[redacted]

[redacted]

by

Scott Greenberg

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

(a) (3 Points) Read in the table. Show the first 6 lines of this table. Load all R packages you need for the three questions from this homework in this question part. Often, you will only need 1 or 2 functions from a package. Show your R code.

```
> library(httr)
> library(XML)
> library(data.table)
> library(magrittr)
> doc <- GET("https://en.wikipedia.org/wiki/2016_Summer_Olympics_medal_table")
> pagetext <- content(doc, as = "text")
> tabs <- readHTMLTable(pagetext, header = TRUE)
> medal.tab <- tabs[[2]]
> head(medal.tab, 6)
```

	Rank	NOC	Gold	Silver	Bronze	Total
1	1	United States (USA)	46	37	38	121
2	2	Great Britain (GBR)	27	23	17	67
3	3	China (CHN)	26	18	26	70
4	4	Russia (RUS)	19	17	20	56
5	5	Germany (GER)	17	10	15	42
6	6	Japan (JPN)	12	8	21	41

(b) (8 Points) While the head of your table seems to look fine, what happens further down, e.g., for ranks 35, 51, etc.? Take a closer look at these ranks on the wikipedia page and you should understand what is going on. Use R code to fix this. Do do so, first understand the pattern of the incorrect rows, then adjust these rows. Try to avoid loops and rather work with apply functions whenever possible. Hints: (i) "is.na" allows you to check for missing entries. (ii) Your initial table likely consists of factors for each column. See `https://stackoverflow.com/questions/2851015/convert-data-frame-columns-from-factors-to-characters` how to translate them into characters. Show your final adjusted table (without the "Total" row – but keep Moldova) and your R code.

```
> medal.tab <- medal.tab[-1*length(medal.tab[, 1]), ]
> ties.medal <- is.na(medal.tab[, 6])
> medal.tab[ties.medal,] <-  medal.tab[ties.medal,c(6, 1:5)]
> medal.tab[, 1] <- nafill(as.integer(medal.tab[, 1]), "locf")
> medal.tab[, 3] <- as.integer(medal.tab[, 3])
> medal.tab[, 4] <- as.integer(medal.tab[, 4])
> medal.tab[, 5] <- as.integer(medal.tab[, 5])
> medal.tab[, 6] <- as.integer(medal.tab[, 6])
> medal.tab
```

	Rank	NOC	Gold	Silver	Bronze	Total
1	1	United States (USA)	46	37	38	121
2	2	Great Britain (GBR)	27	23	17	67
3	3	China (CHN)	26	18	26	70
4	4	Russia (RUS)	19	17	20	56
5	5	Germany (GER)	17	10	15	42
6	6	Japan (JPN)	12	8	21	41
7	7	France (FRA)	10	18	14	42
8	8	South Korea (KOR)	9	3	9	21
9	9	Italy (ITA)	8	12	8	28
10	10	Australia (AUS)	8	11	10	29
11	11	Netherlands (NED)	8	7	4	19
12	12	Hungary (HUN)	8	3	4	15
13	13	Brazil (BRA)*	7	6	6	19
14	14	Spain (ESP)	7	4	6	17
15	15	Kenya (KEN)	6	6	1	13
16	16	Jamaica (JAM)	6	3	2	11
17	17	Croatia (CRO)	5	3	2	10
18	18	Cuba (CUB)	5	2	4	11
19	19	New Zealand (NZL)	4	9	5	18
20	20	Canada (CAN)	4	3	15	22
21	21	Uzbekistan (UZB)	4	2	7	13
22	22	Kazakhstan (KAZ)	3	5	10	18
23	23	Colombia (COL)	3	2	3	8
24	24	Switzerland (SUI)	3	2	2	7
25	25	Iran (IRI)	3	1	4	8
26	26	Greece (GRE)	3	1	2	6
27	27	Argentina (ARG)	3	1	0	4
28	28	Denmark (DEN)	2	6	7	15
29	29	Sweden (SWE)	2	6	3	11
30	30	South Africa (RSA)	2	6	2	10
31	31	Ukraine (UKR)	2	5	4	11
32	32	Serbia (SRB)	2	4	2	8
33	33	Poland (POL)	2	3	6	11

