



Project 13: Going to the Dogs



Project 13: Going to the Dogs

- This is an extension to the class Dog example on the class web site.
- http://www.csee.usf.edu/~turnerr/Programming_Concepts/Downloads/2016_03_21_Writing_Classes/
 - You may use the example as a starting point for this project.
 - It is OK to copy code from any file posted on the class web site.



Assignment

- Add a constructor to the Dog class that takes a file Scanner object as its only parameter.
 - The file will consist of attributes of dogs.
 - As read from the keyboard in the Dogs example
- Let the new constructor read three lines from the file, using the scanner object passed by the caller, to get the attributes of a single dog, and use those values to initialize a Dog object:
 - Name
 - Breed
 - Age
- Remember that you will need to clear a blank line from the Scanner after reading the dog's age as an integer.



Assignment

- Let the new constructor have the signature
 - `Dog(Scanner dogScanner)`
- Modify the Dog class to keep track of how many Dog objects have been created since the program started.
- Add a static member function to return the number of Dogs in the system:
 - `public static int Get_Dog_Count()`



Assignment

Create a test driver called DogsFromFile.

- Accepts input from the keyboard for a file name.
- Creates a Scanner object for the file.
- Repeatedly invokes the new Dog constructor, passing the Scanner object as the only parameter value.
- Adds the resulting Dog object to an array.



DogsFromFile.java (continued)

- Upon reaching end of file, steps through the array of Dogs and outputs the attributes of each dog to the screen.
- Finally, invokes the new static member function of class Dog, `Get_Dog_Count`, to get the number of Dog objects that have been created and outputs the value to the screen.



DogsFromFile.java

- You may assume that there will be no more than 100 dogs in the file.
- If the file specified by the user does not exist, output an error message and let the user try again.



Implementation Tips

- As always, work in tiny steps.
- Start with the Dog class in the posted example.
- Add a static member variable to keep track of the number of Dogs that have been created.
- Add a constructor initialize a Dog object from a file.
 - The new constructor should take a Scanner object as its only parameter.
 - Use the Scanner object to read three lines from the file and initialize the Dog member variables with the values from the file.
 - Modify each constructor in class Dog to increment the count of dogs that have been created.
- Get your modified Dog.java to compile.



Step 2

- Write a test driver to test the new constructor with just one dog.
- Create a test input file with information for one dog.
 - Be sure you have a newline character after the last dog's age.



Step 2 (continued)

- Let the test driver
 - Create a File object for your test input file, using a fixed file name.
 - Create a Scanner object using the File object.
 - Create a Dog object using the new constructor that takes a Scanner as its parameter.
 - Display the dog using println.
 - Display the number of dogs created.
 - Should be 1.



Implementation Tips

Step 3

- Extend your test driver to work with an input file having an arbitrary number of dogs.
 - (No more than 100.)
- Still use a fixed (hard coded) file name.
- Add more dogs to your test input file.
- Build an array of Dog objects containing the dogs from the file.
- Upon reaching end of file, step through the array and output the attributes of each Dog, using `println()`.



