МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Практикум №7

з курсу «Аналіз даних в інформаційних системах»

на тему: «Аналіз часових послідовностей»

Викладач:
Ліхоузова Т.А.

Виконав:
студент 2 курсу
групи ІП-14 ФІОТ
Шляхтун Денис

Київ-2023

Тема: Аналіз часових послідовностей.

Мета роботи: ознайомитись з методами моделювання часових послідовностей.

Основне завдання

1. Побудувати та проаналізувати часовий ряд для статистики захворювань на Covid в двох сусідніх країнах по вашому вибору (дані взяти в інтернеті).
2. Побудувати та проаналізувати часовий ряд для курсу гривня/долар або гривня/євро за останні 3 роки (дані взяти в інтернеті).

Додаткове завдання

Потрібно з'ясувати, чи є сезонна компонента в кількості опадів в Сіетлі.

1. Градуси перевести в Цельсії.
2. Чи є кореляція між температурою та опадами?
3. Скласти прогноз опадів на 2018 рік, оцінити точність прогнозу

Виконання основного завдання.

Виконання комп'ютерного практикуму здійснювалося засобами R та RStudio.

Завдання 1.

Аналіз буде здійснюватися для Франції та Німеччини.

Дані захворювання на Covid:
https://www.kaggle.com/datasets/caesarmario/our-world-in-data-covid19-dataset

Для виконання завдання було імпортовано файл owid-covid-data.csv, розглянуто його структуру та досліджено на відсутні значення. Відсутні значення нових випадків захворювання були замінені на 0, щоб не порушувати цілісність часового виміру. Дати було переведено у формат Date з chr.

```
> colSums(is.na(dataCovid))/nrow(dataCovid)*100
 location      date new_cases
 0.000000  0.000000  2.846036
> str(dataCovid)
'data.frame':   311521 obs. of  3 variables:
 $ location : chr  "Afghanistan" "Afghanistan" "Afghanistan" "Afghanistan" ...
 $ date     : chr  "2020-01-03" "2020-01-04" "2020-01-05" "2020-01-06" ...
 $ new_cases: num  0 0 0 0 0 0 0 0 0 0 ...
> summary(dataCovid)
   location             date             new_cases
 Length:311521      Length:311521      Min.    :      0
 Class :character   Class :character   1st Qu.:      0
 Mode  :character   Mode  :character   Median :     17
                                       Mean   :  10734
                                       3rd Qu.:    523
                                       Max.   :7460822
                                       NA's   :8866
```

Рис. 1 – аналіз даних до виправлення

```
> colSums(is.na(dataCovid))/nrow(dataCovid)*100
 location      date new_cases
        0         0         0
> str(dataCovid)
'data.frame':   311521 obs. of  3 variables:
 $ location : chr  "Afghanistan" "Afghanistan" "Afghanistan" "Afghanistan" ...
 $ date     : Date, format: "2020-01-03" "2020-01-04" "2020-01-05" "2020-01-06" ...
 $ new_cases: num  0 0 0 0 0 0 0 0 0 0 ...
> summary(dataCovid)
   location             date             new_cases
 Length:311521      Min.   :2020-01-01   Min.    :      0
 Class :character   1st Qu.:2020-11-08   1st Qu.:      0
 Mode  :character   Median :2021-09-11   Median :     13
                    Mean   :2021-09-10   Mean   :  10428
                    3rd Qu.:2022-07-15   3rd Qu.:    477
                    Max.   :2023-05-23   Max.   :7460822
```

Рис. 2 – аналіз даних після виправлення

Далі було виокремлено окремі датафрейми для Франції та Німеччини.

