More About Object-Oriented Design

Comma Separated Values

Objectives

You will be able to

- Use Comma Separated Values files in your Java programs.
- Understand the concepts of serialization and deserialization.

Comma Separated Values

- A widely used method of representing structured data in a text file.
  - Lines of plain text with values separated by commas.
- Can be opened in Excel
  - Each line of the file becomes a row.
  - Each value in the line goes into a column of the table.
- Can be edited with any plaintext editor.
- Easily read and understood in a Java program.
  - Can be used to initialize an object.

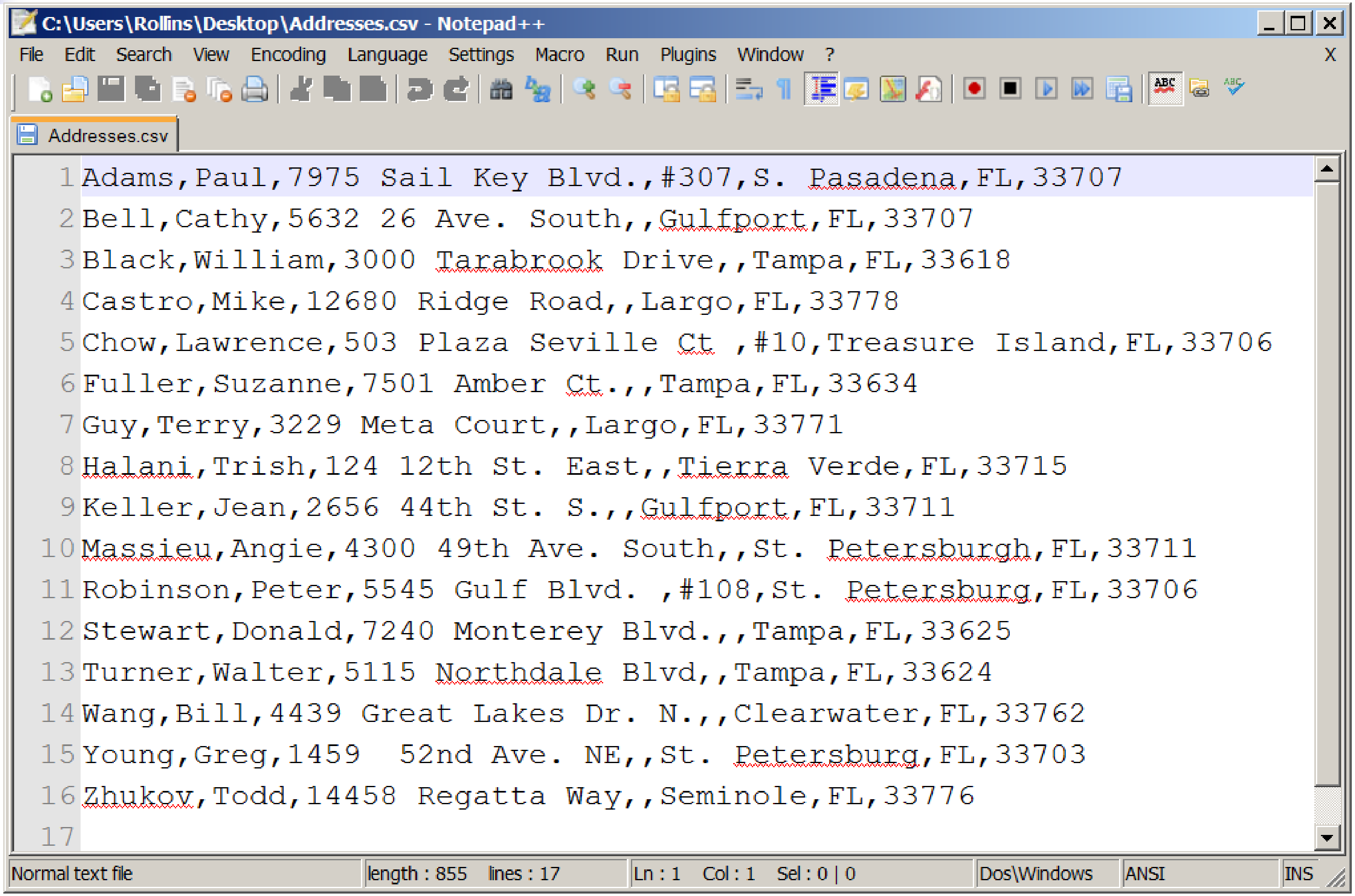
Example of Comma Separated Values

- Addresses
  - Each of the standard parts of a maining address is included on one line.
  - Parts separated by commans

- Download from the class web site:
- http://www.csee.usf.edu/~turnerr/Programming_Concepts/Downloads/
- File Addresses.csv

