

## 1

# Introduction to Computers, the Internet and the World Wide Web



*Our life is frittered away by detail.... Simplify, simplify.*

— Henry David Thoreau

*The chief merit of language is clearness.*

— Galen

*My object all sublime I shall achieve in time.*

— W. S. Gilbert

*He had a wonderful talent for packing thought close, and rendering it portable.*

— Thomas B. Macaulay

*“Egad, I think the interpreter is the hardest to be understood of the two!”*

— Richard Brinsley Sheridan

*Man is still the most extraordinary computer of all.*

— John F. Kennedy



# OBJECTIVES

In this chapter you will learn:

- Basic computer hardware and software concepts.
- Basic object technology concepts, such as classes, objects, attributes, behaviors, encapsulation, inheritance and polymorphism.
- The different types of programming languages.
- Which programming languages are most widely used.
- A typical Java development environment.
- Java's role in developing distributed client/server applications for the Internet and the web.
- The history of the UML—the industry-standard object-oriented design language.
- The history of the Internet and the World Wide Web.
- To test-drive Java applications.



# Outline

- 1.1 Introduction**
- 1.2 What Is a Computer?**
- 1.3 Computer Organization**
- 1.4 Early Operating Systems**
- 1.5 Personal, Distributed and Client/Server Computing**
- 1.6 The Internet and the World Wide Web**
- 1.7 Machine Languages, Assembly Languages and High-Level Languages**
- 1.8 History of C and C++**
- 1.9 History of Java**
- 1.10 Java Class Libraries**



# Outline

- 1.11 Fortran, COBOL, Pascal and Ada**
- 1.12 BASIC, Visual Basic, Visual C++, C# and .NET**
- 1.13 Typical Java Development Environment**
- 1.14 Notes about Java and *Java How to Program, Seventh Edition***
- 1.15 Test-Driving a Java Application**
- 1.16 Software Engineering Case Study: Introduction to Object Technology and the UML**
- 1.17 Web 2.0**
- 1.18 Software Technologies**
- 1.19 Wrap-Up**
- 1.20 Web Resources**



# 1.1 Introduction

- **Java Standard Edition (Java SE) 6**
- **Sun's implementation called the Java Development Kit (JDK)**
- **Object-Oriented Programming**
- **Java is language of choice for networked applications**
- **Java Enterprise Edition (Java EE) geared toward large-scale distributed applications and web applications**
- **Java Micro Edition (Java ME) geared toward applications for small, memory constrained devices**



## 1.2 What Is a Computer?

- **Computer**
  - Performs computations and makes logical decisions
  - Millions or billions of times faster than human beings
- **Computer programs**
  - Sets of instructions for which computer processes data
- **Hardware**
  - Physical devices of computer system
- **Software**
  - Programs that run on computers



# 1.3 Computer Organization

- **Six logical units of computer system**
  - **Input unit**
    - Mouse, keyboard
  - **Output unit**
    - Printer, monitor, audio speakers
  - **Memory unit**
    - Retains input and processed information
  - **Arithmetic and logic unit (ALU)**
    - Performs calculations
  - **Central processing unit (CPU)**
    - Supervises operation of other devices
  - **Secondary storage unit**
    - Hard drives, floppy drives





# 1.4 Early Operating Systems

- **Batch processing**
  - One job (task) at a time
  - Operating systems
    - Developed to make computers more convenient to use
    - Made transitions between jobs easier
    - More throughput
- **Multiprogramming**
  - “Simultaneous” jobs
  - Timesharing operating systems



# 1.5 Personal, Distributed and Client/Server Computing

- **Personal computing**
  - Computers for personal use
- **Distributed computing**
  - Networked computers
  - Computing performed among several computers
- **Client/server computing**
  - Servers offer common store of programs and data
  - Clients access programs and data from server



## 1.6 The Internet and the World Wide Web

- **Internet**
  - **Developed more than four decades ago with DOD funding**
  - **Originally for connecting few main computer systems**
  - **Now accessible by over a billion computers**
- **World Wide Web (WWW)**
  - **Allows for locating/viewing multimedia-based documents**



# 1.7 Machine Languages, Assembly Languages and High-Level Languages

- **Machine language**
  - “Natural language” of computer component
  - Machine dependent
- **Assembly language**
  - English-like abbreviations represent computer operations
  - Translator programs (assemblers) convert to machine language
- **High-level language**
  - Allows for writing more “English-like” instructions
    - Contains commonly used mathematical operations
  - Compiler converts to machine language
- **Interpreter**
  - Execute high-level language programs without compilation