34	34	North Korea (PRK)	2	3	2	7
35	35	Belgium (BEL)	2	2	2	6
36	35	Thailand (THA)	2	2	2	6
37	37	Slovakia (SVK)	2	2	0	4
38	38	Georgia (GEO)	2	1	4	7
39	39	Azerbaijan (AZE)	1	7	10	18
40	40	Belarus (BLR)	1	4	4	9
41	41	Turkey (TUR)	1	3	4	8
42	42	Armenia (ARM)	1	3	0	4
43	43	Czech Republic (CZE)	1	2	7	10
44	44	Ethiopia (ETH)	1	2	5	8
45	45	Slovenia (SLO)	1	2	1	4
46	46	Indonesia (INA)	1	2	0	3
47	47	Romania (ROU)	1	1	2	4
48	48	Bahrain (BRN)	1	1	0	2
49	48	Vietnam (VIE)	1	1	0	2
50	50	Chinese Taipei (TPE)	1	0	2	3
51	51	Bahamas (BAH)	1	0	1	2
52	51	Independent Olympic Athletes (IOA)	1	0	1	2
53	51	Ivory Coast (CIV)	1	0	1	2
54	54	Fiji (FIJ)	1	0	0	1
55	54	Jordan (JOR)	1	0	0	1
56	54	Kosovo (KOS)	1	0	0	1
57	54	Puerto Rico (PUR)	1	0	0	1
58	54	Singapore (SIN)	1	0	0	1
59	54	Tajikistan (TJK)	1	0	0	1
60	60	Malaysia (MAS)	0	4	1	5
61	61	Mexico (MEX)	0	3	2	5
62	62	Venezuela (VEN)	0	2	1	3
63	63	Algeria (ALG)	0	2	0	2
64	63	Ireland (IRL)	0	2	0	2
65	65	Lithuania (LTU)	0	1	3	4
66	66	Bulgaria (BUL)	0	1	2	3
67	67	India (IND)	0	1	1	2
68	67	Mongolia (MGL)	0	1	1	2
69	69	Burundi (BDI)	0	1	0	1
70	69	Grenada (GRN)	0	1	0	1
71	69	Niger (NIG)	0	1	0	1
72	69	Philippines (PHI)	0	1	0	1
73	69	Qatar (QAT)	0	1	0	1
74	74	Norway (NOR)	0	0	4	4
75	75	Egypt (EGY)	0	0	3	3
76	75	Tunisia (TUN)	0	0	3	3

```
77   77                      Israel (ISR)    0      0      2     2
78   78                     Austria (AUT)    0      0      1     1
79   78          Dominican Republic (DOM)    0      0      1     1
80   78                     Estonia (EST)    0      0      1     1
81   78                     Finland (FIN)    0      0      1     1
82   78                     Morocco (MAR)    0      0      1     1
83   78                     Nigeria (NGR)    0      0      1     1
84   78                    Portugal (POR)    0      0      1     1
85   78         Trinidad and Tobago (TTO)    0      0      1     1
86   78        United Arab Emirates (UAE)    0      0      1     1
```

(c) (8 Points) The country names consist of the name and the abbreviation of the country in parentheses e.g., "(ABC)". Use regular expressions and other means to extract the full country names from the "NOC" column. Make sure that there are no extra spaces at the end of a country name. Show the resulting vector with your country names (there should be 87) and your R code.

```
> gsub(".{6}$", "", medal.tab$NOC, perl = TRUE)
 [1] "United States"               "Great Britain"
 [3] "China"                       "Russia"
 [5] "Germany"                     "Japan"
 [7] "France"                      "South Korea"
 [9] "Italy"                       "Australia"
[11] "Netherlands"                 "Hungary"
[13] "Brazil "                     "Spain"
[15] "Kenya"                       "Jamaica"
[17] "Croatia"                     "Cuba"
[19] "New Zealand"                 "Canada"
[21] "Uzbekistan"                  "Kazakhstan"
[23] "Colombia"                    "Switzerland"
[25] "Iran"                        "Greece"
[27] "Argentina"                   "Denmark"
[29] "Sweden"                      "South Africa"
[31] "Ukraine"                     "Serbia"
[33] "Poland"                      "North Korea"
[35] "Belgium"                     "Thailand"
[37] "Slovakia"                    "Georgia"
[39] "Azerbaijan"                  "Belarus"
[41] "Turkey"                      "Armenia"
[43] "Czech Republic"              "Ethiopia"
[45] "Slovenia"                    "Indonesia"
[47] "Romania"                     "Bahrain"
[49] "Vietnam"                     "Chinese Taipei"
[51] "Bahamas"                     "Independent Olympic Athletes"
[53] "Ivory Coast"                 "Fiji"
```

```
[55] "Jordan"                       "Kosovo"
[57] "Puerto Rico"                  "Singapore"
[59] "Tajikistan"                   "Malaysia"
[61] "Mexico"                       "Venezuela"
[63] "Algeria"                      "Ireland"
[65] "Lithuania"                    "Bulgaria"
[67] "India"                        "Mongolia"
[69] "Burundi"                      "Grenada"
[71] "Niger"                        "Philippines"
[73] "Qatar"                        "Norway"
[75] "Egypt"                        "Tunisia"
[77] "Israel"                       "Austria"
[79] "Dominican Republic"           "Estonia"
[81] "Finland"                      "Morocco"
[83] "Nigeria"                      "Portugal"
[85] "Trinidad and Tobago"          "United Arab Emirates"
```

(d) (8 Points) Similar to the question above, use regular expressions and other means to extract the country abbreviations from the "NOC" column. Make sure that there are no extra spaces or symbols at the end of a country abbreviation. Show the resulting vector with your country abbreviations (there should be 87) and your R code.

```
> gsub("^[^()]*|[()]", "", medal.tab$NOC, perl = TRUE)
```

```
 [1] "USA"  "GBR"  "CHN"  "RUS"  "GER"  "JPN"  "FRA"  "KOR"  "ITA"  "AUS"
[11] "NED"  "HUN"  "BRA*" "ESP"  "KEN"  "JAM"  "CRO"  "CUB"  "NZL"  "CAN"
[21] "UZB"  "KAZ"  "COL"  "SUI"  "IRI"  "GRE"  "ARG"  "DEN"  "SWE"  "RSA"
[31] "UKR"  "SRB"  "POL"  "PRK"  "BEL"  "THA"  "SVK"  "GEO"  "AZE"  "BLR"
[41] "TUR"  "ARM"  "CZE"  "ETH"  "SLO"  "INA"  "ROU"  "BRN"  "VIE"  "TPE"
[51] "BAH"  "IOA"  "CIV"  "FIJ"  "JOR"  "KOS"  "PUR"  "SIN"  "TJK"  "MAS"
[61] "MEX"  "VEN"  "ALG"  "IRL"  "LTU"  "BUL"  "IND"  "MGL"  "BDI"  "GRN"
[71] "NIG"  "PHI"  "QAT"  "NOR"  "EGY"  "TUN"  "ISR"  "AUT"  "DOM"  "EST"
[81] "FIN"  "MAR"  "NGR"  "POR"  "TTO"  "UAE"
```