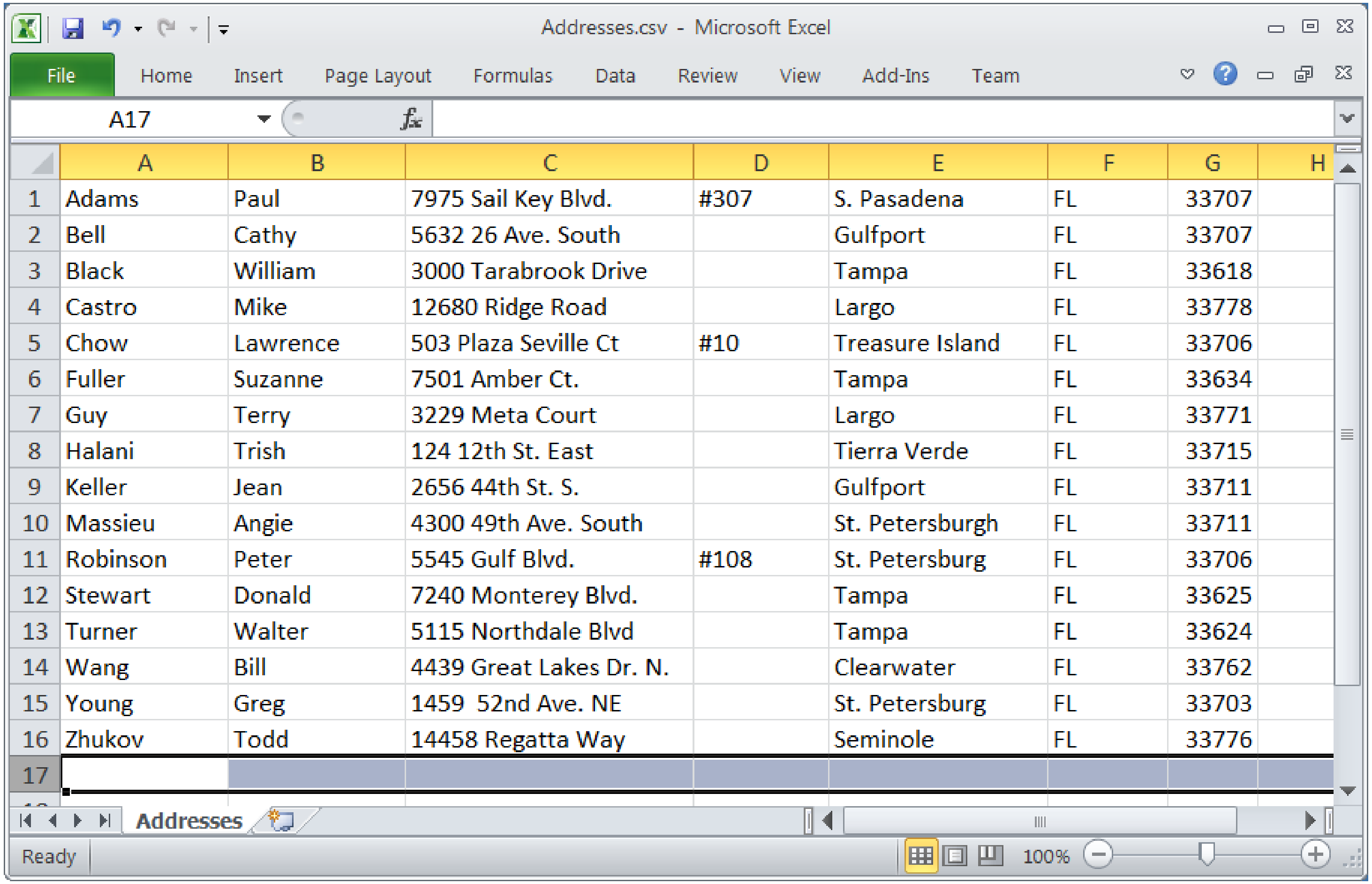
- Look at file in a text editor.
- Open it in Excel.

Addresses.csv in Notepad++



```
Adams,Paul,7975 Sail Key Blvd.,#307,S. Pasadena,FL,33707
Bell,Cathy,5632 26 Ave. South,,Gulfport,FL,33707
Black,William,3000 Tarabrook Drive,,Tampa,FL,33618
Castro,Mike,12680 Ridge Road,,Largo,FL,33778
Chow,Lawrence,503 Plaza Seville Ct ,#10,Treasure Island,FL,33706
Fuller,Suzanne,7501 Amber Ct.,,Tampa,FL,33634
Guy,Terry,3229 Meta Court,,Largo,FL,33771
Halani,Trish,124 12th St. East,,Tierra Verde,FL,33715
Keller,Jean,2656 44th St. S.,,Gulfport,FL,33711
Massieu,Angie,4300 49th Ave. South,,St. Petersburgh,FL,33711
Robinson,Peter,5545 Gulf Blvd. ,#108,St. Petersburg,FL,33706
Stewart,Donald,7240 Monterey Blvd.,,Tampa,FL,33625
Turner,Walter,5115 Northdale Blvd,,Tampa,FL,33624
Wang,Bill,4439 Great Lakes Dr. N.,,Clearwater,FL,33762
Young,Greg,1459  52nd Ave. NE,,St. Petersburg,FL,33703
Zhukov,Todd,14458 Regatta Way,,Seminole,FL,33776