## 1.8 History of C and C++

- **C++ evolved from C, which evolved from BCPL and B**
- **C**
  - **Developed at Bell Labs**
  - **Popularized as the language of the UNIX operating system**
- **C++**
  - **Developed by Bjarne Stroustrup**
  - **Provides object-oriented programming capabilities**
  - **Hybrid language**
- **Objects**
  - **Reusable software components that model real-world items**
  - **Attributes and behaviors**



## 1.9 History of Java

- **Java**
  - **Originally for intelligent consumer-electronic devices**
  - **Then used for creating web pages with dynamic content**
  - **Now also used to:**
    - **Develop large-scale enterprise applications**
    - **Enhance web server functionality**
    - **Provide applications for consumer devices (cell phones, etc.)**



## 1.10 Java Class Libraries

- **Java programs consist of classes**
  - Include methods that perform tasks
    - Return information after task completion
- **Java provides class libraries**
  - Known as Java APIs (Application Programming Interfaces)
- **To use Java effectively, you must know**
  - Java programming language
  - Extensive class libraries



## Software Engineering Observation 1.1

---

**Use a building-block approach to create programs. Avoid reinventing the wheel—use existing pieces wherever possible. Called *software reuse*, this practice is central to object-oriented programming.**





## Software Engineering Observation 1.2

---

**When programming in Java, you will typically use the following building blocks: Classes and methods from class libraries, classes and methods you create yourself and classes and methods that others create and make available to you.**



## Performance Tip 1.1

---

**Using Java API classes and methods instead of writing your own versions can improve program performance, because they are carefully written to perform efficiently. This technique also shortens program development time.**



## Portability Tip 1.1

---

**Using classes and methods from the Java API instead of writing your own improves program portability, because they are included in every Java implementation.**



## Software Engineering Observation 1.3

---

**Extensive class libraries of reusable software components are available over the Internet and the web, many at no charge.**



## 1.11 FORTRAN, COBOL, Pascal and Ada

- **Fortran**
  - **FORmula TRANslator**
  - **Developed by IBM for scientific and engineering applications**
- **COBOL**
  - **COmmon Business Oriented Language**
  - **Used for commercial applications requiring precise/efficient manipulation of large amounts of data**
- **Pascal**
  - **Developed by Prof. Niklaus Wirth**
  - **Designed to teach structured programming**
- **Ada**
  - **Developed under the sponsorship of the U.S. Department of Defense**
  - **Needed a single language to fill most of its needs**
  - **Provided multitasking so programmers could specify parallel tasks**



# 1.12 BASIC, Visual Basic, Visual C++, C# and .NET

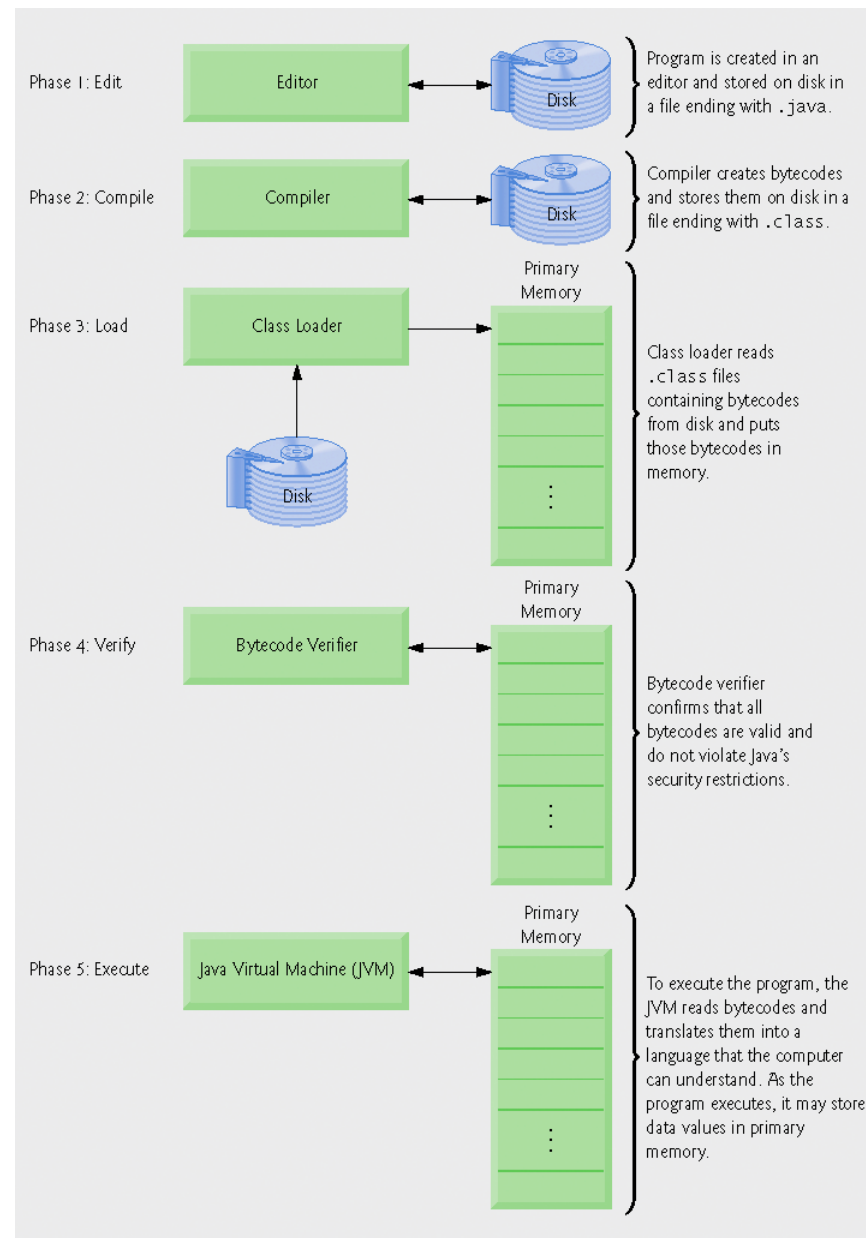
- **BASIC**
  - **Beginner's All-Purpose Symbolic Instruction Code**
  - **Developed at Dartmouth College to familiarize novices with programming techniques**
- **Visual Basic introduced by Microsoft in the early 1990s**
- **.NET platform**
  - **Part of Microsoft's corporate-wide strategy to incorporate the Internet and web into computer applications**
- **Three primary .NET programming languages**
  - **Visual Basic .NET (based on BASIC)**
  - **Visual C++ .NET (based on C++)**
  - **C# (based on C++ and Java)**



