Przygotowywanie danych pomiarowych

1. Uruchom RStudio.
2. Ustaw swój Working Directory używając polecenia setwd().

```
setwd("./inazwisko")
```

3. Sprawdź czy istnieje katalog data. Jeśli nie istnieje – utwórz katalog, jeśli istnieje – wypisz wszystkie pliki.

```
if(!file.exists("./data")){
      dir.create("./data")
} else {
      list.files()
}
```

4. Na stronie http://monitoring.krakow.pios.gov.pl/ wybierz następującą ścieżkę: dane pomiarowe > dane aktualne > Automatyczne, a następnie wybiera raport miesięczny z 01.2016, stację Kraków-Kurdwanów. Po zastosowaniu filtrów wyświetli się strona o adresie: http://monitoring.krakow.pios.gov.pl/dane-pomiarowe/automatyczne/stacja/16/parametry/144-149-146-148-242-143-145/miesieczny/01.2016. Nad tabelą znajduje się przycisk „eksportuj do: CSV". Pobierz plik i zapisz go w katalogu data znajdującym się w Twoim Working Directory jako plik data1.csv.
5. Następnie wybierz raport miesięczny 02.2016 ze stacji Kraków-Kurdwanów. Zapisz CSV w katalogu data znajdującym się w Twoim Working Directory jako plik data2.csv
6. Wczytaj plik z danymi pomiarowymi ze stycznia i zapisz jako dataframe o nazwie data1.

```
data1 <- read.table("./data/data1.csv", sep=";", header=TRUE, nrow=
31)
```

7. Wczytaj plik z danymi pomiarowymi z lutego i zapisz jako dataframe o nazwie data2.

```
data2 <- read.table("./data/data2.csv", sep=";", header=TRUE, nrow=
29)
```

8. Połącz wczytane dane zapisując je jako data. Do połączenia danych wykorzystaj funkcję rbind.

```
data <- rbind(data1,data2)
```

9. Wypisz i przeanalizuj nazwy kolumn, a następnie zmień nazwy na bardziej czytelne i wypisz pierwsze wiersze.

```
names(data)

names(data) <-
c("day.month","SO2","NO2","NOx","NO","O3","O38h","PM10","PM25")

head(data)
```

10. Wyświetl podsumowanie danych za pomocą poniższych funkcji.

```
summary(data)
```

```
str(data)

quantile(data$PM25,na.rm=T, probs=seq(0,1,0.1))
```

11. Wyświetl typ danych poszczególnych kolumn.

```
for(i in names(data)) print(class(data[[i]]))
```

12. Zmodyfikuj typ kolumn „day.month” oraz „SO2” na character, co ułatwi nam dalszą procedurę czyszczenia danych.

```
data[[1]] <- as.character(data[[1]])

data[[2]] <- as.character(data[[2]])
```

13. W kolumnie SO2 zamień znak przecinka na kropkę. Do modyfikacji wykorzystujemy wyrażenia regularne. W kolejnym kroku dokonujemy konwersji typu kolumny na typ numeric.

```
data$SO2 <- gsub(",",".",data$SO2)

data$SO2 <- as.numeric(data$SO2)
```

14. Wykorzystując funkcje strsplit podziel kolumnę day.month na 2 kolumny day oraz month.

```
for(i in 1:nrow(data)) {

      tmp <- strsplit(data[i,"day.month"],"\\.")[[1]]

      data[i,"day"] <- tmp[1]

      data[i,"month"] <- tmp[2]

}
```

15. Dodaj kolumnę „year”, w której zostanie wpisana wartość 2016.

```
data["year"] <- 2016
```

16. Usuń kolumnę „day.month”.

```
data <- data[,-1]
```

17. Stwórz nową kolumnę, w której zapiszesz datę każdego pomiaru wykorzystując kolumny day, month oraz year. Następnie usuń wykorzystane kolumny.

```
data$date <- as.Date(paste(data$year, data$month, data$day, sep="-
"))

data <- data[,-(9:11)]
```

18. Przeanalizuj działanie poniższych funkcji związanych z typem Date.

```
today <- Sys.Date()

days <- julian(today)

format(today, "Dzisiaj jest %A (%d %B %y)")

weekdays(today)

as.Date(c("01.03.16"),"%d.%m.%y")
```

```
diff <- as.numeric(today - data$date)
```