(e) (3 Points) Create a simple boxplot for the total number of medals won by these 87 countries. Choose a horizontal layout and add labels and titles (as needed). Are there any outliers? Which? Keep in mind that you (likely) have characters (or factor levels) in your data frame when you create your boxplot and have to transfer these to integer first. Show the final optimized boxplot and your R code.

```
> boxplot(medal.tab$Total, horizontal = TRUE, xlab = "Number of Medals",
+         main = "Medal count of listed countries in 2016 Summer Olympics")
> print("Outliers")
```

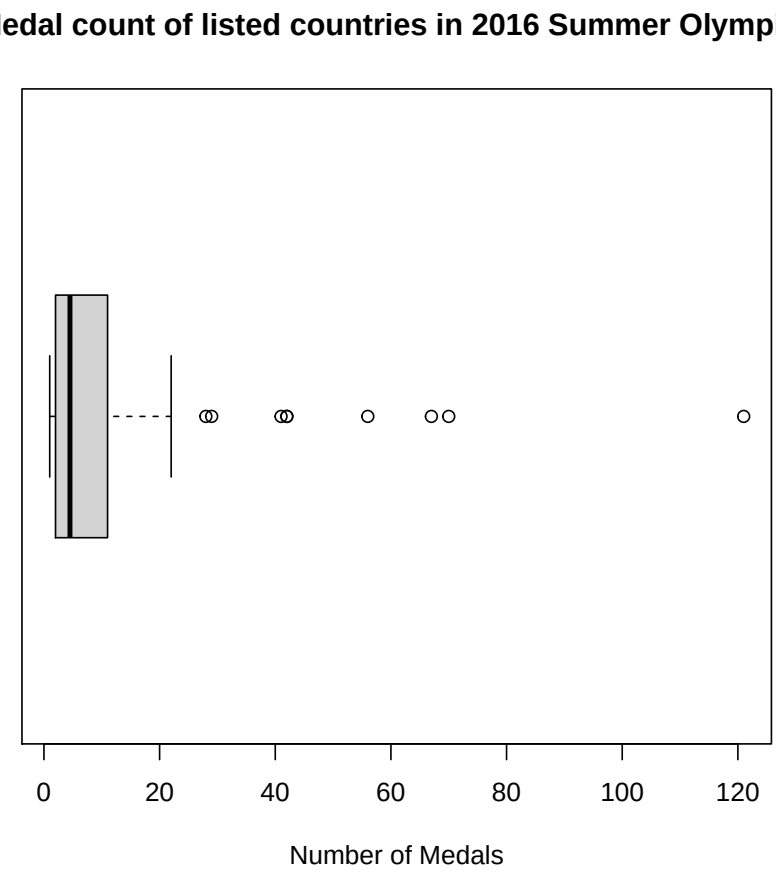
```
[1] "Outliers"
```

```
> medal.tab$NOC[medal.tab$Total > min(boxplot.stats(medal.tab$Total)$out)]
```

```
[1] "United States (USA)" "Great Britain (GBR)" "China (CHN)"
```

```
[4] "Russia (RUS)"        "Germany (GER)"       "Japan (JPN)"
[7] "France (FRA)"        "Australia (AUS)"

>
```



[redacted]

(a) (4 Points) Use OCR to obtain the text from these two jpg images. Combine both text strings into a single one for further processing. Show the resulting text string and your R code.

```
> path.one <- "C:/Users/Scott/Downloads/Olympics_1.jpg"
> path.two <- "C:/Users/Scott/Downloads/Olympics_2.jpg"
> library(tesseract)
> library(dplyr)
> ocr.one <- ocr(path.one)
> ocr.two <- ocr(path.two)
> ocr.olympics <- paste0(ocr.one, ocr.two)
> ocr.olympics
```

```
[1] "■ LA 2028 5 BEIWING 2008 2 SEOUL 1988 &B MEXICO 1968 shy LONDON 1948\n\na PARIS 2024 a ATH
```

(b) (6 Points) Use regular expressions to obtain the city names and years from the text string obtained in (a). Note that some characters may not have been recognized correctly. So, you may have to use a more general regular expression than what would seem to be needed for the correct city names and years. Show the resulting text (31 city names/years) and your R code.

```
> ocr.olympics.cities <- strsplit(gsub("[0-9]{2,}", "00",
+                                      ocr.olympics, perl = TRUE), "00") %>%
+                        unlist()
> ocr.olympics.years <-  strsplit(gsub("[A-Z]{2,}[^[0-9]*]*|[a-z]{5} *",
+                                      "seperator",
+                                      ocr.olympics, perl = TRUE),
+                                "seperator") %>%
+                        unlist()
> ocr.olympics.cities <- ocr.olympics.cities[-1*length(ocr.olympics.cities)]
> ocr.olympics.years <- ocr.olympics.years[-1]
> mapply(paste, ocr.olympics.cities, ocr.olympics.years,
+        SIMPLIFY = TRUE, USE.NAMES = FALSE)
```

```
 [1] "■ LA  2028 5 "
 [2] " 5 BEIWING  2008 2 "
 [3] " 2 SEOUL  1988 &B "
 [4] " &B MEXICO  1968 shy "
 [5] " shy LONDON  1948\n\na "
 [6] "\n\na PARIS  2024 a "
 [7] " a ATHENS  2004 cs "
 [8] " cs LOS ANGELES  1984 @ "
 [9] " @ TOKYO  1964 Al "
[10] " Al BERLIN  1936\n\n© =\n\n‘"
```

```
[11] "\n\n© =\n\n‘TOKYO  2020 =| "
[12] " =| SYDNEY  2000 "
[13] " MOSCOW  1980 mS "
[14] " mS ROME  1960 oe "
[15] " oe LOS ANGELES  1932\n& | 6 "
[16] "\n& | 6 MELBOURNE/ g\n\ncose RIO  2016 "
[17] " ATLANTA  1996 ae "
[18] " ae MONTREAL  1976 | "
[19] " | STOGKHOLM‘ 656 "
[20] " AMSTERDAM  1928\nD4 "
[21] "\nD4 LONDON  2012 a "
[22] " a BARCELONA  1992 sm "
[23] " sm MUNICH  1972 1 -| "
[24] " 1 -| HELSINKI  1952 \""
[25] " \"paris  1924\n8 "
[26] "\n8 ANTWERP  1920 g "
[27] " g ATHENS  1896\n2 "
[28] "\n2 STOCKHOLM  1912\n\n8 "
[29] "\n\n8 LONDON  1908\n\n"
[30] "\n\nST. LOUIS  1904\n\n4s "
[31] "\n\n4s PARIS  1900\n"

>
>
```