# 1.13 Typical Java Development Environment

- **Java programs go through five phases**
  - **Edit**
    - Programmer writes program using an editor; stores program on disk with the `.java` file name extension
  - **Compile**
    - Use `javac` (the Java compiler) to create bytecodes from source code program; bytecodes stored in `.class` files
  - **Load**
    - Class loader reads bytecodes from `.class` files into memory
  - **Verify**
    - Bytecode verifier examines bytecodes to ensure that they are valid and do not violate security restrictions
  - **Execute**
    - Java Virtual Machine (JVM) uses a combination of interpretation and just-in-time compilation to translate bytecodes into machine language





**Fig. 1.1 | Typical Java development environment.**





# Common Programming Error 1.1

---

**Errors like division by zero occur as a program runs, so they are called *runtime errors* or *execution-time errors*. *Fatal runtime errors* cause programs to terminate immediately without having successfully performed their jobs. *Nonfatal runtime errors* allow programs to run to completion, often producing incorrect results.**

---



# 1.14 Notes about Java and Java How to Program, Seventh Edition

- **Stresses clarity**
- **Portability**
  - **An elusive goal due to differences between compilers, JVMs and computers**
  - **Always test programs on all systems on which the programs should run**



# Good Programming Practice 1.1

---

**Write your Java programs in a simple and straightforward manner. This is sometimes referred to as KIS (“keep it simple”). Do not “stretch” the language by trying bizarre usages.**



## Portability Tip 1.2

---

**Although it is easier to write portable programs in Java than in other programming languages, differences between compilers, JVMs and computers can make portability difficult to achieve. Simply writing programs in Java does not guarantee portability.**



## Error-Prevention Tip 1.1

---

**Always test your Java programs on all systems on which you intend to run them, to ensure that they will work correctly for their intended audiences.**



## Good Programming Practice 1.2

---

**Read the documentation for the version of Java you are using. Refer to it frequently to be sure you are aware of the rich collection of Java features and are using them correctly.**



## Good Programming Practice 1.3

---

**Your computer and compiler are good teachers. If, after carefully reading your Java documentation manual, you are not sure how a feature of Java works, experiment and see what happens. Study each error or warning message you get when you compile your programs (called *compile-time errors* or *compilation errors*), and correct the programs to eliminate these messages.**

---



## Software Engineering Observation 1.4

---

**Some programmers like to read the source code for the Java API classes to determine how the classes work and to learn additional programming techniques.**