Implementation Tips

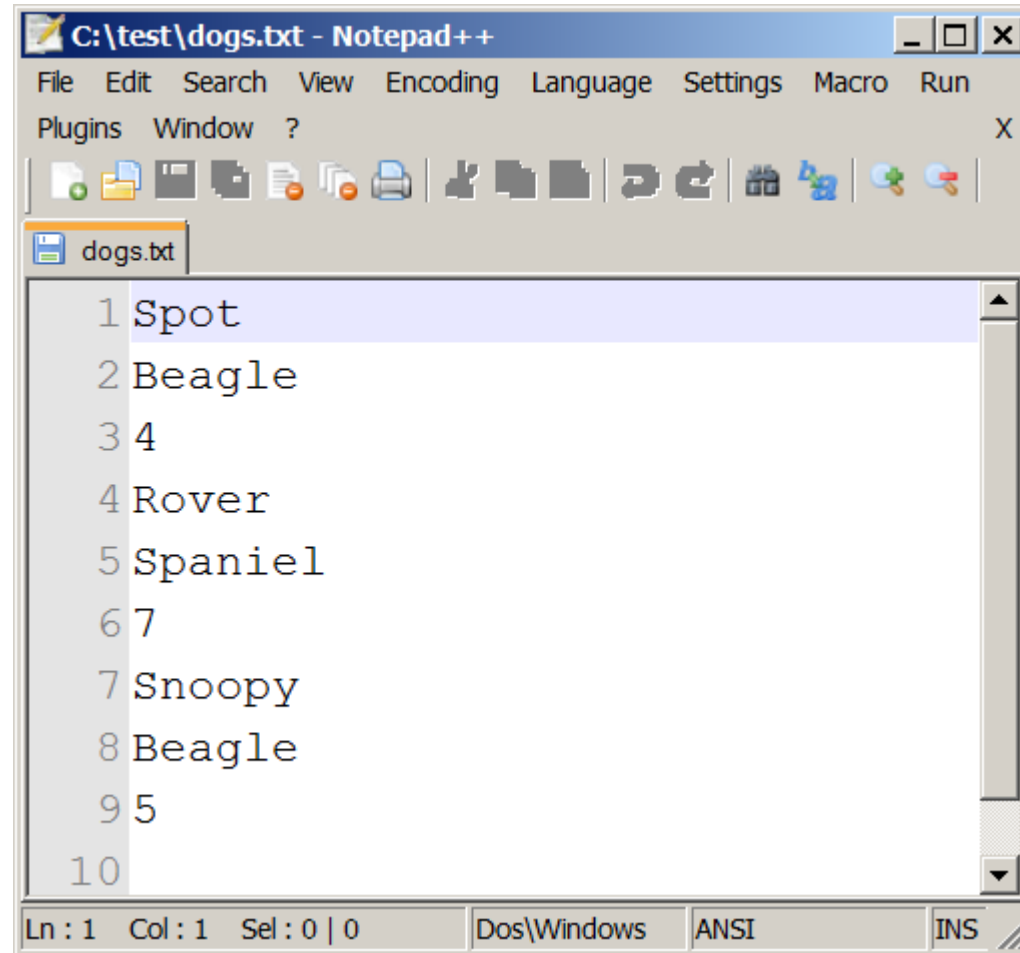
Step 4

- Add code to your test driver to get the file name from the user.
 - Initially let the program crash if the file does not exist.

Step 5

- Add code to output an error message if the file specified by the user does not exist and let the user try again.

