**Chapter 1 Section 1.3 – String Class**

* String methods:
  + str.length()
  + str1.equals(str2)
  + str1.equalsIgnoreCase(str2)
  + str.toLowerCase( )
  + str.toUpperCase( )
  + str.trim( )
  + str.charAt(integer)
  + str.substring(start)
  + str.substring(start, end)
  + str1.indexOf(str2)
  + str1.indexOf(str2, start)
  + str1.lastIndex(str2)
  + str1.compareTo(str2)
  + str1.compareToIgnoreCase(str2)
* lexicographic order (ASCII character set): digit, uppercase letter, lowercase letter

**Chapter 4 Defining Classes**

* Example creating a class: **BankAccount**

**1** /\*\*

**2**  A bank account has a balance that can be changed by

**3**  deposits and withdrawals.

**4** \*/

**5** public class **BankAccount** {

**6**

**7**  private double balance;

**8**

**9**  /\*\*

**10**  Constructs a bank account with a zero balance.

**11**  \*/

**12**  public BankAccount() {

**13**  balance = 0;

**14**  }

**15**

**16**  /\*\*

**17**  Deposits money into the bank account.

**18**  @param amount the amount to deposit

**19**  \*/

**20**  public void deposit(double amount) {

**21**  balance = balance + amount;

**22**  }

**23**

**24**  /\*\*

**25**  Withdraws money from the bank account.

**26**  @param amount the amount to withdraw

**27**  \*/

**28**

**29**  public void withdraw(double amount) {

**30**

**31**  balance = balance - amount;

**32**  }

**33**

**34**  /\*\*

**35**  Gets the current balance of the bank account.

**36**  @return the current balance

**37**  \*/

**38**  public double getBalance() {

**39**

**40**  return balance;

**41**  }

**42** }

* Driver (Testing) class for BankAccout: **BankAccountTester**

**1** /\*\*

**2**  A class to test the BankAccount class.

**3** \*/

**4** public class BankAccountTester {

**5**

**6** /\*\*

**7**  Tests the methods of the BankAccount class.

**8**  @param args not used

**9**  \*/

**10**  public static void main(String[] args) {

**11**

**12**  BankAccount harrysChecking = new BankAccount();

**13**  harrysChecking.deposit(2000);

**14**  harrysChecking.withdraw(500);

**15**  System.out.println(harrysChecking.getBalance());

**16**  System.out.println("Expected: 1500");

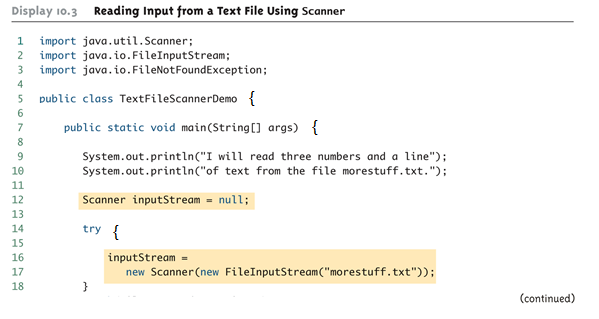