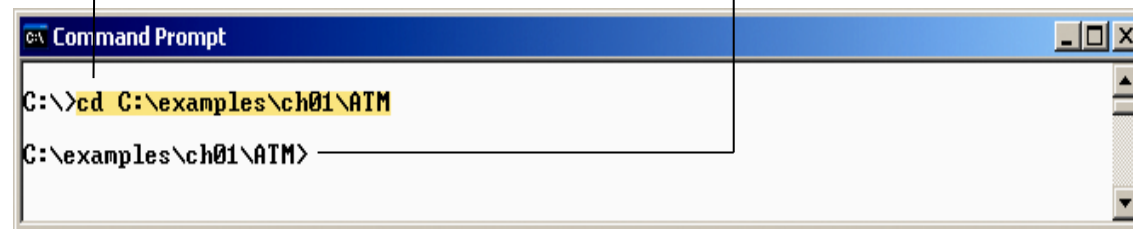
# 1.15 Test-Driving a Java Application

- **Test-driving the ATM application**
  - Check system setup
  - Locate the ATM application (Fig. 1.2)
  - Run the ATM application (Fig. 1.3)
  - Enter an account number (Fig. 1.4)
  - Enter a PIN (Fig. 1.5)
  - View the account balance (Fig. 1.6)
  - Withdraw money from the account (Fig. 1.7)
  - Confirm that the account information has been updated (Fig. 1.8)
  - End the transaction (Fig. 1.9)
  - Exit the ATM application
- **Additional applications (Fig. 1.10)**



Using the cd command to  
change directories

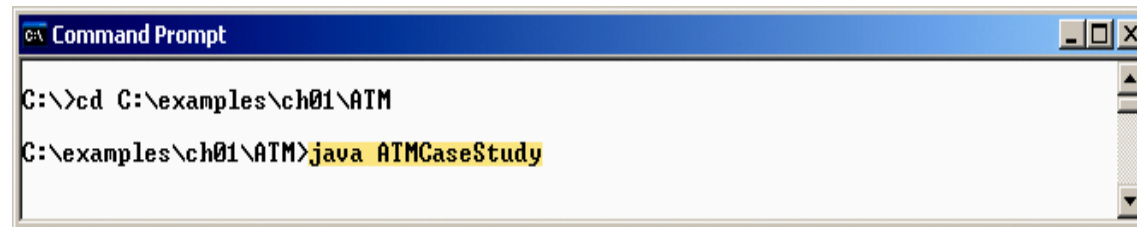
File location of the ATM application



The screenshot shows a Windows XP Command Prompt window with a blue title bar that says "C:\ Command Prompt". The command prompt text area shows the command `C:\>cd C:\examples\ch01\ATM` where the `cd` command is highlighted in yellow. Below the command, the prompt has changed to `C:\examples\ch01\ATM>`. Two lines with arrows point from the text above to the screenshot: one points to the `cd` command, and the other points to the path `C:\examples\ch01\ATM`.

**Fig. 1.2 | Opening a Windows XP *Command Prompt* and changing directories.**





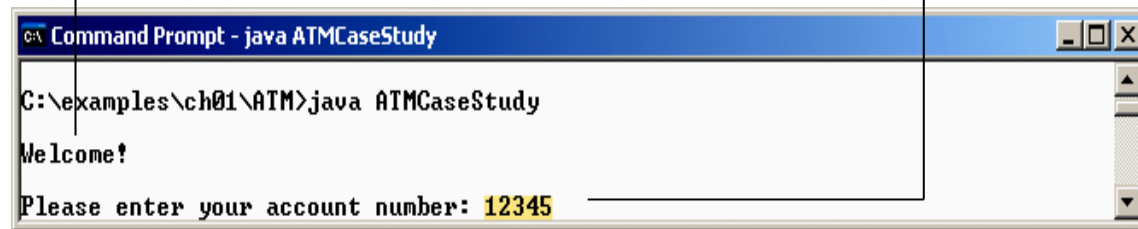
```
C:\ Command Prompt
C:\>cd C:\examples\ch01\ATM
C:\examples\ch01\ATM>java ATMCaseStudy
```

