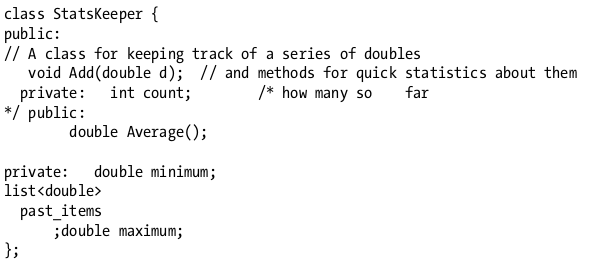
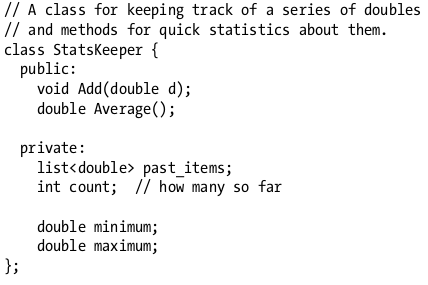
**Chapter 4 Aesthetics**

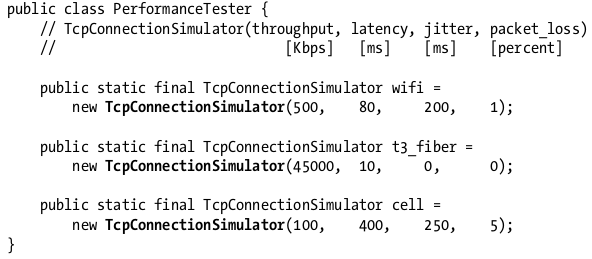
* good source code should be just as easy on the eyes
* use consistent layout
* make similar code similar
* group related lines of code into blocks



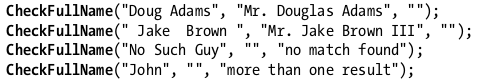
Better:



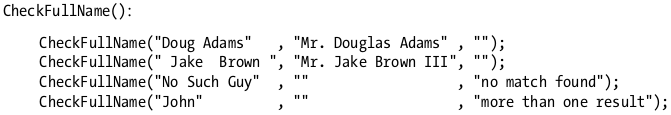
* it’s easier to work with code that’s aesthetically pleasing.
* rearrange line breaks to be consistent and compact.



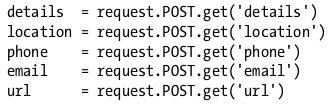
* it is recommended to move the comments up to the top and then put all the parameters on one line. so even though the comment isn’t right to each number, the “data” is lined up in a more compact table.
* use methods to clean up irregularity



* it eliminates a lot of duplicated code from before, making the code more compact.
* important parts of each test case are now by themselves, in plain sight.
* use column alignment when helpful
  + straight edges and columns make it easier for readers to scan through text.



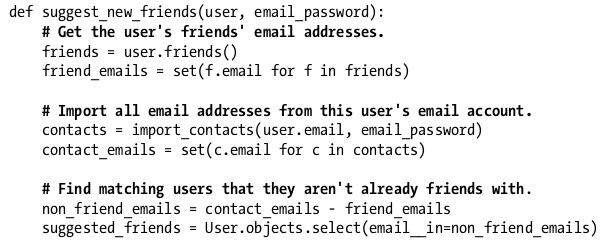
* some programmers don’t like column alignment, one reason is that it takes more work to set up and maintain the alignment. another reason is it creates a larger diff when making changes- a one line change might cause five other lines to change.
* pick a meaningful order and use it consistently.



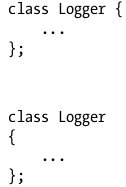
* organize declaration into blocks



* break code into “Paragraphs”
  + it’s a way to group similar ideas together and set them apart from other ideas.
  + provides a visual “stepping stone” - without it, it’s easy to lose your place on the page.
  + facilitates navigation from one paragraph to another.



* there are certain aesthetic choices that just boil down to personal style. like where the open brace for a class definition should go.



* consistent style is more important than the “right” style.