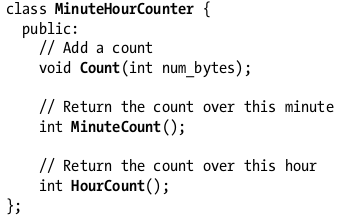
**Chapter 15 Designing and Implementing a “Minute/Hour Counter”**

**Problem**

* Keep track of how many bytes a web server has transferred over the past minute and hour.



**To be improved:**

* names should be specific, concrete, and easy to say.
* method named with Get as prefix should only do lightweight jobs.
* method name should not be confusing.
* The method name of Count() is confusing. Should be renamed with Add(int num\_bytes)
* num\_bytes is too specific so should be replaced by count. so the class would be usable to other users of different purpose.



* but now the comment is redundant so it should be removed or improved.

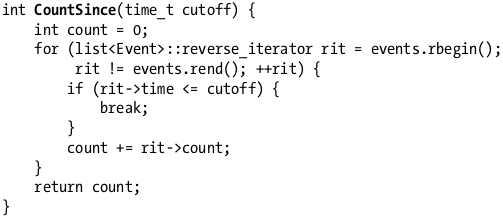


* comments should not be misleading.
* now the MinuteCount() the comment is misleading and should be changed to:



**Attempt 1:A Naive Solution**

* Avoid duplicating codes.
* MinuteCount() and HourCount() are identical. They would make the code smaller if they’ll share duplicated code.
* rit is used for reverse iterator.
* traditional for loop format are easier to read.



**Performance Problem**

* Data that are not needed anymore should be removed already because it will cause memory problem as the program continues to run over time.
* The MinuteCounter should automatically delete events that are older than an hour because they are needed anymore.
* MinuteHourCounter should keep separate minute\_count and hour\_count variables that are kept up date with each call to Add().

**Attempt 2 Conveyor Belt Design**

* Delete data that are not necessary.
* Keep minute and hour count up to date.
* one copy is better than having two copies of data.

**Attempt 3: A Time-Bucketed Design**

* bucket all events within a small window together, and have a summary of events with a single total, this way the memory consumption is fixed over time.
* To get control of the data you can bucket the events that way you can remove files out of the time frame and get more precise result.

**Trailing Bucket Counter**

* create helper class to wrap your codes and not be confused with lot of codes in your codebase.
* do one task at a time.

**ConveyorQueue**

* job is to deal with underlying counts and totals.