**17**  }

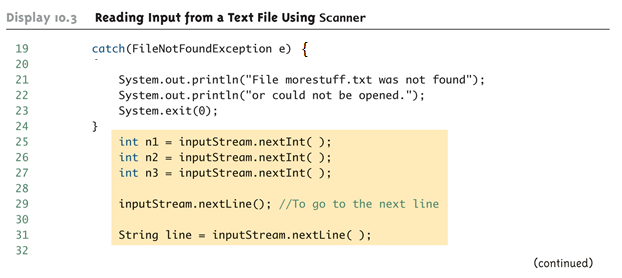
**18** }

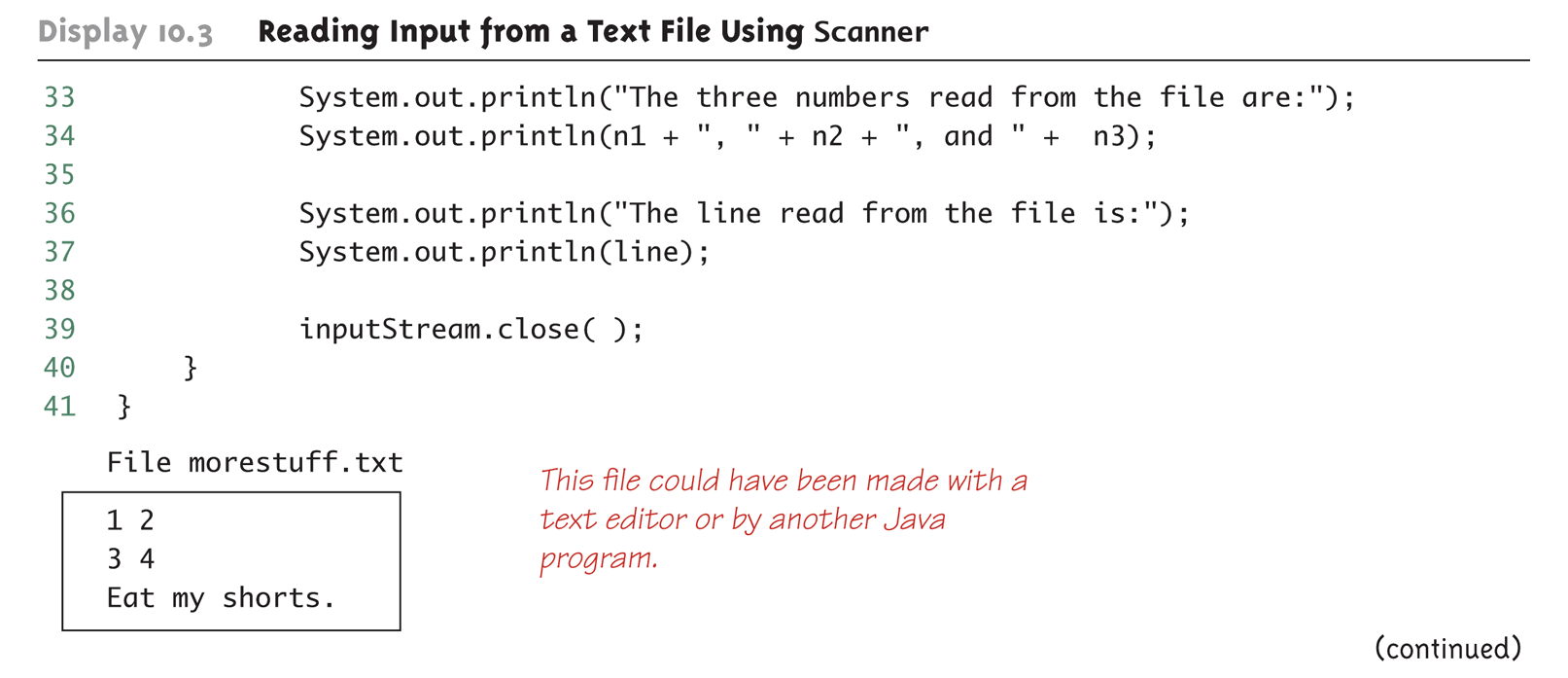
* Instance Variables
  + Automatically initialized
    - Boolean to false
    - Primitive type to zero,
    - Class types (Objects) to null (includes strings)
    - We prefer to explicitly initial them in a constructor
* Constructors
  + Called when an object is created of the class
  + Can be multiple constructors for a class
  + Can call other methods from constructor
  + Use this parameter when you need to refer to the “current” object
  + A no-argument is automatically created when you include no constructors

**Chapter 10 File I/O**

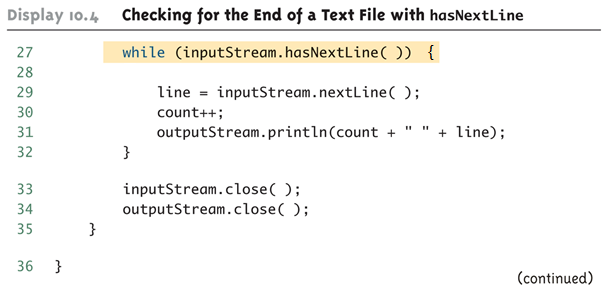
* Reading from a Text File







* Testing for end of File



**Chapter 6 Arrays**

* Declaring an array:
  + double[ ] score = new double[5]; OR
  + double[ ] score;

score = new double[5];

* + String str = new String[6];
  + BankAccount myChecking = new BankAccount[3];
* Initialize an array: int[ ] age = {2, 12, 1};
* Array instance variable: length
  + score.length returns the length (number of elements) of the array
* Array method sort: Arrays.sort(arrayName); java.util.Arrays
* Need to use for loop to initialize, update, or read an array

double[ ] score = new double[100];

for (int index = 0; index < score.length; index++)

score[index] = 0;

* Convert an array of char to String

char[ ] a = {'A', 'B', 'C'};

String s1 = new String(a); s1 = “ABC”

String s2 = new String(a,0,2); s2 = “AB”

* Both array indexed variables and entire arrays can be used as arguments to methods
* public void myMethod1(int [ ] score); myMethod1(score);

public void myMethod2(int score); myMethod2(score[2]);

* Making two arrays, a and b, equal:

for ( int i = 0; (i < a.length) && (i < b.length); i++ )

b[ i ] = a[ i ];

* Checking if two arrays are equal: (java.util.Arrays)

Arrays.equals(a, b);