1 Binary I/O

CST141

2 Binary Data

- **a**All data is stored in computers in binary format
 - All ones (1) and zeros (0)
- Text (including digits) is converted to some coding scheme (ASCII, Unicode, etc.)
- Binary does not require conversions—the binary numeric value is written directly to the file

3 InputStream and OutputStream

- These are the abstract classes from which all binary input and output classes extend
- All methods from all subclasses throw the checked IOException (or one of its subclasses) which must be caught
 - Must be imported from java.io

5 The FileInputStream Class (Page 1)

- Used for creating input streams that read only positive byte numeric data from a file
- A subclass of InputStream
 - Inherits the read() method that reads one byte of data to the stream
- Located in the java.io package import java.io.FileInputStream;

6 The FileInputStream Class (Page 2)

■Format:

<u>FileInputStream</u> *fileInputStreamObject* = new <u>FileInputStream("path| filename")</u>;

- ☐ Throws a java.io.FileNotFoundException if the file does not exist.
- Throws a java.io.DirectoryNotFoundException if the folder on the drivedoes not exist

FileInputStream input = new FileInputStream("temp.dat");

7 The FileOutputStream Class (Page 1)

- Used for creating output stream objects that write only positive byte numeric data to a file
- A subclass of OutputStream
 - Inherits the write() method that writes one byte of data to the stream
- Located in the java.io package import java.io.FileOutputStream;

8 ■ The FileOutputStream Class (Page 2)

■Format:

<u>FileOutputStream fileOutputStreamObject</u> = new <u>FileOutputStream("path/filename")</u>;

Examples:

FileOutputStream input = new FileOutputStream("temp.dat");

9 The read() Method

M method of class FileInputStream (inherited from FileStream) that reads the next byte

```
of data from the input stream—from the file)
      Format:
        fileInputStreamObject.read();
      Example:
        byte credits = input.read();
10 The write() Method

a method of class FileInputStream (inherited from FileStream) that writes a specified

       byte of data to the output stream—to the file)
      Format:
        fileOutputStreamObject.write( byteValue );
      Example:
        output.write(credits);
17 Filter Streams
      The FilterInputStream and FilterOutputStream classes filter streams filter bytes for
       some purpose
      Subclasses of these two classes read and write integers, floats, strings, characters and
       booleans
18 The DataInputStream Class (Page 1)
      "Wraps" around a FileInputStream object to give it the ability to read integers, floats,
       strings, booleans and characters
      A subclass of FilterInputStream
      Located in the java.io package
        import java.io.DataInputStream;
19 The DataInputStream Class (Page 2)
      Format:
        DataInputStream dataInputStreamObject = new
          DataInputStream(fileInputStreamObject);
      Example:
        DataInputStream input = new DataInputStream( new FileInputStream( "temp.dat") );
20 The DataOutputStream Class (Page 1)
      "Wraps" around a FileOutputStream object to give it the ability to write integers,
       floats, strings, booleans and characters
      A subclass of FilterOutputStream
      Located in the java.io package
        import java.io.DataOutputStream;
21 The DataOutputStream Class (Page 2)
      Format:
        DataOutputStream dataInputStreamObject = new
          DataOutputStream(fileOutputStreamObject);
      Example:
        DataOutputStream input = new DataOutputStream( new FileOutputStream(
          "temp.dat"));
22 Primitive Read Methods for Class DataInputStream
                                                                       (Page 1)
      Reads primitives including integers, floats, booleans and characters from the input
```

```
stream—from a file):
        - readByte()
        - readChar()
        - readFloat()
        - readDouble()
        - readInt()
        - readLong()
        - readShort()
        – readBoolean()
23 Primitive Read Methods for Class DataInputStream
                                                                        (Page 2)
      format:
        dataInputStreamObject.readPrimitive();
      Example:
        int credits = input.readInt();
24 The readUTF() Method

a method of class DataInputStream that reads a series of bytes from the input stream.

       that are in UTF-8 format and converts them into a string
      Format:
        dataInputStreamObject.readUTF();
      Example:
        String firstName = input.readUTF();
25 Primitive Write Methods for Class DataOutputStream
                                                                             (Page 1)
      Writes primitives including integers, floats, booleans and characters to the output
       stream—to a file):
        - writeByte()
        – writeChar()
        – writeFloat()
        – writeDouble()
        – writeInt()
        – writeLong()
        – writeShort()
        - writeBoolean()
26 Primitive Write Methods for Class DataOutputStream
                                                                             (Page 2)
      Format:
        dataOutputStreamObject.writePrimitive();
      Example:
        output.writeInt(credits);
27 The writeUTF() Method
      A method of class DataInputString that converts a series of bytes into a string (UTF-8)
       format) and writes them into the output stream
      Format:
        dataInputStreamObject.readUTF();
      Example:
        ouput.readUTF(firstName);
```