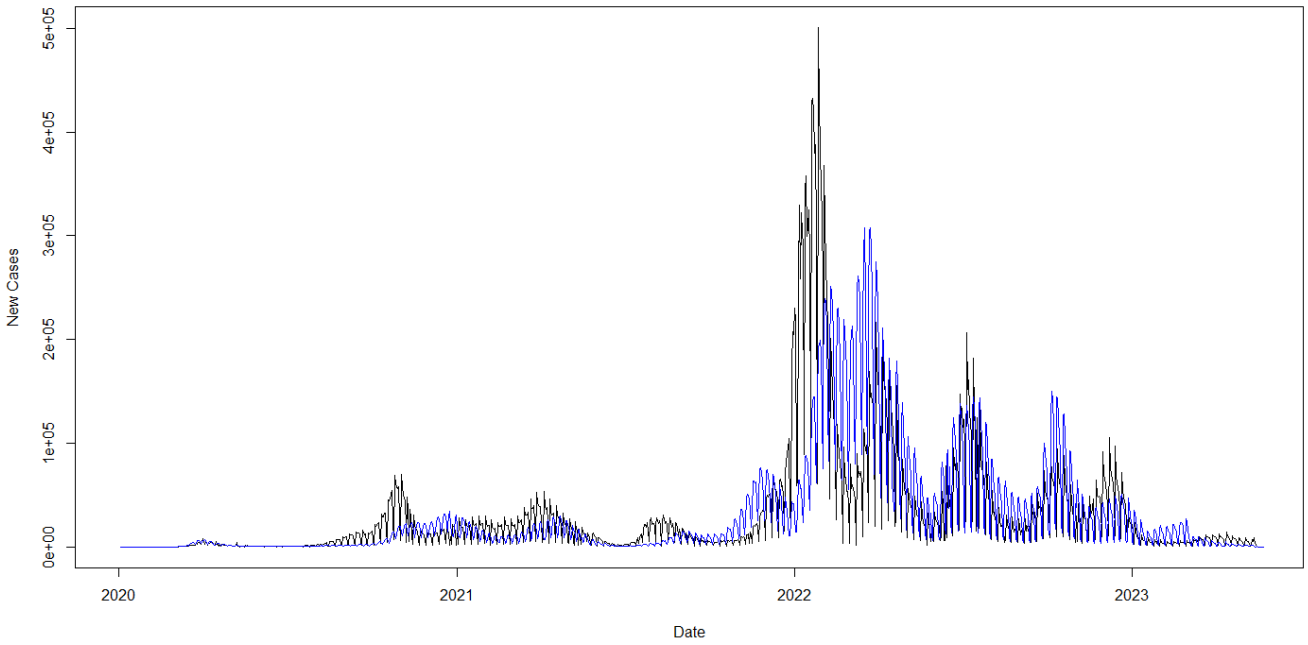

Рис. 3 – нові випадки Covid для Франції (чорний) та Німеччини (синій)

Далі було створено часові ряди для Франції і Німеччини за допомогою функції ts().

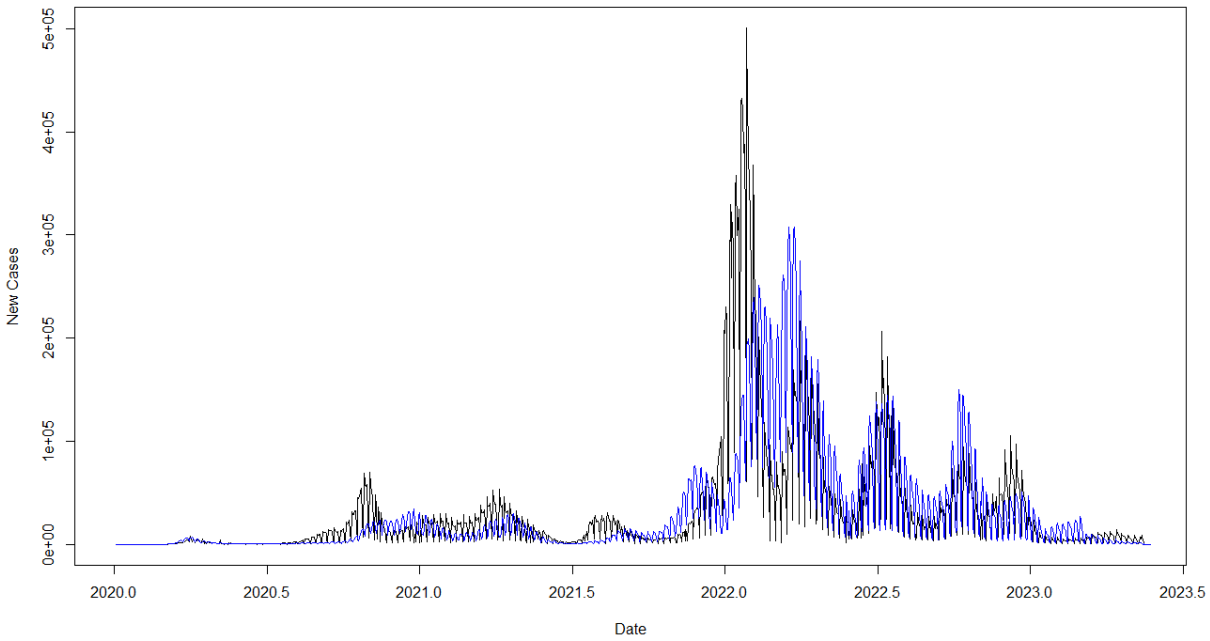

Рис. 4 – часові ряди для Франції та Німеччини

Було здійснено декомпозицію на тренд, сезонну та випадкову складові за допомогою функції decompose().

