Often we start out analyzing data by writing straight R code at the console. This code is designed to accomplish a single task---whatever it is that we are trying to do *right now*. For example, consider the following code that operates on download logs published by RStudio from their mirror of the Comprehensive R Archive Network (CRAN). This code counts the number of times the filehash package was downloaded on July 20, 2016.

```
library(readr)
library(dplyr)

## Download data from RStudio (if we haven't already)
if(!file.exists("data/2016-07-20.csv.gz")) {
        download.file("http://cran-logs.rstudio.com/2016/2016-07-20.csv.gz",
                      "data/2016-07-20.csv.gz")
}
cran <- read_csv("data/2016-07-20.csv.gz", col_types = "ccicccccci")
cran %>% filter(package == "filehash") %>% nrow
[1] 179
```

This computation is fairly straightforward and if one were only interested in knowing the number of downloads for this package on this day, there would be little more to say about the code. However, there are a few aspects of this code that one might want to modify or expand on:

- the **date**: this code only reads data for July 20, 2016. But what about data from other days? Note that we would first need to obtain that data if we were interested in knowing download statistics from other days.
- the **package**: this code only returns the number of downloads for the filehash package. However, there are many other packages on CRAN and we may want to know how many times these other packages were downloaded.

Once we've identified which aspects of a block of code we might want to modify or vary, we can take those things and abstract them to be arguments of a function.

Mark as completed