dholstius (2013)

(c) (6 Points) Further process the city names and years obtained in (b): Remove empty strings or strings with just spaces (if any), remove spaces before and after the city names/year, and remove (sequences of) single letters (followed by a space) at the start of a city name. As "LA 2028" is a valid city name/year combination, you likely have two other invalid city name/year combinations. Also correct these. Use regular expressions and substitutions as needed. Show the resulting text (31 city names/years) and your R code.

```
> ocr.olympics.cities <- gsub("[^A-Z a-z/.]{1,}", "", ocr.olympics.cities)
> ocr.olympics.cities <- gsub("(^| )[a-z]{1,4} |(^|[^A-Z]*)[A-Z]{1}[^A-Z]+",
+                             "", ocr.olympics.cities)
> ocr.olympics.cities <- gsub("^ *| *$", "", ocr.olympics.cities)
> ocr.olympics.cities <- gsub(" [a-z]{5} ", " ", ocr.olympics.cities)
> ocr.olympics.years <- gsub("[^0-9].*", "", ocr.olympics.years)
> mapply(paste, ocr.olympics.cities, ocr.olympics.years, SIMPLIFY = TRUE,
+        USE.NAMES = FALSE)
 [1] "LA 2028"             "BEIWING 2008"        "SEOUL 1988"
 [4] "MEXICO 1968"         "LONDON 1948"         "PARIS 2024"
 [7] "ATHENS 2004"         "LOS ANGELES 1984"    "TOKYO 1964"
[10] "BERLIN 1936"         "TOKYO 2020"          "SYDNEY 2000"
```

```
[13] "MOSCOW 1980"          "ROME 1960"           "LOS ANGELES 1932"
[16] "MELBOURNE/ RIO 2016" "ATLANTA 1996"        "MONTREAL 1976"
[19] "STOGKHOLM 656"        "AMSTERDAM 1928"      "LONDON 2012"
[22] "BARCELONA 1992"       "MUNICH 1972"         "HELSINKI 1952"
[25] "paris 1924"           "ANTWERP 1920"        "ATHENS 1896"
[28] "STOCKHOLM 1912"       "LONDON 1908"         "ST. LOUIS 1904"
[31] "PARIS 1900"
```

dholstius (2013)

(d) (6 Points) Create a data frame that consists of the city names and the year, based on your resulting text from (c). Sort the rows in ascending order of the years, i.e., starting with 1896. Show the resulting data frame (31 rows, 2 columns) and your R code.

```
> arrange(data.frame(cities = ocr.olympics.cities,
+                    years = as.numeric(ocr.olympics.years)), years)
```

	cities	years
1	STOGKHOLM	656
2	ATHENS	1896
3	PARIS	1900
4	ST. LOUIS	1904
5	LONDON	1908
6	STOCKHOLM	1912
7	ANTWERP	1920
8	paris	1924
9	AMSTERDAM	1928
10	LOS ANGELES	1932
11	BERLIN	1936
12	LONDON	1948
13	HELSINKI	1952
14	ROME	1960
15	TOKYO	1964
16	MEXICO	1968
17	MUNICH	1972
18	MONTREAL	1976
19	MOSCOW	1980
20	LOS ANGELES	1984
21	SEOUL	1988
22	BARCELONA	1992
23	ATLANTA	1996
24	SYDNEY	2000
25	ATHENS	2004
26	BEIWING	2008
27	LONDON	2012
28	MELBOURNE/ RIO	2016
29	TOKYO	2020

```
30          PARIS  2024
31             LA  2028
```

(e) (4 Points) Look carefully at your results in (d). Does everything match the original two jpg images? If not, further adjust the data frame you got. I had to do three adjustments of the city names and two adjustments of the years on my side and also adjust cases (to upper). When you run into an error here, you may use factors in your data frame. Replace these with character strings or immediately create a data frame that consists of character strings in step (d). Resort your data frame once more (if necessary). Show the resulting data frame (31 rows, 2 columns) and your R code.

```
> ocr.olympics.cities[2] <- "BEIJING"
> ocr.olympics.cities[16] <- "RIO"
> ocr.olympics.cities[19] <- "MELBOURNE/STOCKHOLM"
> ocr.olympics.cities[25] <- "PARIS"
> ocr.olympics.years[19] <- "1956"
> ocr.olympics.years <- as.integer(ocr.olympics.years)
> olympic.df <- arrange(data.frame(city = ocr.olympics.cities,
+                                  year = ocr.olympics.years), year)
> olympic.df
```

```
                   city year
1               ATHENS 1896
2                PARIS 1900
3            ST. LOUIS 1904
4               LONDON 1908
5            STOCKHOLM 1912
6              ANTWERP 1920
7                PARIS 1924
8            AMSTERDAM 1928
9          LOS ANGELES 1932
10              BERLIN 1936
11              LONDON 1948
12            HELSINKI 1952
13 MELBOURNE/STOCKHOLM 1956
14                ROME 1960
15               TOKYO 1964
16              MEXICO 1968
17              MUNICH 1972
18            MONTREAL 1976
19              MOSCOW 1980
20         LOS ANGELES 1984
21               SEOUL 1988
22           BARCELONA 1992
23             ATLANTA 1996
24              SYDNEY 2000
```