```

Addresses.csv in Excel



	A	B	C	D	E	F	G	H
1	Adams	Paul	7975 Sail Key Blvd.	#307	S. Pasadena	FL	33707	
2	Bell	Cathy	5632 26 Ave. South		Gulfport	FL	33707	
3	Black	William	3000 Tarabrook Drive		Tampa	FL	33618	
4	Castro	Mike	12680 Ridge Road		Largo	FL	33778	
5	Chow	Lawrence	503 Plaza Seville Ct	#10	Treasure Island	FL	33706	
6	Fuller	Suzanne	7501 Amber Ct.		Tampa	FL	33634	
7	Guy	Terry	3229 Meta Court		Largo	FL	33771	
8	Halani	Trish	124 12th St. East		Tierra Verde	FL	33715	
9	Keller	Jean	2656 44th St. S.		Gulfport	FL	33711	
10	Massieu	Angie	4300 49th Ave. South		St. Petersburgh	FL	33711	
11	Robinson	Peter	5545 Gulf Blvd.	#108	St. Petersburg	FL	33706	
12	Stewart	Donald	7240 Monterey Blvd.		Tampa	FL	33625	
13	Turner	Walter	5115 Northdale Blvd		Tampa	FL	33624	
14	Wang	Bill	4439 Great Lakes Dr. N.		Clearwater	FL	33762	
15	Young	Greg	1459 52nd Ave. NE		St. Petersburg	FL	33703	
16	Zhukov	Todd	14458 Regatta Way		Seminole	FL	33776	
17								

Processing a CSV File in Java

- The Java String class has a method that allows us to easily divide a CSV line into separate strings.
  - **`String[] split(` *`separator`* `)`**

- The separator can be a String consisting of a single character.
  - In our case a comma (",")

- Example:
  - **`String[] parts = line.split(",");`**

Class Address

- Let's create a class to hold addresses from a CSV file.

- Constructor takes a single line from the file as its only parameter.
  - Splits the line into its parts.
  - Initializes member variables from the resulting parts.

```
//************************************************
//  Address.java
//
//  Represents an address.
//
//************************************************
public class Address
{
    // Instance variables
    private String last_name;
    private String first_name;
    private String street_address_1;
    private String street_address_2;
    private String city;
    private String state;
    private String zip_code;
```

Address.java (continued)

```
//-----------------------------------------------------
// Constructor - sets up an Address object from one line
// of a Comma Separated Values file.
//-----------------------------------------------------
public Address(String csv_str)
{
    String[] address_info = csv_str.split(",");

    this.last_name = address_info[0];
    this.first_name = address_info[1];
    this.street_address_1 = address_info[2];
    this.street_address_2 = address_info[3];
    this.city = address_info[4];
    this.state = address_info[5];
    this.zip_code = address_info[6];
}
```

Address.java (continued)

```
//------------------------------------------------------
// Returns a single string with all address components
// in a printable format.
//------------------------------------------------------
public String toString()
{
    return
        this.first_name + " " +
        this.last_name + "\n" +
        this.street_address_1 + " " +
        this.street_address_2  + "\n" +
        this.city + ", " +
        this.state + " " +
        this.zip_code + "\n";
}
```

Compile it.

Test_Address.java

```java
//**************************************************
//  Test_Address.java
//     A test driver for class Address.
//**************************************************
import java.util.Scanner;
import java.io.File;
import java.io.FileNotFoundException;

public class Test_Address
{
    //-----------------------------------------------
    // Reads a CSV file of addresses and creates
    // Address objects from the contents.
    //-----------------------------------------------
    public static void main (String[] args)
    {
        Address address;
```

Test_Address.java (continued)