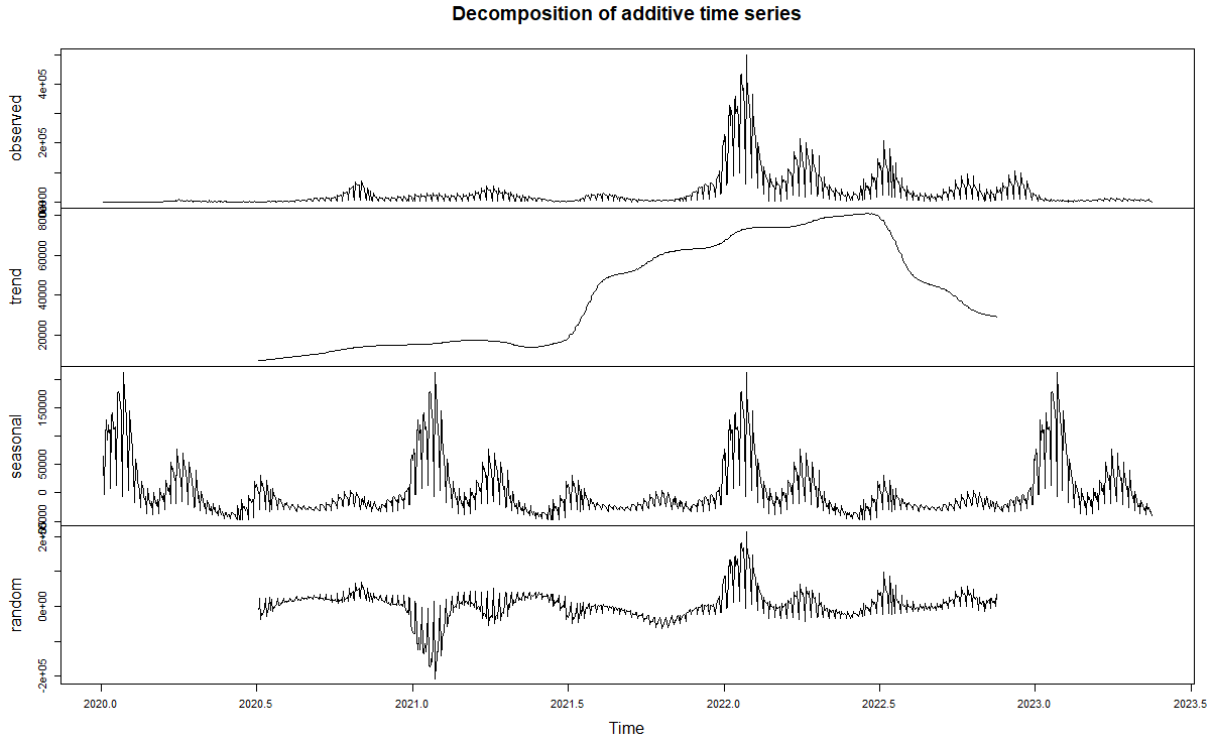

Рис. 5 – декомпозиція числового ряду Франції

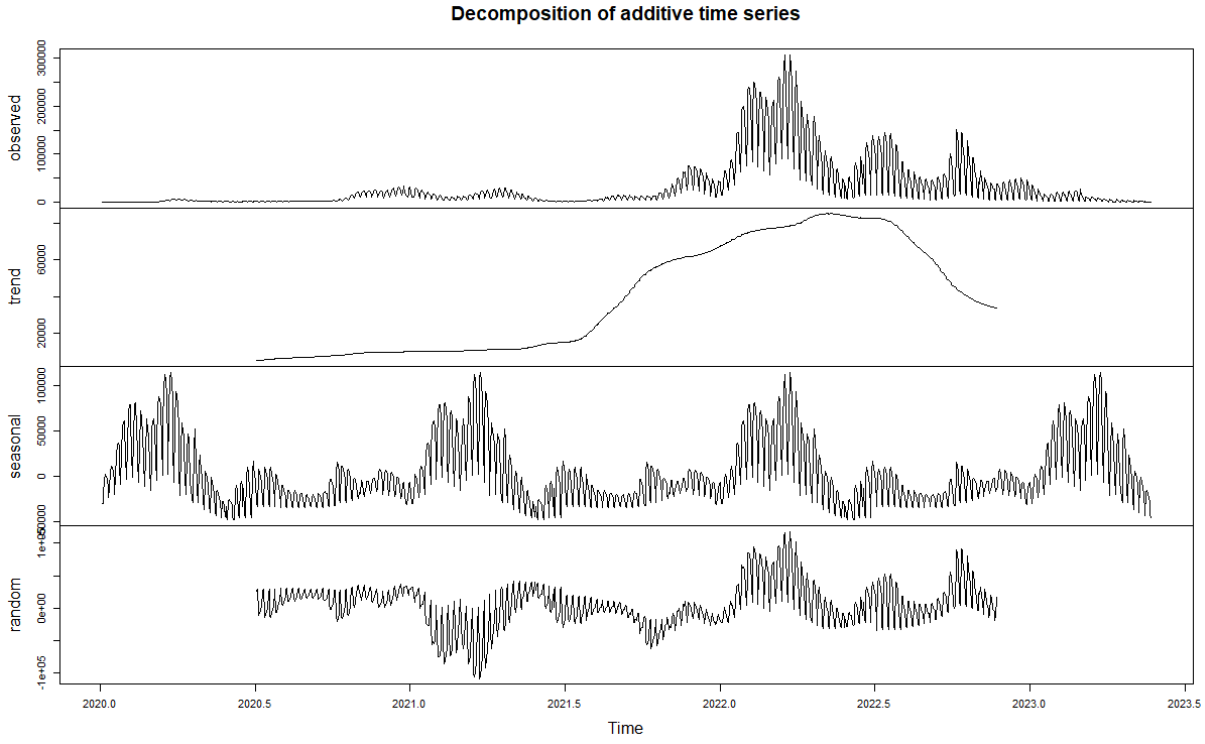

Рис. 6 – декомпозиція числового ряду Німеччини