**Fig. 1.3 | Using the java command to execute the ATM application.**



ATM welcome message

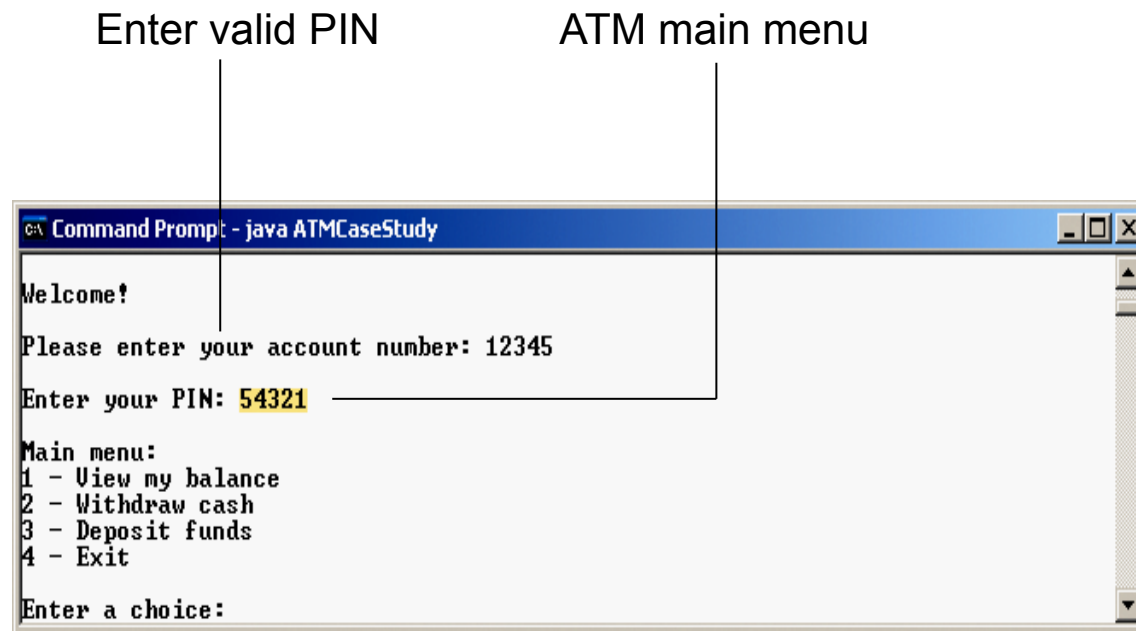
Enter account number prompt



```
Command Prompt - java ATMCaseStudy
C:\examples\ch01\ATM>java ATMCaseStudy
Welcome!
Please enter your account number: 12345
```