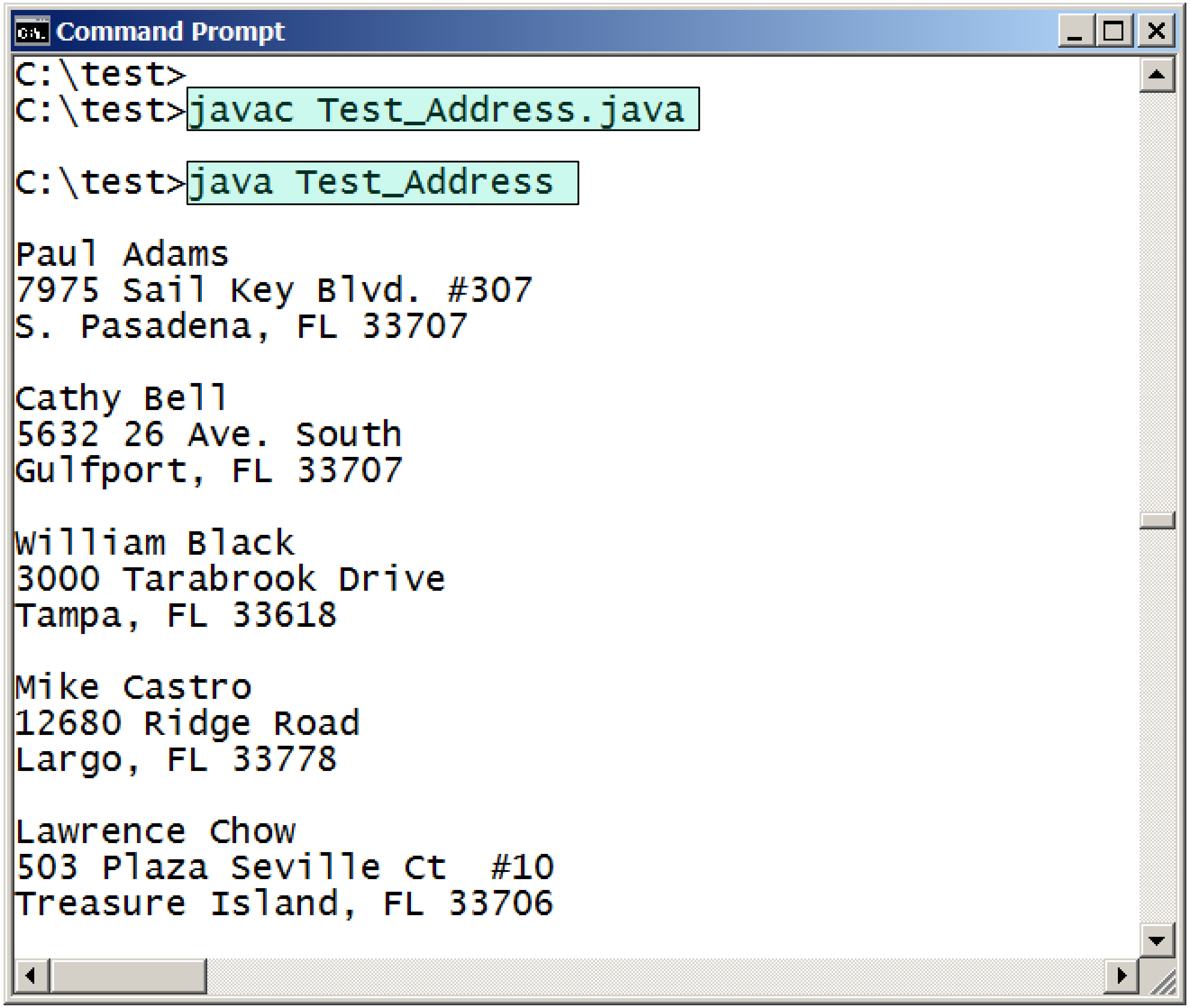
```
try
{
    // Create a scanner for the addresses file.
    File file = new File("Addresses.csv");
    Scanner fileScanner = new Scanner(file);
    System.out.println();

    // Read each line of the file and create an
    // Address object from the contents.
    while (fileScanner.hasNext())  // Test for end of file
    {
        // Read next line from file
        String str = fileScanner.nextLine();
        Address adr = new Address(str);
        System.out.println(adr);
    }
}
```

Test_Address.java (continued)

```
        catch (FileNotFoundException ex)
        {
            System.out.println("File Addresses.csv was not found.");
        }

        System.out.println();
        System.out.println("Program complete.");
    }
}
```

Compile and test.

Test_Address.java in Action

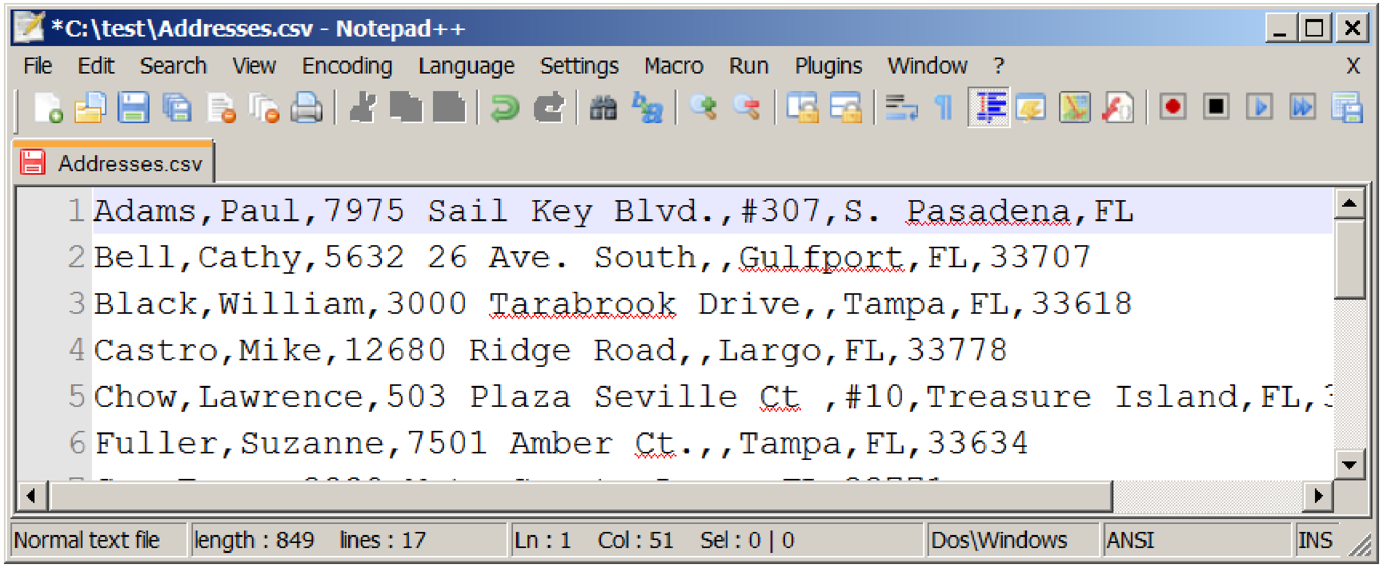


```
C:\test>
C:\test>javac Test_Address.java

C:\test>java Test_Address

Paul Adams
7975 Sail Key Blvd. #307
S. Pasadena, FL 33707

Cathy Bell
5632 26 Ave. South
Gulfport, FL 33707

William Black
3000 Tarabrook Drive
Tampa, FL 33618

Mike Castro
12680 Ridge Road
Largo, FL 33778

Lawrence Chow
503 Plaza Seville Ct  #10
Treasure Island, FL 33706
```