25	ATHENS 2004
26	BEIJING 2008
27	LONDON 2012
28	RIO 2016
29	TOKYO 2020
30	PARIS 2024
31	LA 2028

[redacted]

(a) (6 Points) Use your final data frame from Question (ii) to create the URL for the 28 past Olympics (1896 to 2016). Immediately replace spaces (" "), dashes ("/"), and periods (".") in the city names with a single hyphen ("-"). "Melbourne/Stockholm-1956" may be tricky! Check whether these adjusted URLs exist and indicate the results. Manually (in R) adjust those URLs that still do not exist. Show your R code.

```
> olympic.list <- mapply(paste, as.list(olympic.df[, 1]),
+                        as.list(olympic.df[, 2]), SIMPLIFY = TRUE,
+                        USE.NAMES = FALSE)
> olympic.list <- olympic.list[-31]
> olympic.list <- olympic.list[-30]
> olympic.list <- olympic.list[-29]
> olympic.list <- gsub("[^A-Z0-9]+","-", olympic.list)
> olympic.list <- gsub("^", "https://WWW.Olympic.org/", olympic.list)
> olympic.list
 [1] "https://WWW.Olympic.org/ATHENS-1896"
 [2] "https://WWW.Olympic.org/PARIS-1900"
 [3] "https://WWW.Olympic.org/ST-LOUIS-1904"
 [4] "https://WWW.Olympic.org/LONDON-1908"
 [5] "https://WWW.Olympic.org/STOCKHOLM-1912"
 [6] "https://WWW.Olympic.org/ANTWERP-1920"
 [7] "https://WWW.Olympic.org/PARIS-1924"
 [8] "https://WWW.Olympic.org/AMSTERDAM-1928"
 [9] "https://WWW.Olympic.org/LOS-ANGELES-1932"
[10] "https://WWW.Olympic.org/BERLIN-1936"
[11] "https://WWW.Olympic.org/LONDON-1948"
[12] "https://WWW.Olympic.org/HELSINKI-1952"
[13] "https://WWW.Olympic.org/MELBOURNE-STOCKHOLM-1956"
[14] "https://WWW.Olympic.org/ROME-1960"
[15] "https://WWW.Olympic.org/TOKYO-1964"
[16] "https://WWW.Olympic.org/MEXICO-1968"
[17] "https://WWW.Olympic.org/MUNICH-1972"
[18] "https://WWW.Olympic.org/MONTREAL-1976"
[19] "https://WWW.Olympic.org/MOSCOW-1980"
[20] "https://WWW.Olympic.org/LOS-ANGELES-1984"
[21] "https://WWW.Olympic.org/SEOUL-1988"
[22] "https://WWW.Olympic.org/BARCELONA-1992"
[23] "https://WWW.Olympic.org/ATLANTA-1996"
[24] "https://WWW.Olympic.org/SYDNEY-2000"
```

```
[25] "https://WWW.Olympic.org/ATHENS-2004"
[26] "https://WWW.Olympic.org/BEIJING-2008"
[27] "https://WWW.Olympic.org/LONDON-2012"
[28] "https://WWW.Olympic.org/RIO-2016"
```

dholstius (2013)

