1.
2.
3.

4. Figure 1 shows the content of a JSON file, "data.json" and the following code reads this file into R.

```
> library(jsonlite)
> crimes <- fromJSON("data.json")
```

Write down what the result of the following R code would be.

```
> crimes
```

```
                 category location_type location.latitude location.street.id
1 anti-social-behaviour         Force         51.497899             953525
2 anti-social-behaviour         Force         51.507309             956645
  location.longitude       id   month
1          -0.119685 34274772 2014-07
2          -0.128348 34290854 2014-07
```

Write down what the result of the following R code would be.

```
> dim(crimes)
```

```
[1] 2 5
```

The following code creates a mongoDB collection from the JSON file.

```
> library(mongolite)
> m <- mongo(collection="testcrimes")
> m$drop()
> m$insert(crimes)
```

Write down what the result of the following R code would be.

```
> m$find(query='{ "id": 34274772 }',
+        fields='{ "_id": 0, "category": 1, "location_type": 1, "month": 1 }')

 Found 1 records...
 Imported 1 records. Simplifying into dataframe...
                category location_type   month
1 anti-social-behaviour         Force 2014-07
```

```
[
    {
        "category": "anti-social-behaviour",
        "location_type": "Force",
        "location": {
            "latitude": "51.497899",
            "street.id": 953525,
            "longitude": "-0.119685"
        },
        "id": 34274772,
        "month": "2014-07"
    },
    {
        "category": "anti-social-behaviour",
        "location_type": "Force",
        "location": {
            "latitude": "51.507309",
            "street.id": 956645,
            "longitude": "-0.128348"
        },
        "id": 34290854,
        "month": "2014-07"
    }
]
```

Figure 1: The JSON file "`data.json`"

5. Figure 2 shows the content of an XML file, `"data.xml"`.

   Write R code to read that file into R and extract all donation elements where the donation amount is larger than 2000.

   The output that your code should produce is shown below:

```
> library(rvest)
> data <- xml("data.xml")
> xml_nodes(data, xpath="//donation[@amount > 2000]")

[[1]]
<donation id="d1" amount="15000.00" donor="D4"/>

[[2]]
<donation id="d2" amount="10000.00" donor="D1"/>

[[3]]
<donation id="d3" amount="5383.73" donor="D5"/>

[[4]]
<donation id="d5" amount="2940.00" donor="D3"/>

attr(,"class")
[1] "XMLNodeSet"
```

```xml
<?xml version="1.0"?>
<ElectoralDonations>
  <party id="P2" name="National">
    <candidate id="C1" name="Amy" surname="ADAMS" electorate="E2">
      <donation id="d1" amount="15000.00" donor="D4"/>
      <donation id="d2" amount="10000.00" donor="D1"/>
    </candidate>
  </party>
  <party id="P1" name="Labour">
    <candidate id="C2" name="Glenda" surname="ALEXANDER" electorate="E3">
      <donation id="d3" amount="5383.73" donor="D5"/>
      <donation id="d4" amount="2000.00" donor="D6"/>
    </candidate>
    <candidate id="C3" name="Cliff" surname="ALLEN" electorate="E1">
      <donation id="d5" amount="2940.00" donor="D3"/>
      <donation id="d6" amount="2000.00" donor="D2"/>
    </candidate>
  </party>
  <donor id="D1" name="Douglas Catley (D H Catley Trust)"/>
  <donor id="D2" name="(Hamilton East Labour Electorate Committee)"/>
  <donor id="D3" name="J &amp; K Broughan"/>
  <donor id="D4" name="New Zealand National Party"/>
  <donor id="D5" name="Nordmeyer Trust"/>
  <donor id="D6" name="NZ Meatworkers Union"/>
  <electorate id="E1" name="Hamilton East"/>
  <electorate id="E2" name="Selwyn"/>
  <electorate id="E3" name="Waitaki"/>
</ElectoralDonations>
```