**Fig. 1.4 | Prompting the user for an account number.**

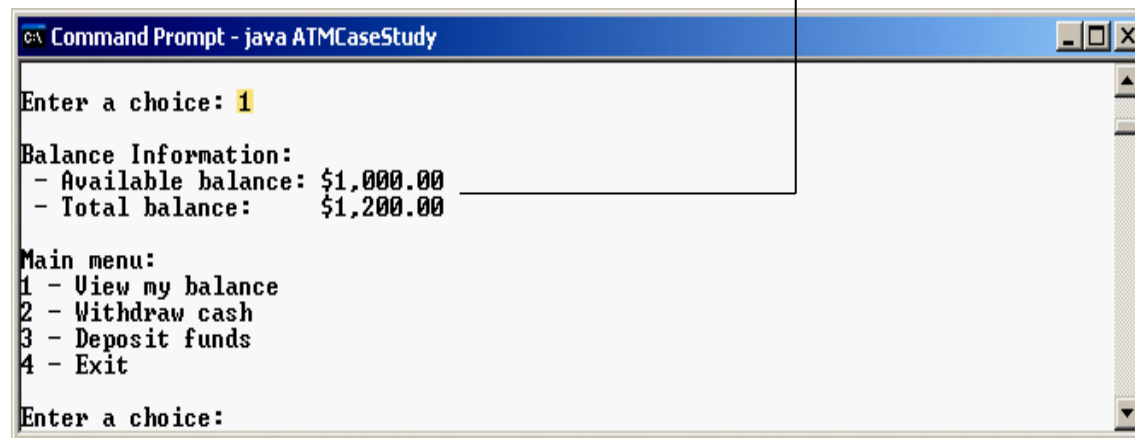




**Fig. 1.5 | Entering a valid PIN number and displaying the ATM application's main menu.**



Account balance information



```
C:\> Command Prompt - java ATMCaseStudy

Enter a choice: 1
Balance Information:
- Available balance: $1,000.00
- Total balance: $1,200.00

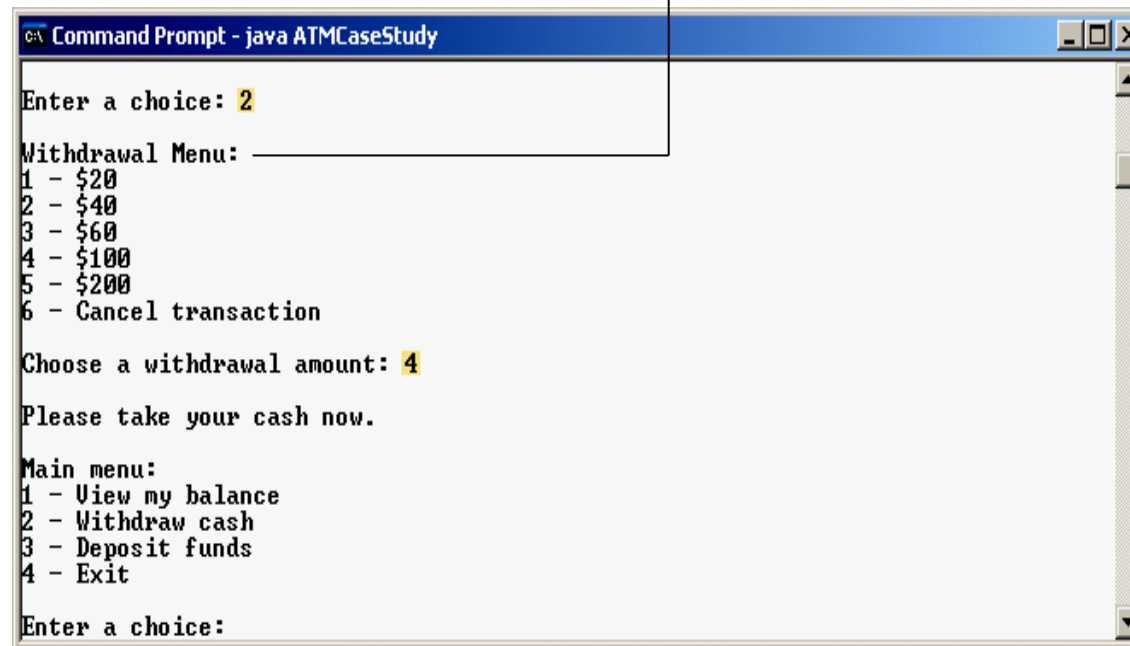
Main menu:
1 - View my balance
2 - Withdraw cash
3 - Deposit funds
4 - Exit

Enter a choice:
```

**Fig. 1.6 | ATM application displaying user account balance information.**



ATM withdrawal menu



```
CA Command Prompt - java ATMCaseStudy

Enter a choice: 2

Withdrawal Menu:
1 - $20
2 - $40
3 - $60
4 - $100
5 - $200
6 - Cancel transaction

Choose a withdrawal amount: 4

Please take your cash now.

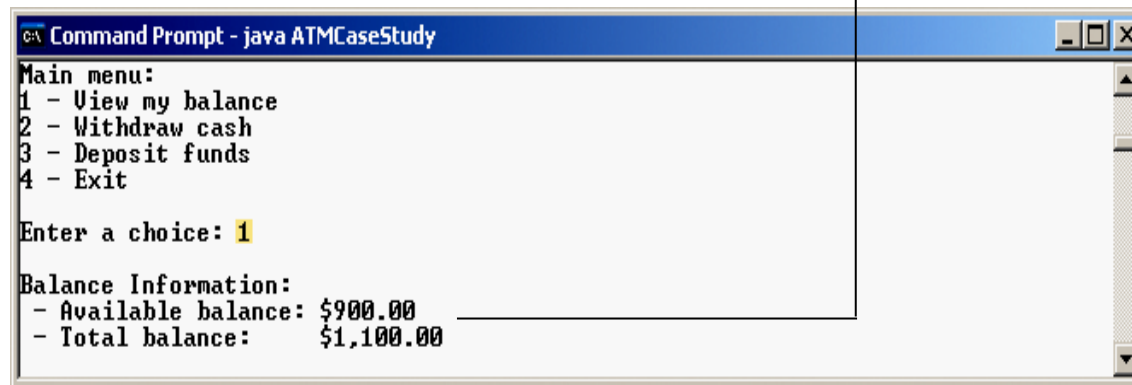
Main menu:
1 - View my balance
2 - Withdraw cash
3 - Deposit funds
4 - Exit

Enter a choice:
```

