Cumulative Algorithms

These are the algorithms that accumulate or compute a running sum. These algorithms include averaging and more complex algorithms like standard deviation and variance. Here is the pseudocode for computing an average:

```
counter <- 0
accumulator <- 0
examine each item in the collection
  if the item meets the condition then
    count the item
    add the item to the accumulator
if the counter is > 0 then
  average <- accumulator / counter
```

Here's a loop that calculates an **average daily temperature** from a list of readings.

```cpp
double sum{0.0};
for (auto t : temperatures)
{
    sum += t;
}
double avg = sum / temp.size(); // nan if no elements
```

Because `sum` is type `double`, this loop sets `avg` to `nan` if there are no elements in the `vector`, using that as an **error code**. If both were `int`, then the program would crash from the division by zero. In addition, since this loop counts **all** of the readings, you don't need a counter, but can use the `vector` size instead.

Here's another example, which computes the **average word size** in a `vector<string>`. Because you don't want to make unnecessary copies of each element, nor to inadvertently modify an element, the loop variable is `const string&`.

```cpp
double sum{0};
for (const string& word : words)
{
  sum += word.size();
}
double avg_word_size = sum / words.size();
```