The BufferedInputStream Class (Page 1) Wraps" around a FileInputStream object to speed up input by reducing the number	of
disk reads	OI
The whole block of data is read into the RAM buffer at once	
40 The BufferedInputStream Class (Page 2)	
a Subclass of FilterInputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own—all are inherited from InputStream a It has no methods of its own and i	
all Located in the java.io package	
import java.io.BufferedInputStream;	
1 The BufferedInputStream Class (Page 3)	
Format:	
<u>BufferedInputStream</u> <u>bufferedInputStreamObject</u> = new <u>BufferedInputStream(</u> <u>fileInputStreamObject</u>);	
and the state of t	
BufferedInputStream input = new BufferedInputStream(new FileInputStream(
"temp.dat"));	
12 The BufferedInputStream Class (Page 4)	
DataInputStream dataInputStreamObject = new DataOutputStream(new	
<u>BufferedInputStream(</u> <i>fileInputStreamObject</i>));	
<pre> ① Example: </pre>	
DataInputStream input = new DataInputStream(new BufferedInputStream(new	
FileInputStream("temp.dat")));	
The BufferedOutputStream Class (Page 1)	
"Wraps" around a FileOutputStream object to speed up output by reducing the number of disk writes	er
The whole block or data first is written into the RAM buffer; when the buffer is full, the	ne.
block is written to disk	
14 The BufferedOutputStream Class (Page 2)	
A subclass of FilterOutputStream	
It has no methods of its own—all are inherited from OutputStream Located in the java.io package	
import java.io.BufferedOutputStream;	
45 The BufferedOutputStream Class (Page 3)	
aFormat:	
<u>BufferedOutputStream</u> <u>bufferedOutputStreamObject</u> = new <u>BufferedOutputStream(</u>	
fileOutputStreamObject);	
Example:	_
BufferedOutputStream output = new BufferedOutputStream(new FileOutputStream ("temp.dat"));	V
46 The BufferedOutputStream Class (Page 4)	
A DataOutputStream can be "wrapped" around the BufferedOutputStream to provide	

```
functionality to write primitives and strings:
        DataOutputStream dataOutputStreamObject = new DataOutputStream( new
          BufferedOutputStream( fileOutputStreamObject ) );

Example:
        DataOutputStream input = new DataOutputStream(new BufferedOutputStream( new
         FileOutputStream("temp.dat") ) );
53 The ObjectInputStream Class (Page 1)
      "Wraps" around a FileInputStream object to give it the ability to read "serializable"
       objects from the input stream
      Reads primitives and strings as well (contains all the methods of class)
        DataInputStream)
      Located in the java.io package
        import java.io.ObjectInputStream;
54 The ObjectInputStream Class (Page 2)
      Format:
        ObjectInputStream objectInputStreamObject = new ObjectInputStream(
          fileInputStreamObject );

Example:
        ObjectInputStream input = new ObjectInputStream( new FileInputStream(
         "temp.dat") );
55 The ObjectOutputStream Class (Page 1)
      n"Wraps" around a FileOutputStream object to give it the ability to write "serializable"
       objects to the output stream
      Writes primitives and strings as well (contains all the methods of class
       DataOutputStream)
      Located in the java.io package
        import java.io.ObjectOutputStream;
56 The ObjectOutputStream Class (Page 2)
      Format:
        ObjectOutputStream objectOutputStreamObject = new ObjectOutputStream(
          fileOutputStreamObject );

Example:
        ObjectOutputStream input = new ObjectOutputStream( new FileOutputStream(
         "temp.dat") );
57 The readObject() Method
      A method of class ObjectInputStream that reads a "serializable" object from the input
      May throw a ClassNotFoundException because when the JVM restores the object, it
       must first load the class into RAM
      ■Format:
        objectInputStreamObject.readObject();
      Example:
        SuffolkResident student = input.readObject();
58 The writeObject() Method
```

```
a method of class ObjectInputString that writes a "serializable" object into the output

        stream
      ■Format:
         objectOutputStreamObject.writeObject( object );
      Example:
         output.writeObject( new SuffolkResident() );
70 The Serializable Interface (Page 1)
      Objects that can be written to the output stream are said to serializable
         - Classes from which such objects are instantiated implement the Serializable
          interface
      To store an object, the JVM must store:

    The class name and its signature

         - All current property (data field) values of the class and its superclasses
      This process, called object serialization, is implemented in ObjectDataStream objects
72 The Serializable Interface
                                      (Page 2)
      Format:
         public class ClassName [extends ClassName] <u>implements Serializable</u>
         { ...
      Example:
         public class Student extends Object implements Serializable
         { ...
74 Serializing Arrays
                                      (Page 1)
      If all elements of an array are serializable, the array is serializable
      aAn array can be saved using writeObject() and restored using readObject()

    Recall that arrays are objects

75 Serializing Arrays
                                      (Page 2)
      Format for readObject():
        type[] arrayObject = ( castType[] ) objectInputStreamObject.readObject();
         - Method returns an object so it must be cast to the array object type
      Example:
         SuffolkResident[] student = (SuffolkResident[]) input.readObject();
76 Serializing Arrays
                                      (Page 3)
      Format for writeObject():
         objectInputStreamObject.readObject( arrayObject );
      Example:
         SuffolkResident[] student = new SuffolkResident[3];
         output.writeObject(student);
77 Random Access Files
                                      (Page 1)
      Refers to ability to access records from a data file at random
         - The opposite of random access is sequential access in which data must be accessed
          by passing through all intervening points
      Temperature Enables reading or writing information from or to any point in the file
```