**Fig. 1.7 |** Withdrawing money from the account and returning to the main menu.



Confirming updated account balance  
information after withdrawal transaction



```
C:\> Command Prompt - java ATMCaseStudy
Main menu:
1 - View my balance
2 - Withdraw cash
3 - Deposit funds
4 - Exit

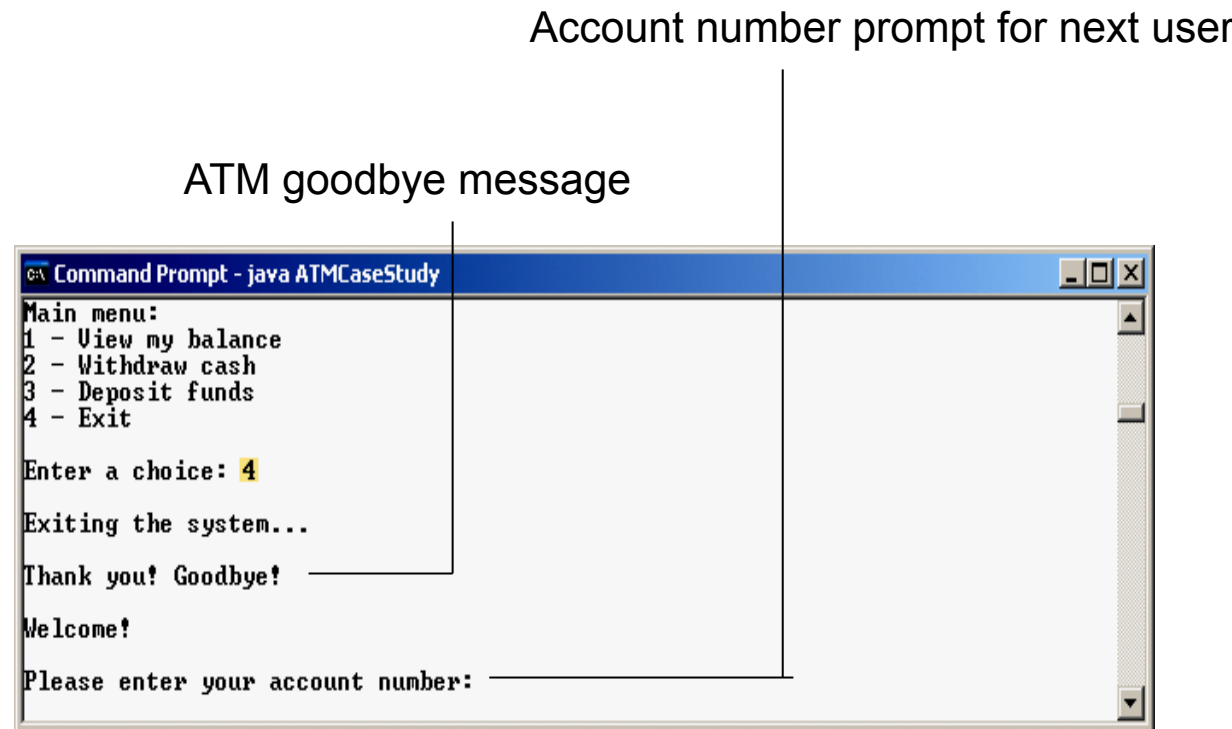
Enter a choice: 1

Balance Information:
- Available balance: $900.00
- Total balance:    $1,100.00
```

**Fig. 1.8 | Checking new balance.**







**Fig. 1.9 | Ending an ATM transaction session.**



Application Name	Chapter Location	Commands to Run
Tic-Tac-Toe	Chapters 8 and 24	<code>cd C:\examples\ch01\Tic-Tac-Toe</code> <code>java TicTacToeTest</code>
Guessing Game	Chapter 11	<code>cd C:\examples\ch01\GuessGame</code> <code>java GuessGame</code>
Logo Animator	Chapter 21	<code>cd C:\examples\ch01\LogoAnimator</code> <code>java LogoAnimator</code>
Bouncing Ball	Chapter 23	<code>cd C:\examples\ch01\BouncingBall</code> <code>java BouncingBall</code>

**Fig. 1.10 | Examples of additional Java applications found in *Java How to Program*, 6/e.**



## 1.16 Software Engineering Case Study: Introduction to Object Technology and the UML (Required)

- **Object orientation**
- **Unified Modeling Language (UML)**
  - Graphical language that uses common notation
  - Allows developers to represent object-oriented designs



# 1.16 Software Engineering Case Study (Cont.)

- **Objects**
  - **Reusable software components that model real-world items**
  - **Look all around you**
    - **People, animals, plants, cars, etc.**
  - **Attributes**
    - **Size, shape, color, weight, etc.**
  - **Behaviors**
    - **Babies cry, crawl, sleep, etc.**



# 1.16 Software Engineering Case Study (Cont.)

- **Object-oriented design (OOD)**
  - Models software in terms similar to those used to describe real-world objects
  - Class relationships
  - Inheritance relationships
  - Models communication among objects
  - Encapsulates attributes and operations (behaviors)
    - Information hiding
    - Communication through well-defined interfaces
- **Object-oriented language**
  - Programming in object-oriented languages is called *object-oriented programming (OOP)*
  - Java



## 1.16 Software Engineering Case Study (Cont.)

- **Classes are to objects as blueprints are to houses**
- **Associations**
  - Relationships between classes
- **Packaging software in classes facilitates reuse**