A Test File

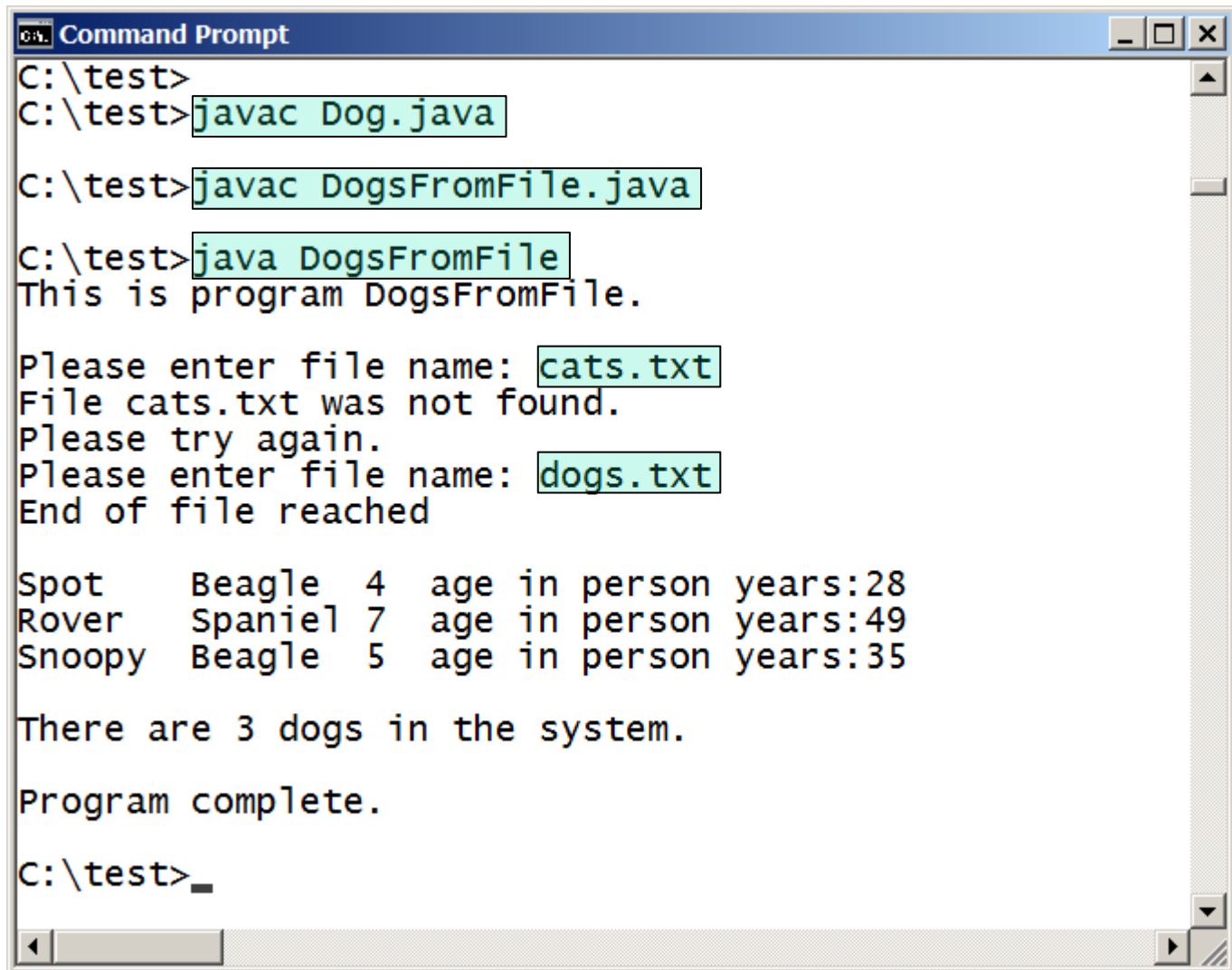


The image shows a Notepad++ window titled "C:\test\dogs.txt - Notepad++". The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Macro, Run, Plugins, Window, and ?. The toolbar contains icons for opening, saving, printing, and other file operations. The text area displays the following content:

```
1 Spot
2 Beagle
3 4
4 Rover
5 Spaniel
6 7
7 Snoopy
8 Beagle
9 5
10
```

The status bar at the bottom shows "Ln : 1 Col : 1 Sel : 0 | 0", "Dos\Windows", "ANSI", and "INS".

Sample Run



```
Command Prompt
C:\test>
C:\test>javac Dog.java
C:\test>javac DogsFromFile.java
C:\test>java DogsFromFile
This is program DogsFromFile.

Please enter file name: cats.txt
File cats.txt was not found.
Please try again.
Please enter file name: dogs.txt
End of file reached

Spot      Beagle  4   age in person years:28
Rover     Spaniel 7   age in person years:49
Snoopy    Beagle  5   age in person years:35

There are 3 dogs in the system.

Program complete.

C:\test>_
```



Submission

- Put your Java source files into a folder and zip it.
- Submit your zipped folder via Canvas Assignments.
- Project is due by 11:59 PM
 - Sunday, April 10 **All Sections**
- Recommendation:

Do this project in your lab session or help session.



Ground Rules

- It is OK to *discuss* the project with other students BUT
 - Do not share your code with other students.
 - Before or after submitting the project.
- Do not copy any other student's code.
 - Or even look at it.
- Do not let anyone copy or examine your code.



Ground Rules

Except for code posted on the class web site

- Do not copy code from the Internet
 - or any other source (other than the textbook.)
- Do not ask for help on an Internet forum.
 - If you need help, ask your instructor or a TA.
 - Come to lab and help sessions.
- Write your own code.