Figure 2: The XML file "`data.xml`"

```
2000,1,28,5,1647,1647,1906,1859,HP,154,N808AW,259,252,233,7,0,ATL,PHX,1587,15,11,0
2000,1,29,6,1648,1647,1939,1859,HP,154,N653AW,291,252,239,40,1,ATL,PHX,1587,5,47,0
2000,1,30,7,NA,1647,NA,1859,HP,154,N801AW,NA,252,NA,NA,NA,ATL,PHX,1587,0,0,1
2000,1,31,1,1645,1647,1852,1859,HP,154,N806AW,247,252,226,-7,-2,ATL,PHX,1587,7,14,0
2000,1,1,6,842,846,1057,1101,HP,609,N158AW,255,255,244,-4,-4,ATL,PHX,1587,3,8,0
2000,1,2,7,849,846,1148,1101,HP,609,N656AW,299,255,267,47,3,ATL,PHX,1587,8,24,0
2000,1,3,1,844,846,1121,1101,HP,609,N803AW,277,255,244,20,-2,ATL,PHX,1587,6,27,0
2000,1,1,6,1702,1657,1912,1908,HP,611,N652AW,250,251,232,4,5,ATL,PHX,1587,5,13,0
2000,1,2,7,1658,1657,1901,1908,HP,611,N807AW,243,251,233,-7,1,ATL,PHX,1587,3,7,0
2000,1,3,1,1656,1657,1922,1908,HP,611,N807AW,266,251,241,14,-1,ATL,PHX,1587,5,20,0
2000,1,4,2,1955,1932,2230,2153,HP,613,N509DC,275,261,232,37,23,ATL,PHX,1587,5,38,0
2000,1,5,3,1934,1932,2133,2153,HP,613,N509DC,239,261,224,-20,2,ATL,PHX,1587,5,10,0
2000,1,6,4,1929,1932,2125,2153,HP,613,N303AW,236,261,220,-28,-3,ATL,PHX,1587,5,11,0
2000,1,7,5,1932,1932,2146,2153,HP,613,N173AW,254,261,237,-7,0,ATL,PHX,1587,4,13,0
2000,1,9,7,2008,1932,2221,2153,HP,613,N168AW,253,261,237,28,36,ATL,PHX,1587,4,12,0
2000,1,10,1,1926,1932,2147,2153,HP,613,N160AW,261,261,235,-6,-6,ATL,PHX,1587,7,19,0
2000,1,11,2,1932,1932,2126,2153,HP,613,N160AW,234,261,217,-27,0,ATL,PHX,1587,6,11,0
2000,1,12,3,1936,1932,2142,2153,HP,613,N322AW,246,261,227,-11,4,ATL,PHX,1587,7,12,0
2000,1,13,4,1942,1932,2153,2153,HP,613,N160AW,251,261,220,0,10,ATL,PHX,1587,5,26,0
2000,1,14,5,1932,1932,2131,2153,HP,613,N314AW,239,261,218,-22,0,ATL,PHX,1587,6,15,0
```

Figure 3: The first few lines of the CSV file `"data.csv"`

.

6. Figure 3 shows the first few lines of a CSV file, `"data.csv"`. The complete file has 6,000,000 rows.

   Estimate the amount of memory that this data set would occupy if it was read into R using the following R code (and explain your reasoning).

   ```
   > data <- read.csv("data.csv", stringsAsFactors=FALSE)
   > # about 1GB
   > (18*8 + 4*8)*6000000
   
   [1] 1.056e+09
   ```

   Describe an alternative way to work with the data set in R that would require less memory.

   Store the data in a database, such as SQLite and access it using the DBI and RSQLite packages with something like ... `con <- dbConnect(RSQLite(), "data"); dbGetQuery(con, "SELECT * FROM data_table")`

7. Figure 4 shows some of the output from `top` on a Linux computer.

   How many CPU cores does this machine have? How much RAM does this machine have? How busy are the CPU cores? How much RAM is currently being used?