What if parts are missing?

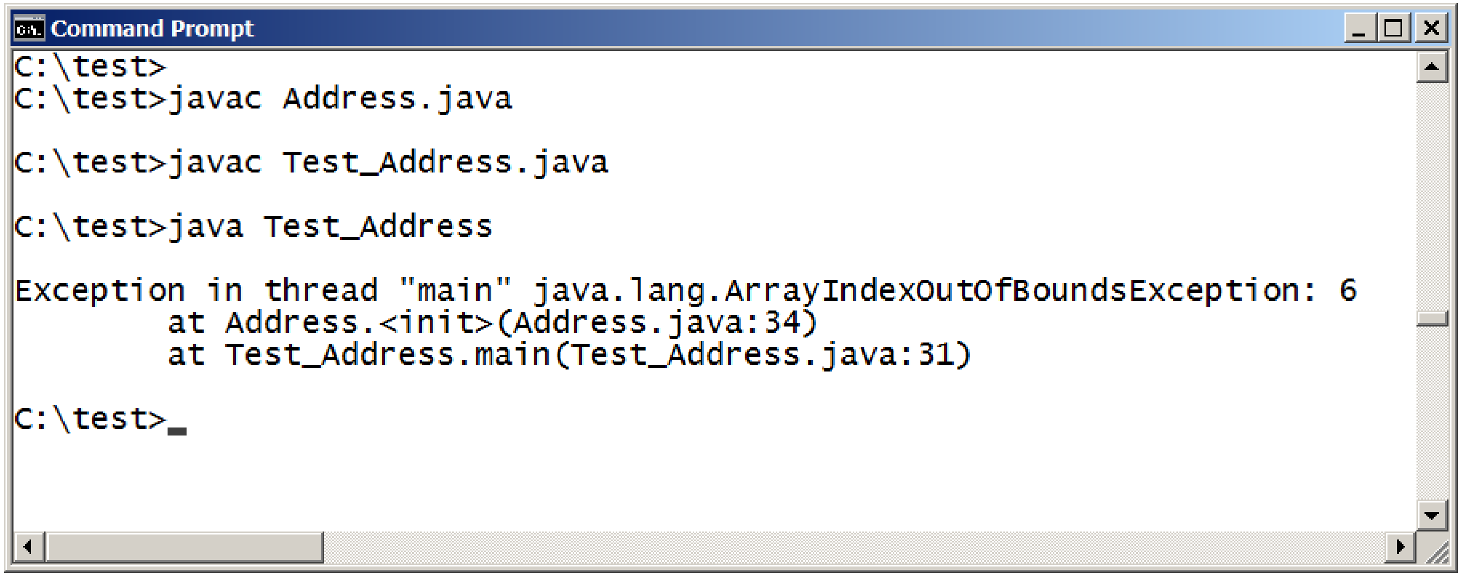
- What happens if the file has a line without the expected number of parts?
  - Let's try it and see.

- Delete the zip code from the first line of Addresses.csv



```
Adams,Paul,7975 Sail Key Blvd.,#307,S. Pasadena,FL
Bell,Cathy,5632 26 Ave. South,,Gulfport,FL,33707
Black,William,3000 Tarabrook Drive,,Tampa,FL,33618
Castro,Mike,12680 Ridge Road,,Largo,FL,33778
Chow,Lawrence,503 Plaza Seville Ct ,#10,Treasure Island,FL,3
Fuller,Suzanne,7501 Amber Ct.,,Tampa,FL,33634
```

Bad Line in Input File



```
C:\test>
C:\test>javac Address.java

C:\test>javac Test_Address.java

C:\test>java Test_Address

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 6
        at Address.<init>(Address.java:34)
        at Test_Address.main(Test_Address.java:31)

C:\test>_
```

Throws an exception.
Bad!
We need to avoid this.