(b) (10 Points) First look carefully at the source code of the web pages from the 28 past Olympics (1896 to 2016) to understand the structure of these pages. We want to extract the date, host country, number of participating athletes, number of participating countries, and number of events from each of these 28 web pages. Use any of the methods discussed in class to automatically extract this information. Add your extracted data to the data frame from Question (ii) that contains the city names and years (but only for the 28 past Olympics). The URL `https://stackoverflow.com/questions/24958490/how-to-convert-html-lists-into-data-frame-in-r` may provide you with some useful hints how to extract the data from these web pages. Hints: (i) Start with a very general path and then fill in additional path elements and attributes to specify the location of these data. This will require some work and fine tuning. When you obtain NULL or an empty list, your path (or attribute) very likely is incorrect; (ii) Start with a single Olympic, then test your code for 1 or 2 more Olympics. Finally run your code for all Olympics; (iii) Be careful with the 1956 Olympics where the data appears in a (slightly) different format; (iv) For convenience, a "for" loop is OK here. Show the resulting data frame (28 rows, 7 columns) and your R code.

```
> d.xpath <- "//*[@id=\"wrapper\"]/div[1]/section[2]/div/div[1]/ul/li[1]/div"
> date.list <- lapply(1:length(olympic.list), function(i) {
+   gsub("^[^0-9]*|[^a-z]*$", "",
+        as.character(xmlValue(getNodeSet(htmlParse(GET(olympic.list[i])),
+                                         d.xpath)[[1]][3][[1]])))
+
+ })
> date <- unlist(date.list)
> h.xpath <- "//*[@id=\"wrapper\"]/div[1]/section[2]/div/div[1]/ul/li[2]/div/a"
> host.list <- lapply(1:length(olympic.list), function(i) {
+    host.node <- getNodeSet(htmlParse(GET(olympic.list[i])), h.xpath)
+    host.value <- integer(length(host.node))
+    for (j in 1:length(host.node)){
+      host.value[j] <- as.character(xmlValue(host.node[[j]][1][[1]]))
+      host <- paste(host.value[j-1], host.value[j], sep = ", ")
+    }
+    gsub("^, ","", host)
+ })
> host <- unlist(host.list)
> a.xpath <- "//*[@id=\"wrapper\"]/div[1]/section[2]/div/div[1]/ul/li[3]/div"
> num.ath.list <- lapply(1:length(olympic.list), function(i) {
+   gsub("[^0-9]", "",
```

```
+        as.character(xmlValue(getNodeSet(htmlParse(GET(olympic.list[i])),
+                                         a.xpath)[[1]][3][[1]])))
+
+ })
> num.ath <- as.integer(unlist(num.ath.list))
> c.xpath <- "//*[@id=\"wrapper\"]/div[1]/section[2]/div/div[1]/ul/li[4]/div"
> num.countries.list <- lapply(1:length(olympic.list), function(i) {
+   gsub("[^0-9]", "",
+        as.character(xmlValue(getNodeSet(htmlParse(GET(olympic.list[i])),
+                                         c.xpath)[[1]][3][[1]])))
+
+ })
> num.countries <- as.integer(unlist(num.countries.list))
> e.xpath <- "///*[@id=\"wrapper\"]/div[1]/section[2]/div/div[1]/ul/li[5]/div"
> num.events.list <- lapply(1:length(olympic.list), function(i) {
+   gsub("[^0-9]", "",
+        as.character(xmlValue(getNodeSet(htmlParse(GET(olympic.list[i])),
+                                         e.xpath)[[1]][3][[1]])))
+
+ })
> num.events <- as.integer(unlist(num.events.list))
> past.olympic.df <- mutate(olympic.df[-29:-31, ], date = date, country = host,
+                           "number of athletes" = num.ath,
+                           "number of countries" = num.countries,
+                           "number of events" = num.events)
> past.olympic.df
```

	city	year	date	country
1	ATHENS	1896	06 Apr - 15 Apr	Greece
2	PARIS	1900	14 May - 28 Oct	France
3	ST. LOUIS	1904	01 Jul - 23 Nov	United States of America
4	LONDON	1908	27 Apr - 31 Oct	Great Britain
5	STOCKHOLM	1912	05 May - 27 Jul	Sweden
6	ANTWERP	1920	20 Apr - 12 Sep	Belgium
7	PARIS	1924	04 May - 27 Jul	France
8	AMSTERDAM	1928	17 May - 12 Aug	Netherlands
9	LOS ANGELES	1932	30 Jul - 14 Aug	United States of America
10	BERLIN	1936	01 Aug - 16 Aug	Germany
11	LONDON	1948	29 Jul - 14 Aug	Great Britain
12	HELSINKI	1952	19 Jul - 03 Aug	Finland
13	MELBOURNE/STOCKHOLM	1956	22 Nov - 08 Dec	Australia, Sweden
14	ROME	1960	25 Aug - 11 Sep	Italy
15	TOKYO	1964	10 Oct - 24 Oct	Japan
16	MEXICO	1968	12 Oct - 27 Oct	Mexico

17	MUNICH	1972	26 Aug - 11 Sep	Germany
18	MONTREAL	1976	17 Jul - 01 Aug	Canada
19	MOSCOW	1980	19 Jul - 03 Aug	USSR
20	LOS ANGELES	1984	28 Jul - 12 Aug	United States of America
21	SEOUL	1988	17 Sep - 02 Oct	Republic of Korea
22	BARCELONA	1992	25 Jul - 09 Aug	Spain
23	ATLANTA	1996	19 Jul - 04 Aug	United States of America
24	SYDNEY	2000	15 Sep - 01 Oct	Australia
25	ATHENS	2004	13 Aug - 29 Aug	Greece
26	BEIJING	2008	08 Aug - 24 Aug	People's Republic of China
27	LONDON	2012	27 Jul - 12 Aug	Great Britain
28	RIO	2016	05 Aug - 21 Aug	Brazil