19. Na potrzeby niniejszego ćwiczenia (w celu obserwacji pewnych działań) modyfikujemy nasze dane:

```
data[31,"PM25"] <- NA
```

20. Sprawdź w jaki sposób można wyświetlić konkretne wiersze:

```
head(data,5)

data[2:3,]

data[seq(1,31,by=5),]

tail(data)
```

21. Sprawdź czy któraś z wartości PM25 to NA

```
any(is.na(data$PM25))

!all(!is.na(data$PM25))
```

22. Wyświetl wszystkie dane, dla których PM25 przekracza poziom dopuszczalny (PM25>125). Zwróć uwagę, że funkcja which() sprawia, że nie wyświetlają się wiersze, dla których PM25 przyjmuje wartość NA.

```
data[(data$PM25>125),]

data[which(data$PM25>125),]
```

23. Wyświetl wszystkie dane, dla których PM10 lub PM25 przekracza poziom dopuszczalny (PM10>125, PM25>125). Zwróć uwagę na znak logiczny (| - lub, & - i).

```
data[which(data$PM10>125 | data$PM25>125),]
```

24. Przyporządkuj grupy do poszczególnych pomiarów: L (dla PM25 < kwantylu rzędu ¼), M (dla PM25 >= kwantylu rzędu ¼ oraz PM25 <= kwantylu rzędu ¾), H (dla PM25 > kwantylu rzędu ¾)

```
data[which(data$PM25 < quantile(data$PM25, na.rm=T,
probs=0.25)),"pm25group"] <- "L"

data[which(data$PM25 >= quantile(data$PM25, na.rm=T, probs=0.25) &
data$PM25 <= quantile(data$PM25, na.rm=T, probs=0.75)),"pm25group"]
<- "M"

data[which(data$PM25 > quantile(data$PM25, na.rm=T,
probs=0.75)),"pm25group"] <- "H"
```

25. Podobnie przyporządkuj grupy ze względu na wartość PM10.

```
data[which(data$PM10 < quantile(data$PM10, na.rm=T,
probs=0.25)),"pm10group"] <- "L"

data[which(data$PM10 >= quantile(data$PM10, na.rm=T, probs=0.25) &
data$PM10 <= quantile(data$PM10, na.rm=T, probs=0.75)),"pm10group"]
<- "M"
```

```
data[which(data$PM10 > quantile(data$PM10, na.rm=T,
probs=0.75)),"pm10group"] <- "H"
```

26. Wyświetl liczebność grup

```
table(data$pm25group, data$pm10group)
```

27. Posortuj dane wartości PM25 od największej do najmniejszej

```
sort(data$PM25, decreasing=TRUE)
```

28. Posortuj dataframe wg PM25 od największej do najmniejszej wartości

```
data[order(data$PM25, decreasing = T),]
```

29. Wykorzystaj bibliotekę dplry oraz operator %>% wykonaj następujące czynności:
    a. ograniczyć wartości do wartości z miesiąca luty
    b. ukryj kolumny pm25group, pm10group
    c. wyznacz dni tygodnia
    d. pogrupuj po dniu tygodnia
    e. wyświetl zestawienia ze średnią PM25 oraz PM10 dla każdego dnia tygodnia
    f. posortuj zestawienie wg PM25 od największej do najmniejszej wartości

```
library(dplyr)
```

```
data %>% filter(months(date) == "luty") %>% select(-
(pm25group:pm10group)) %>% mutate(weekday=weekdays(date)) %>%
group_by(weekday) %>% summarize(PM25 = mean(PM25), PM10 =
mean(PM10)) %>% arrange(desc(PM25))
```

30. Wykorzystując bibliotekę reshape2 zapisz pomiar każdej wartości w osobnym wierszu tabeli, a następnie zamień tabelę, gdzie w jednym wierszu będą wyświetlone pomiary wszystkich wartości z 1 dnia.

```
library(reshape2)
```

```
melt_data <- melt(data, id=c("date"), measure.vars=c("SO2", "NO2",
"NOx", "NO", "O3", "O38h", "PM10", "PM25"))
```

```
dcast(melt_data, date ~ variable)
```