Validation Check

- Add code to verify that the line has the expected number of parts.

- Constructor has to return an object.
  - Add "isValid" member to tell caller that the returned Address object is not valid.

- Check isValid in the toString method.

Address.java

- Add member variable

```
private boolean isValid;
```

- In constructor, check number of parts found in the line and set isValid accordingly.

Constructor

```
public Address(String csv_str)
{
    String[] address_info = csv_str.split(",");
    if (address_info.length == 7)
    {
        this.last_name = address_info[0];
        this.first_name = address_info[1];
        this.street_address_1 = address_info[2];
        this.street_address_2 = address_info[3];
        this.city = address_info[4];
        this.state = address_info[5];
        this.zip_code = address_info[6];
        isValid = true;
    }
    else
    {
        isValid = false;
    }
}
```

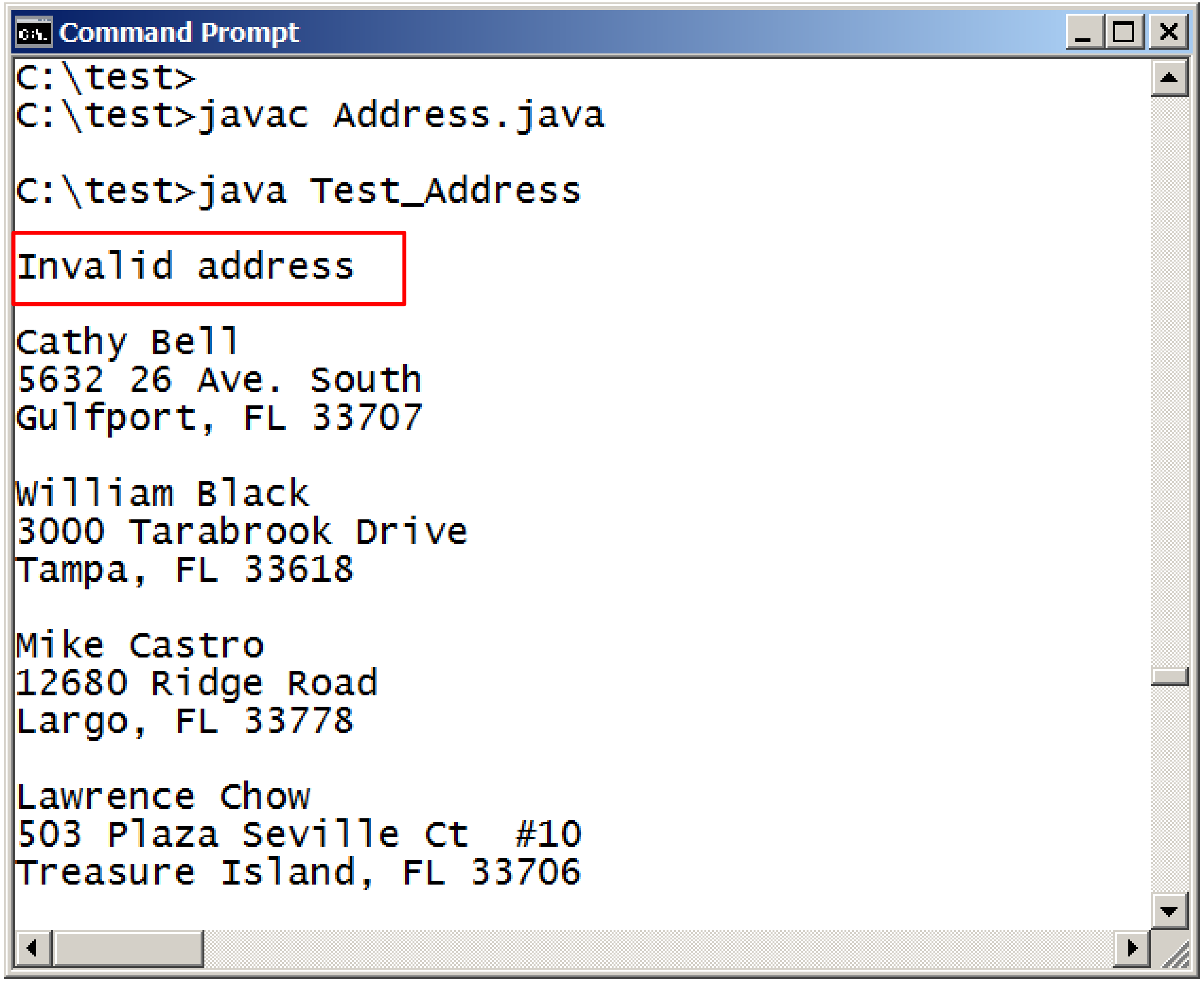
Add Accessor Method

```
//------------------------------------------------------
// Returns a boolean says whether or not the address
// is valid.
//------------------------------------------------------
public boolean isValid()
{
   return this.isValid;
}
```

