Emcien

Douglas Wirtz

June 2, 2016

Question 2

The following R code is used to answer the questions related to this prompt:

```
Let S be the set of numbers greater than zero and less than 100,000
that are evenly divisible by 19.

1. How many numbers are there in S?
2. How many numbers in S have a square that ends in a 1?
3. How many numbers in S have a reflection that is also in S? (The reflection
of 145 is 541)
4. How many numbers in S can be multiplied by some other number in S to
produce a third number in S?
```

To answer the questions, first we must create a data frame containing the set of numbers. That can be accomplished using this function:

```
s <- function(x, y, z){
#This function creates a dataset of numbers that is greater than the lower
bound(x), less than the
#upper bound(y), and evenly divisible by z.
    df <- NULL
    for (i in x:y){
        if(i %% z | i == 0){
            next
        }
        df <- rbind(df, data.frame(i))
    }
    df
}
```

Now we can input the parameters and set it equal to the variable `df`. Then we can look at the structure of the data frame and see how many numbers (observations) are in the set.

```
df <- s(0, 100000, 19)
str(df) #Question 1: 5263 numbers

## 'data.frame':    5263 obs. of  1 variable:
##  $ i: int  19 38 57 76 95 114 133 152 171 190 ...
```

We can answer the second question by using the following function. This function builds the data frame like the function above, but in addition, it builds a squared version of that data frame and searches for the squared values that end in "1". Then, a third data frame is created containing only the values of the original data frame whose squared value ends in "1".

```
s2 <- function(x, y, z){
#This function creates a dataset of numbers that is greater than the lower
bound(x), less than the
#upper bound(y), evenly divisible by z, and contains only numbers that have a
square ending in 1.
    df <- NULL
    output <- NULL
    for (i in x:y){
        if(i %% z | i == 0){
            next
        }
        df <- rbind(df, data.frame(i))
    }
    df2 <- df^2
    for (i in 1:nrow(df2)){
        row <- df2[i,]
        if(substr(df2[i,], floor(log10(df2[i,])) + 1, floor(log10(df2[i,])) +
1) == 1){
            output <- rbind(output, data.frame(df[i,]))
        }
    }
    output
}
```

Similar to above, input the parameters and set it equal to the variable `df2`. Look at the structure of the data frame and see how many numbers are in the set.

```
df2 <- s2(0, 100000, 19)
str(df2) #Question 2: 1053 numbers

## 'data.frame':    1053 obs. of  1 variable:
##  $ df.i...: int  19 171 209 361 399 551 589 741 779 931 ...
```

The next goal is to find how many numbers in the original data frame contain a reflection that also appears in the same frame. The following function will do this by building the data frame based on the parameters (x, y, z), then build a second frame containing only the values with reflections.

```
s3 <- function(x, y, z){
#This function creates a dataset of numbers that is greater than the lower
bound(x), less than the
#upper bound(y), evenly divisible by z, and contains only numbers that have a
reflection within the same frame.
    require(stringi)
    df <- NULL
```

```
    output <- NULL
    for (i in x:y){
        if(i %% z | i == 0){
            next
        }
        df <- rbind(df, data.frame(i))
    }
    for (i in 1:nrow(df)){
        row <- df[i,]
        for (j in 1:nrow(df)){
            row2 <- df[j,]
            if (row == stri_reverse(row2)){
                output <- rbind(output, data.frame(df[i,]))
            }
        }
    }
    output
}
```

Again, input the parameters and set it equal to the variable `df3`. Now we will look at the structure of the data frame see how many observations there are.

```
df3 <- s3(0, 100000, 19)

## Loading required package: stringi

str(df3) #Question 3: 250 numbers

## 'data.frame':    250 obs. of  1 variable:
##  $ df.i...: int  171 323 494 646 969 1881 2166 2318 2489 3553 ...
```

For the final question, we want to find out which numbers in the original data frame can be multiplied by some other number in the frame to produce a third number in the frame. Using the function below, we can input the parameters to see which numbers follow the criteria.

```
s4 <- function(x, y, z){
#This function creates a dataset of numbers that is greater than the lower
bound(x), less than the
#upper bound(y), evenly divisible by z, and contains only numbers that can be
multiplied by some
#other number in the set to produce a third number in the set.
    df <- NULL
    output <- NULL
    for (i in x:y){
        if(i %% z | i == 0){
            next
        }
        df <- rbind(df, data.frame(i))
    }
    #subset the data frame to shorten loops
```

```
    df <- subset.data.frame(df, (df$i*z) < y)
    r <- nrow(df)
    for (j in 1:r){
        row <- df[j,]
        for (k in 1:r){
            row2 <- df[k,]
            for (l in 1:r){
                row3 <- df[l,]
                if (row * row2 == row3){
                    output <- rbind(output, data.frame(df[j,]))
                }
            }
        }
    }
    output <- data.frame(output[!duplicated(output),])
    output
}
```

Lastly, we will set the function and the parameters equal to `df4` and look at its structure for the number of observations.

```
df4 <- s4(0, 100000, 19)
str(df4) #Question 4: 14 numbers
```

```
## 'data.frame':    14 obs. of  1 variable:
##  $ output..duplicated.output....: int  19 38 57 76 95 114 133 152 171 190
...
```