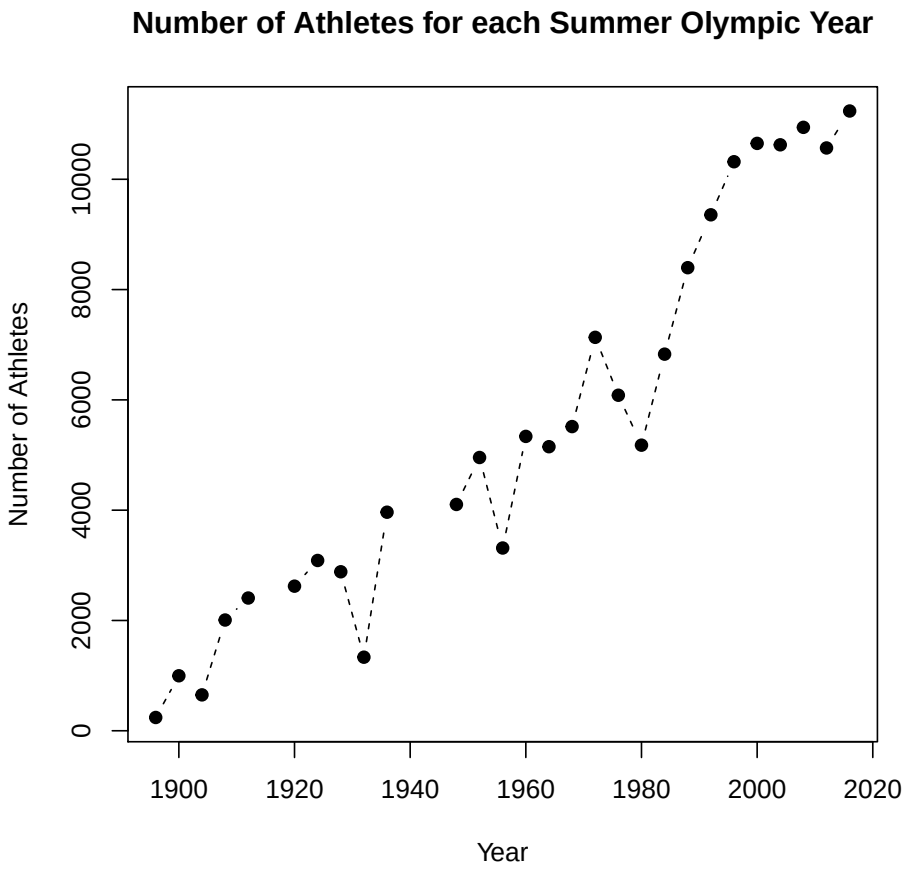
	number of athletes	number of countries	number of events
1	241	14	43
2	997	24	95
3	651	12	95
4	2008	22	110
5	2407	28	102
6	2622	29	156
7	3088	44	126
8	2883	46	109
9	1334	37	117
10	3963	49	129
11	4104	59	136
12	4955	69	149
13	3314	72	151
14	5338	83	150
15	5151	93	163
16	5516	112	172
17	7134	121	195
18	6084	92	198
19	5179	80	203
20	6829	140	221
21	8397	159	237
22	9356	169	257
23	10318	197	271
24	10651	199	300
25	10625	201	301
26	10942	204	302
27	10568	204	302
28	11238	207	306

Mischa Vreeburg (2011), Maurits Evers (2018)

(c) (6 Points) Draw three simple time series (i.e., scatterplots with year on the horizontal axis

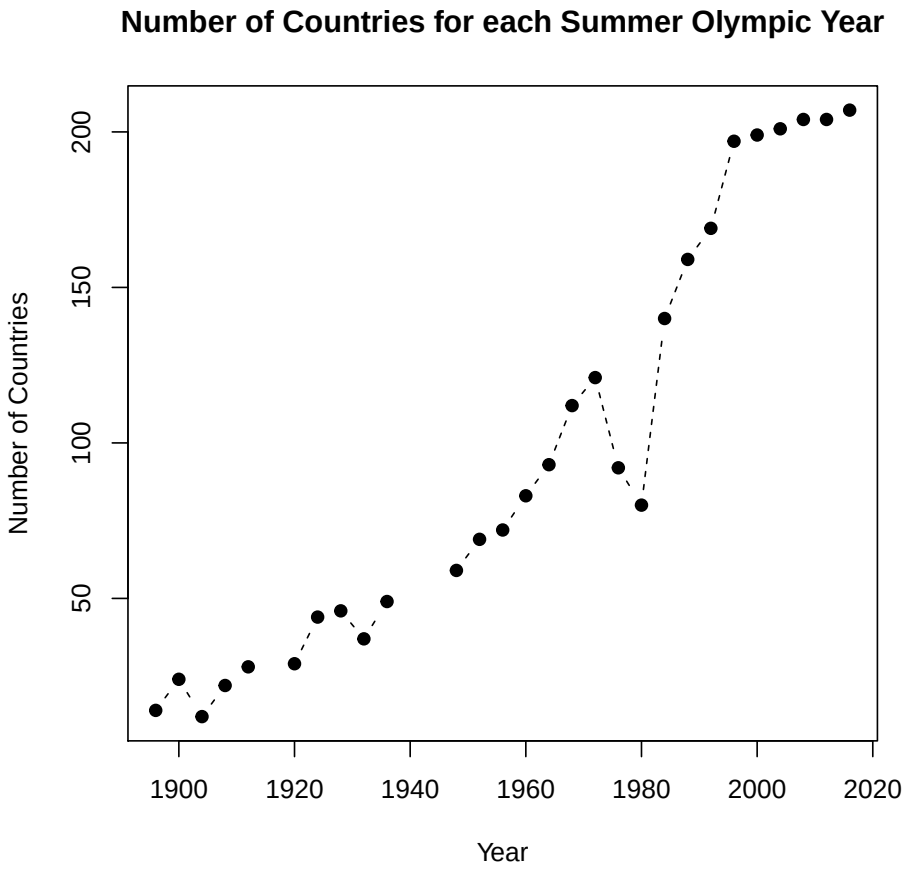
and the variable of interest on the vertical axis) that show the development of number of participating athletes, number of participating countries, and number of events during these past 28 Olympics. Use a solid filled circle for the data points. Connect data points from consecutive Olympics with a dashed line, but do not connect data points when there was a gap of 8 or 12 years as was the case during World Wars I & II. Hint: You can draw your line in segments, leaving out those parts that are further than 4 years apart. Or you could extend your data frame with the missing years and use NA for the counts. Make sure to resort your data frame. There is nothing to draw when something is missing! Show the three resulting time series and your R code. Use these time series as a plausibility check for your extracted data. If something looks strange, revisit the underlying Olympics web page and correct your R code/data extraction from this page.

```
> plot.past.olympic.df <- merge(data.frame(year = ((1896/4):(2016/4))*4),
+                               past.olympic.df, all.x = TRUE)
> plot(plot.past.olympic.df$year, plot.past.olympic.df$`number of athletes`,
+      main = "Number of Athletes for each Summer Olympic Year", xlab = "Year",
+      ylab = "Number of Athletes", type = 'b', lty = 2, pch = 19)
>
```