Update toString Method

```
//------------------------------------------------------
// Returns a single string with all address components
//------------------------------------------------------
public String toString()
{
    if (this.isValid)
    {
        return
            this.first_name + " " +
            this.last_name + "\n" +
            this.street_address_1 + " " +
            this.street_address_2  + "\n" +
            this.city + ", " +
            this.state + " " +
            this.zip_code + "\n";
    }
    else
    {
        return "Invalid address\n";
    }
}
```

Try Again

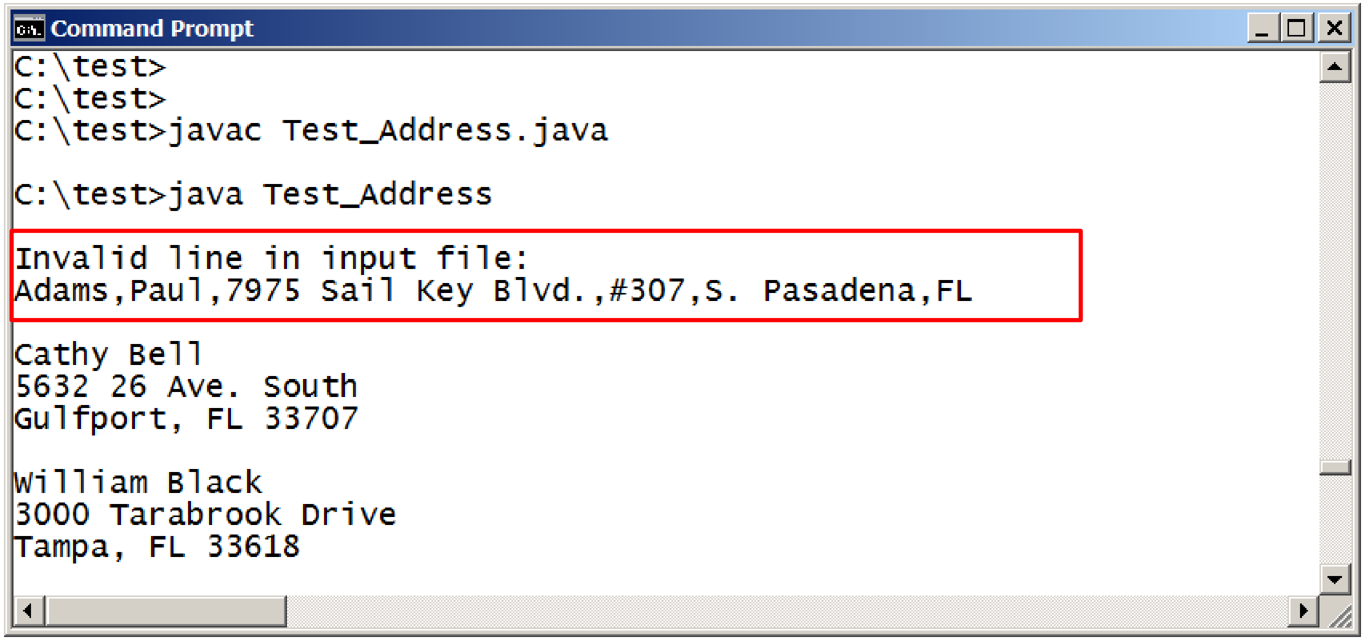


```
C:\test>
C:\test>javac Address.java

C:\test>java Test_Address

Invalid address

Cathy Bell
5632 26 Ave. South
Gulfport, FL 33707

William Black
3000 Tarabrook Drive
Tampa, FL 33618

Mike Castro
12680 Ridge Road
Largo, FL 33778

Lawrence Chow
503 Plaza Seville Ct  #10
Treasure Island, FL 33706
```

Check Validity in Test Driver

- We can provide more information to the user about what was wrong if we check isValid in the test driver.

Outputting an Error Message



```
C:\test>
C:\test>
C:\test>javac Test_Address.java

C:\test>java Test_Address

Invalid line in input file:
Adams,Paul,7975 Sail Key Blvd.,#307,S. Pasadena,FL

Cathy Bell
5632 26 Ave. South
Gulfport, FL 33707

William Black
3000 Tarabrook Drive
Tampa, FL 33618
```

Serialization

- A class that can get its values from a file should also be able to save its values in a file.
  - Serialization / Deserialization