- In a sequential-access file, must be accessed starting from the beginning of the file

79 Random Access Files (Page 2)

- Disks are random access media
 - Tapes are sequential access media
- **Sometimes** also called direct access

80 The RandomAccessFile Class (Page 1)

- **Objects** instantiated from this class have the ability to read and write *randomly*
- Similar to FileWriter and FileReader in that a file can be specified on the system to open when it is created it
 - Use either "path/filename" string or File object

81 The RandomAccessFile Class (Page 2)

- When opening a RandomAccessFile, indicate whether file just will be read from ("r") or also written to ("rw")
 - Must be able to read a file in order to write it
- Located in the java.io package import java.io.RandomAccessFile;

82 The RandomAccessFile Class (Page 3)

Format:

RandomAccessFile *randomAccessObject* = new RandomAccessFile("*path*| *filename*" | *fileObject*, "r"/"rw");

- The strings "r" or "rw" are the access methods
- Example:

RandomAccessFile studentFile = new RandomAccessFile("e:/studentFile.dat", "rw");

83 Random Access File Processing (Page 1)

- After a random file is open, common read and write methods are defined in the DataInput and DataOutput interfaces to perform I/O
 - Class RandomAccessFile implements both DataInput and DataOutput

84 Random Access File Processing (Page 2)

- For example:
 - The writeInt() writes four bytes of numeric integer data to the file
 - The readInt() reads four bytes of numeric integer data from the file
- There are write and read methods for every *primitive* data type

85 The File Pointer

- RandomAccessFile supports the notion of a file pointer:
 - Indicates the current location in the file
 - When the file is first created or opened, the file pointer is set to zero (0), indicating the beginning of the file
 - Calls to read and write methods advances the file pointer by the number of bytes read or written

88 The seek() Method (Page 1)

- **a** Method of the RandomAccessFile class that sets moves the file-pointer position
 - Measured in *bytes* from the beginning of file
 - The location at which the next read or write operation will begin

```
The offset position often is calculated based upon which record is to be accessed next
89 The seek() Method
                                (Page 2)
       Format:
         randomAccessObject.seek(long);
         - long is a long integer value or variable, or an arithmetic expression that evaluates to
           a long
       Examples:
         studentFile.seek(4);
         studentFile.seek(4 * (courseNumber - 1));
101 The length() Method for a RandomAccessFile Object
       For a RandomAccessFile object, returns the length of the file measured in bytes
       Format:
         randomAccessFileName.length();

Example:
         studentFile.seek( studentFile.length() );
108 The readChar() Method for a RandomAccessFile Object
       Reads a single character from an object instantiated from class RandomAccessFile
       Format:
         randomAccessFileObject.readChar();

Example:
         chars[ctr] = inFile.readChar();
109 The getChars() Method for a String Object
                                                               (Page 1)
       Copies characters from String into a destination char array
       Format:
         stringObject.getChars(srcBegin, srcEnd, dest, destBegin);
         - srcBegin—index of the first character to copy
         - srcEnd—index after the last character to copy
         - dest—the destination char array

    destBegin—start offset in destination char array

110 The getChars() Method for a String Object
                                                               (Page 2)
       Format:
         stringObject.getChars(srcBegin, srcEnd, dest, destBegin);

Example:
         outString.getChars(0, length, chars, 0);
         - The variable chars is a char array
111 | The writeChars() Method for a RandomAccessFile Object (Page 1)
       Writes a String to as RandomAccessFile as a sequence of characters ...
         - Each character is written separately to the data output stream as though were being
           written by the writeChar() method contained within a loop
       The write operation starts at the current position of the file pointer
112 The writeChars() Method for a RandomAccessFile Object (Page 2)
       ■Format:
         randomAccessFileName.writeChars(String);
```

Example:

outFile.writeChars(new String(chars));

 The variable chars is a char array and String is a call to String constructor that takes a char array

113 The DataInput and DataOutput Interfaces (Page 1)

- DataInput—an interface that implements the reading bytes from a binary stream
 - The bytes may be reconstructed into any of the Java primitive types
- DataOutput—an interface that implements the writing bytes to a binary stream
 - Data converted from any of the Java primitive types back to a series of bytes

114 The DataInput and DataOutput Interfaces (Page 2)

- The RandomAccessFile class implements both the DataInput and DataOutput interfaces
 - Therefore a RandomAccessFile object "is a" DataInput object and "is a" DataOutput object
- Subtyping allows RandomAccessFile objects to be "assigned" to a DataInput or DataOutput object
- Found in the java.io package import java.io.DataInput; import java.io.DataOutput;

115 The DataInput and DataOutput Interfaces (Page 3)

Examples:

<u>DataOutput</u> courseFile = new RandomAccessFile("courseFile.dat", "rw")

- Subtyping when instantiating a RandomAccessFile object variable public static String readFixedLengthString(int size, <u>DataInput</u> inFile)
- Subtyping when passing a RandomAccessFile object to a parameter variable