   The machine has two cores, neither of which is doing much (they are either 99% idle or 100% idle); the machine has 4GB of RAM and about 2.5GB of RAM is currently being used.

```
top - 10:19:02 up 38 days,  1:59,  3 users,  load average: 0.00, 0.01, 0.05
Tasks: 163 total,   1 running, 162 sleeping,   0 stopped,   0 zombie
Cpu0  :  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu1  :  0.3%us,  0.3%sy,  0.0%ni, 99.3%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   3973448k total,  2512664k used,  1460784k free,   408404k buffers
Swap:  4115452k total,   125816k used,  3989636k free,   945436k cached
```

Figure 4: The first few lines of output from `top` on a Linux machine.

8. Given the following bash commands and output ...

```
$ ls
2000.csv         data.json                data.xml~
Alan.docx        data-science-test.aux    full.txt
code-better.R    data-science-test.log    ideas.txt
code-better.R~   data-science-test.out    ideas.txt~
code-efficiency  data-science-test.pdf    medium.txt
code.R           data-science-test.Rnw    sample.json~
code.R~          data-science-test.Rnw~   sample.txt
data.csv         data-science-test.tex    Test779_2015.pdf
data.csv~        data.xml                 unused-question.Rnw

$ mkdir Temp

$ cp data-science-test.* Temp

$ cp unused-question.Rnw Temp

$ rm Temp/*.Rnw
```

... write down the result of the following bash command:

```
$ ls Temp
```

```
data-science-test.aux
data-science-test.log
data-science-test.out
data-science-test.pdf
data-science-test.Rnw
data-science-test.tex
```

The contents of the file `data.xml` are shown in Figure 2.

Write down the result of the following bash command (and explain what the output means):

```
$ grep party data.xml | wc
```

```
4 8 88
```

Which means that grep found `party` on 3 lines in `data.xml`, those 4 lines contained 8 words, and 88 characters.

9. Explain what the following R code is doing and what the output means.

```
> Rprof("test.out")

> replicate(5, mean(rnorm(1000000)))
[1] -0.0017922088 -0.0011004727 -0.0008793575  0.0017379549  0.0007257155

> Rprof(NULL)

> summaryRprof("test.out")
$by.self
        self.time self.pct total.time total.pct
"rnorm"      0.46      100       0.46       100

$by.total
            total.time total.pct self.time self.pct
"rnorm"           0.46       100      0.46      100
"FUN"             0.46       100      0.00        0
"lapply"          0.46       100      0.00        0
"mean"            0.46       100      0.00        0
"replicate"       0.46       100      0.00        0
"sapply"          0.46       100      0.00        0
```

The code is profiling an R expression to determine the amount of time spent in each R function during the call. The result is that basically all of the time is being spent in the `rnorm()` function (generating random numbers). By comparison, virtually none of the time is being spent in `mean()` or in the process of replication.

10. The following code runs a simple bootstrap permutation test using 10000 replications and measures how long it takes to run the test.

```
> diffs <- function(N) {
+     diffMean <- 1:N
+     for(i in diffMean){
+         GrpSample <- sample(Grp)
+         diffMean[i] <- diff(tapply(BP, GrpSample, mean))
+     }
+     diffMean
+ }
> set.seed(1000)
> BP <- rnorm(10, 100, 20)
> Grp <- rep(1:2, 5)
> system.time(diffs(10000))

   user  system elapsed
  1.208   0.004   1.213
```

Write R code to perform the 10000 replications in parallel on 4 cores. You can assume that the machine you are running on has at least 4 cores. Estimate how much time your code will take to run and explain your reasoning.

```
library(parallel)
mclapply(rep(2500, 4), diffs, num.cores=4)
```

I would expect this parallel version to take less time, but more than a quarter of the time (e.g., 0.4 seconds). This is because there are overheads involved in coordinating the parallel cores. There is also the possibility that other processes are running at the same time, so I would expect variability in my timing.