# 1.16 Software Engineering Case Study (Cont.)

- **Object-Oriented Analysis and Design (OOA/D)**
  - **Essential for large programs**
  - **Analyze program requirements, then develop a design**
  - **UML**
    - **Unified Modeling Language**
    - **Standard for designing object-oriented systems**



# 1.16 Software Engineering Case Study (Cont.)

- **History of the UML**
  - **Need developed for process with which to approach OOA/D**
  - **Brainchild of Booch, Rumbaugh and Jacobson**
  - **Object Management Group (OMG) supervised**
  - **Version 2 is current version**





# 1.16 Software Engineering Case Study (Cont.)

- **UML**
  - **Graphical representation scheme**
  - **Enables developers to model object-oriented systems**
  - **Flexible and extensible**



## 1.17 Web 2.0

- **Web use exploded in mid-to-late 1990s**
- **Dot com economic bust hit in the early 2000s**
- **Resurgence in 2004 with Web 2.0**
  - **First Web 2.0 conference**
  - **Google widely regarded as signature company of Web 2.0**
  - **Web services enable Web 2.0**



## 1.17 Web 2.0 (cont.)

- **Ajax**
  - **Term popularized in 2005**
  - **Group of technologies and programming techniques in use since the late 1990s**
  - **Helps Internet-based applications perform like desktop application**



## 1.17 Web 2.0 (cont.)

- **Blogs**

- Websites that are like online diaries
- About 60 million of them at the time of this writing
- Blogosphere—Collection of all blogs and the blogging community
- Technorati—a leading blog search engine

- **RSS feeds**

- Enable sites to push information to subscribers
- Commonly used to deliver blog postings



## 1.17 Web 2.0 (cont.)

- **Web 3.0**
  - **Next generation web**
  - **Also known as the semantic web**
  - **Web 1.0 mostly HTML based**
  - **Web 2.0 making increasing use of XML (e.g., RSS feeds)**
  - **Web 3.0 will make extensive use of XML to add meaning to content**



# 1.18 Software Technologies

- **Agile Software Development**
  - Methodologies for developing software quickly with fewer resources
- **Extreme Programming (XP)**
  - One of many agile development methodologies
  - Release software frequently in small increments to encourage user feedback
  - Programmers work in pairs at one computer
  - Immediate code review
  - All team programmers able to work on any part of code



## 1.18 Software Technologies (cont.)

- **Refactoring**
  - Reworking code to make cleaner/easier to maintain
  - Widely used in agile development methodologies
  - Many tools available
- **Design patterns**
  - Proven architectures for constructing flexible/maintainable object-oriented software



## 1.18 Software Technologies (cont.)

- **Game programming**
  - **Game business is larger than first run movie business**
  - **College courses and majors now devoted to sophisticated game-programming techniques**





## 1.18 Software Technologies (cont.)

- **Open source software**
  - **Individuals and companies contribute to developing, maintaining and evolving software in exchange for the right to use that software for their own purposes, typically at no charge**
  - **Code typically scrutinized by much larger audiences, so bugs get removed faster**
  - **Java now open source**
  - **Some open source organizations**
    - **Eclipse Foundation, Mozilla Foundation, Apache Software Foundation, SourceForge**



## 1.18 Software Technologies (cont.)

- **Linux**
  - **Open source operating system**
  - **One of the greatest successes of the open source movement**
- **MySQL**
  - **an open source database management system**
- **PHP**
  - **most popular open source server-side Internet “scripting” language for developing Internet-based applications**
- **LAMP—Linux, Apache, MySQL and PHP (or Perl or Python)**
  - **An acronym for the set of open source technologies that many developers used to build web applications**



## 1.18 Software Technologies (cont.)

- **Ruby on Rails**

- **Combines the scripting language Ruby with the Rails web application framework developed by 37Signals**
- **Many Ruby on Rails developers report significant productivity gains over using other languages when developing database-intensive web applications**
- ***Getting Real* ([gettingreal.37signals.com/toc.php](http://gettingreal.37signals.com/toc.php))**
  - **a must read for today's web application developers**



## 1.18 Software Technologies (cont.)

- **Software generally viewed as a product**
  - You buy a software package from a software vendor, then install that software on your computer and run it as needed
  - Upgrade process cumbersome and expensive
- **Software as a service (SAAS)**
  - Software runs on servers elsewhere on the Internet
  - When server is updated, all clients worldwide see the new capabilities
  - Access the software through a browser
  - Salesforce.com, Google and Microsoft's Office Live and Windows Live all offer SAAS.