Key Concept

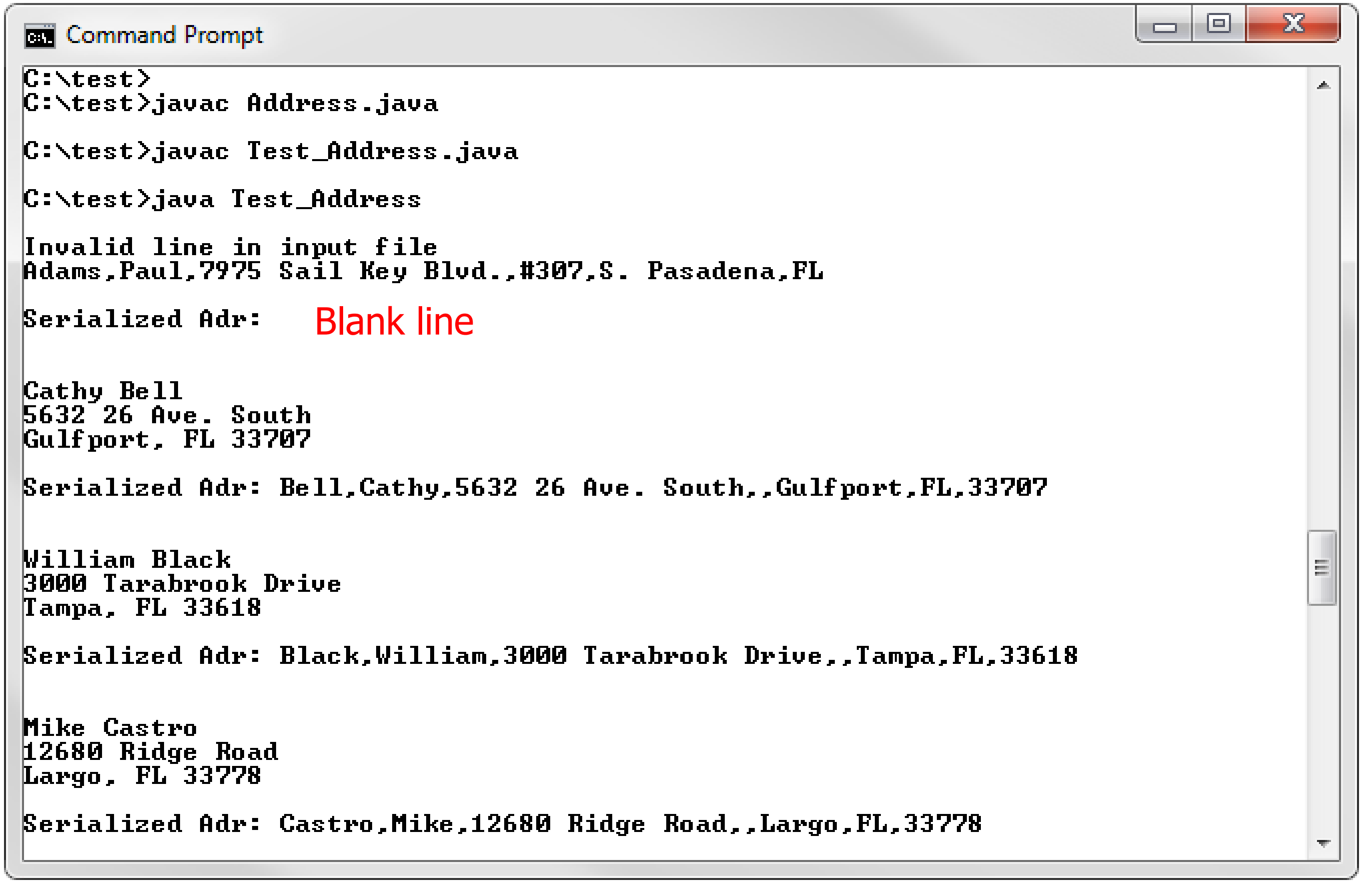
Serialize

```
//------------------------------------------------------
// Returns a single string with all address components
// suitable for input by a Java program or by Excel.
//------------------------------------------------------
public String Serialize()
{
    if (this.isValid)
    {
        return
            this.last_name + "," +
            this.first_name + "," +
            this.street_address_1 + "," +
            this.street_address_2  + "," +
            this.city + "," +
            this.state + "," +
            this.zip_code + "\n";
    }
    else
    {
        return "\n";
    }
}
```

Add to Test Driver

```
while (fileScanner.hasNext())  // Test for end of file
{
    // Read next line from file
    String str = fileScanner.nextLine();
    Address adr = new Address(str);

    if (adr.isValid())
    {
        System.out.println(adr);
    }
    else
    {
        System.out.println("Invalid line in input file");
        System.out.println(str + "\n");
    }
    // Output serialized Address object
    String serialstr = adr.Serialize();
    System.out.println("Serialized Adr: " + serialstr);
    System.out.println();
}
```

Test Run



```
C:\test>
C:\test>javac Address.java

C:\test>javac Test_Address.java

C:\test>java Test_Address

Invalid line in input file
Adams,Paul,7975 Sail Key Blvd.,#307,S. Pasadena,FL

Serialized Adr:


Cathy Bell
5632 26 Ave. South
Gulfport, FL 33707

Serialized Adr: Bell,Cathy,5632 26 Ave. South,,Gulfport,FL,33707


William Black
3000 Tarabrook Drive
Tampa, FL 33618

Serialized Adr: Black,William,3000 Tarabrook Drive,,Tampa,FL,33618


Mike Castro
12680 Ridge Road
Largo, FL 33778

Serialized Adr: Castro,Mike,12680 Ridge Road,,Largo,FL,33778
```

Blank line

Assignment

- Assignment to be submitted and graded:
  - Project 14: Comma Separated Dogs
  - An exercise in serialization and deserialization.