

Reading Data from a File

User input is just one way to get data into a program.

Programs also read data from input files. For instance, in the programming contest program “Parking Lot” data must be read from a file. So when you run your program, it will just print out the results based on what is in the input file.

Here’s a sample class and main method that reads in all the lines of a file, as strings, and prints them out:

```
class TestFile
{
    public static void main(String args[])
    {
        String s="";
        try {

            Scanner scanner = new Scanner(new File("test.dat"));
            while (scanner.hasNext())
            {
                s+=scanner.nextLine()+" ";
            }
            System.out.println(s);
        }
        catch (Exception e)
        {
            System.out.println(e);
        }
    }
}
```

You should use similar code in a “readCars” function of your ParkingLot class. Only you’ll be reading integers, not strings, and you’ll be building a list of cars from the input.

The Scanner class should be familiar to you. But instead of reading from “System.in” which is a ‘file’ representing what the user inputs, this program reads from the file “test.dat”.

The ‘try’ and ‘catch’ statements are code which you probably haven’t seen. The fancy word for such code is *Exception Handling*. Its an advanced topic, but you have to know a bit about it because to read from a file you must use ‘try’-‘catch’

Exceptions – when the program behaves in an unexpected way.

Example: Your program tries to open a file but the file doesn't exist.

In Java, when you write a class, you can specify that some exception might occur and that the client of the class must deal with it. The Java library class `File` is this way—if you use class `File`, you must put the code within try-catch.

```
try {  
    // some code dealing with a file  
  
    // this is the code that will run in most cases  
  
}  
catch (Exception e)  
{  
    // what to do if exception occurs  
    // this code will only run if there is a problem  
}
```

For now, you only need a rudimentary understanding of exception handling. The important thing is to put a try-catch clause around your `File` processing code (as in the sample).