the program and the programmer to access individual values when necessary

Zero-indexing- begin with zero and progress to sequentially by whole numbers to the end of an array

Method overloading- the practice of defining more than one method with the same name

Counter-controlled loop- aka measured loop, a loop that executes a specific number of times

For statement- aka for loop, lists the parameters of a for loop, which include the beginning value, the stop condition that tests whether to continue to loop, and the increment to use for the counter

Assignment operator- aka shortcut operator, is used to perform arithmetic and an assignment operation all within one operator, thus providing a shortened form of that variable accumulation

Add and assign operator(+=)- performs both addition of a new value and assignment to a storage location, replacing the repetitive portion of the above code with the following code: variable += newValue;

Unary operator- an operator that needs only one value, or operand, to perform its function

Increment operator (++)- adds one to the operand

Decrement operator (--)- subtracts one from the operand

Instance method- operates or manipulates variables for an external class

Instance variables- the variables manipulated within an instance method, which are local in scope

Container- a special java object that contains other components, such as user interface controls and other containers

Frame-based applications- aka windowed applications, programs that use a window, or GUI-based interface, typically extend the Frame class and are called windowed applications

Object array- when an array contains values other than primitive data types, such as a reference data type or String, aka control array

Layout manager- one of a set of five classes that help programmers organize components into predefined locations in the window, rather than allowing Java to place them left to right within the space provided

FlowLayout- the default, or preset, layout manager for Panels and Applets

Constant- such as LEFT, RIGHT, or CENTER, is entered in capital letters

BorderLayout- places components into five regions within the container: North, South, East, West, and Center

GridLayout- divides the container into a grid so that components can be placed in rows and columns, from left to right and then top to bottom, within the grid

CardLayout- used primarily to house other containers

GridBagLayout- created in a manner similar to GridLayout, aligns components horizontally and vertically without requiring that the components be the same size

setEditable()- locks each TextArea component, which means the user cannot enter text into the TextArea at run time

populate- insert data into, the drop-down list of the Choice component

valueOf() method- returns a String value from its int argument, effectively converting the int to a String

addWindowListener() method- registers the listener with the Frame

register- connect two objects so that events from one object, the Frame, are sent to the other object, the listener

Adapter classes- provide prewritten methods for interfaces

Select() method- resets the clear Choice components

setEditable() method- limits user input to button clicks

**Chapter 6:**

Private- a modifier used to ensure that the driver class cannot change a variable inadvertently

Class scope- means that these component and instance variables are not accessible outside the class

Java.awt.datatransfer package- facilitate moving data in and out of the system clipboard

MenuBar- used to construct an instance of a menu bar

Menu- creates a new menu command on the menu bar using the specified label using the Menu() constructor method

Tear-off menu- indicates whether or not the menu can be implemented

MenuItem- adds a command to a previously declared Menu component, thus resulting in a new command on the drop-down menu

setMenuBar() method- assigns the MenuBar to the frame

setActionCommand() method- takes a String argument that becomes a reference that can be checked with an if statement

remove() method- deletes an item from the menu

insertSeparator()- inserts a horizontal separator line at the index position

composite component- a container with added components

clipboard- a portion of temporary memory reserved for user storage

Clipboard class- implements the system clipboard to transfer data in and out of the clip board; clipboard class is used by programmers to declare a non-primitive variable that will store the clipboard contents

Toolkit class- the superclass of all actual implementations of the Abstract Window Toolkit; used to bind the various components to particular **native**, or system-dependent, implementations, such as clipboard

Transferable interface- allows for methods that can be used to provide data for a transferable operation

DataFlavor- an argument that encapsulates information about specific data formats

stringFlavor- a static variable in Java representing any generic string of characters

select- when the mouse is dragged across the data, or the SHIFT+ARROW key , to highlight the text

StringSelection class- allows programmers to construct an instance of a StringSelection object

setContents() method- transfers the data to the clipboard

getActionCommand() method- retrieves the keyword

getSource() method- used to compare against a component’s variable name; is a method associated with ActionEvent objects

Throwable- the superclass of all errors and exceptions in the Java language, used in this case to generically catch any exception that might occur; only objects that are instances of this class (or one of its subclasses) are thrown by the Java Virtual Machine or can be thrown by the Java throw statement; similarly, only this class or one of its subclasses can be the argument type in a catch clause

setIconImage() method- assigns the image to the Frame’s title bar icon

**Chapter 11:**

Relational database- maintains data in tables, allowing relationships between the tables to be specified, and typically is controlled by a vendor-specific software called a database management system (DBMS)

Database table- provides a grouping of related data in terms of rows and columns, much as with any table of data, even on a printed page

JDBC- provides an API for database-independent connectivity between the Java programming language and a wide range of tabular data sources, including relational databases

Structured Query Language (SQL)- statements issued to manipulate the database

Normalized- when the condition of having no repeating groups is satisfied, the data is normalized

First normal form- when additional rules of normalization can be applied to database design

Relationship- an association between fields (columns) in the two tables

Cardinality- the cardinality of a relationship is described as one-to-one, one-to-many, or many-to-many, depending on the number of records in a table that may relate to a given record in another table

Primary key- an index with a unique, non-null value for each record

Foreign key- one or more fields in a table that reference the primary key fields of another table

**Chapter 12:**

Web-enabled- they utilize a connection to a Web server and can be accessed from anywhere in the world

Static Web page- a Web page that exists in its entirety on the Web server before it is used

Dynamic Web page- a Web page that is created as needed and usually is customized upon each use; dynamic Web pages often contain data extracted from a database

MVC- Model-view-controller

3 parts

model- defines the business rules layer

view- defines the user presentation layer

controller- defines the application layer responsible for managing the application flow