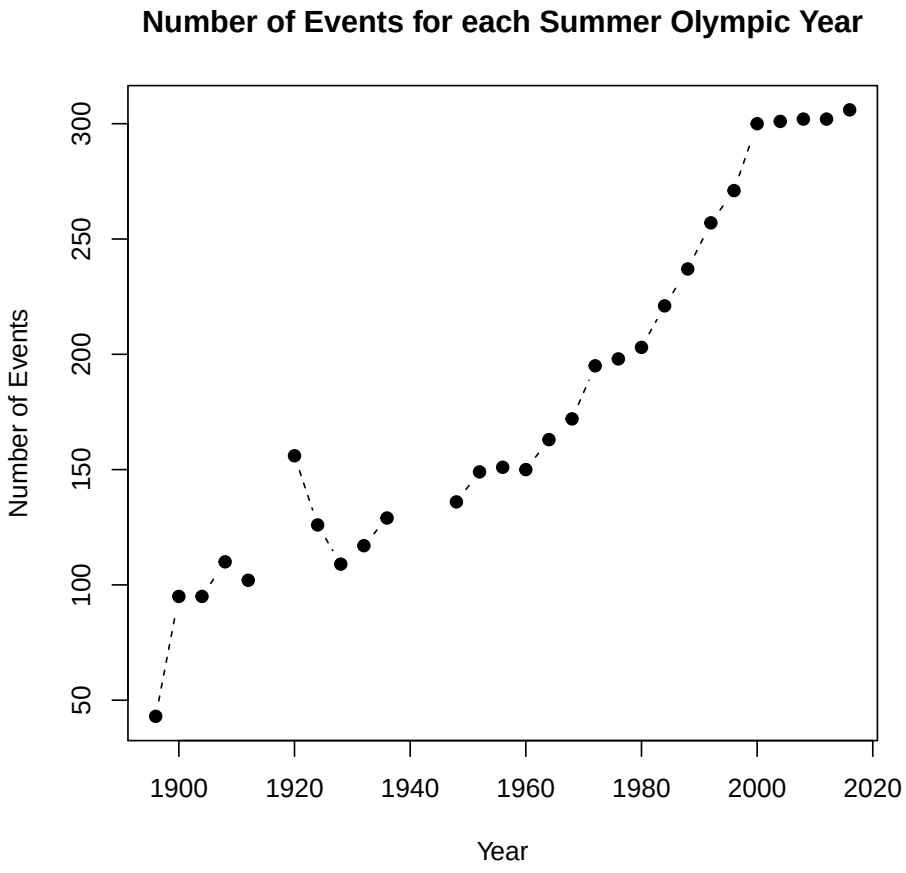



```
> plot(plot.past.olympic.df$year, plot.past.olympic.df$`number of countries`,
```

```
+      main = "Number of Countries for each Summer Olympic Year", xlab = "Year",
+      ylab = "Number of Countries", type = 'b', lty = 2, pch = 19)
>
>
```




```
> plot(plot.past.olympic.df$year, plot.past.olympic.df$`number of events`,
+      main = "Number of Events for each Summer Olympic Year", xlab = "Year",
+      ylab = "Number of Events", type = 'b', lty = 2, pch = 19)
>
```


Number of Events for each Summer Olympic Year
Number of Events
50
100
150
200
250
300
1900
1920
1940
1960
1980
2000
2020
Year

General Instructions

(i) Create a single html or pdf document, using R Markdown, Sweave, or knitr. You only have to submit this one document.

(ii) Include a title page that contains your name, your A-number, the number of the assignment, the submission date, and any other relevant information.

(iii) Start your answers to each main question on a new page (continuing with the next part of a question on the same page is fine). Clearly label each question and question part.

(iv) Before you submit your homework, check that you follow all recommendations from Google's R Style Guide (see `https://google.github.io/styleguide/Rguide.xml`). Moreover, make sure that your R code is consistent, i.e., that you use the same type of assignments and the same type of quotes throughout your entire homework.

(v) Give credit to external sources, such as stackoverflow or help pages. Be specific and include the full URL where you found the help (or from which help page you got the information). Consider R code from such sources as "legacy code or third-party code" that does not have to be adjusted to Google's R Style (even though it would be nice, in particular if you only used a brief code segment).

(vi) **Not following the general instructions outlined above will result in point deductions!**

(vii) For general questions related to this homework, please use the corresponding discussion board in Canvas! I will try to reply as quickly as possible. Moreover, if one of you knows an answer, please post it. It is fine to refer to web pages and R commands, but do not provide the exact R command with all required arguments or which of the suggestions from a stackoverflow web page eventually worked for you! This will be the task for each individual student!

(viii) Submit your single html or pdf file via Canvas by the submission deadline. Late submissions will result in point deductions as outlined on the syllabus.

References

dholstius, . h. . (2013), 'elementwise combination of two lists in r', Stack Overflow. URL: https://stackoverflow.com/a/16753614 (version: 2020-01-28).

Maurits Evers, . h.-e. . (2018), 'Better alternatives for loop r', Stack Overflow. URL:https://stackoverflow.com/a/51393493 (version: 2021-01-28).

Mischa Vreeburg, . h.-v. . (2011), ''externalptr' error in r using xml data', Stack Overflow. URL: https://stackoverflow.com/a/8384940 (version: 2020-01-28).