Для прогнозування використовувалася модель ARIMA. Параметри підбиралися шляхом перебору з мінімізацією інформаційного критерію Акаіке.

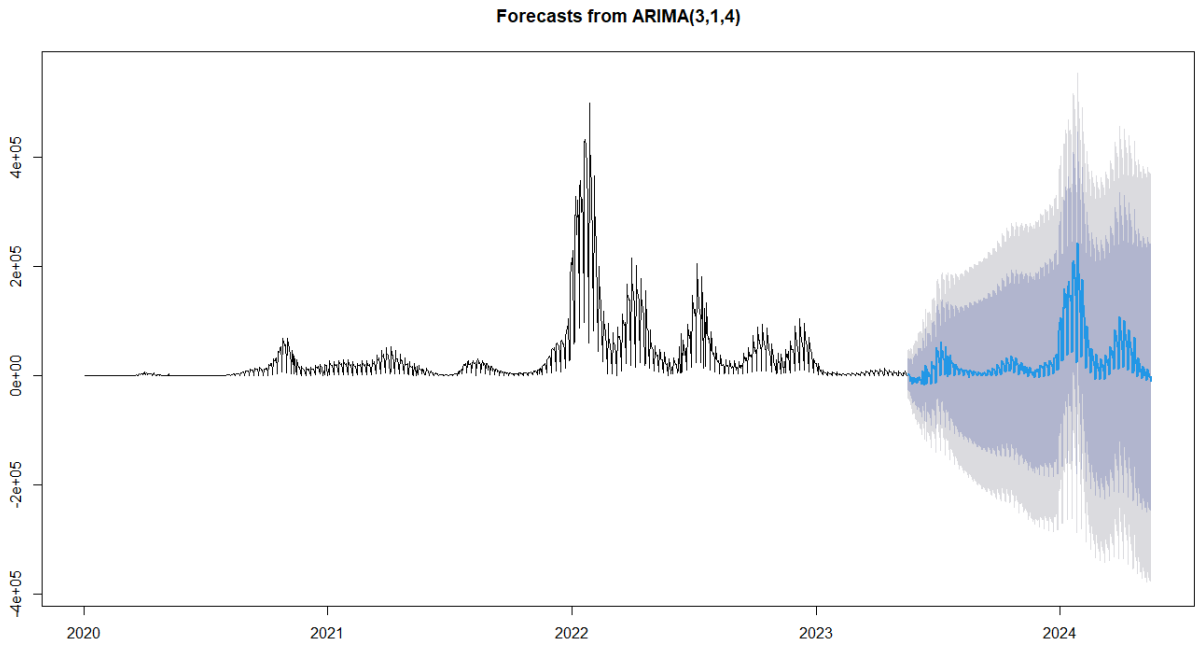

Рис. 7 – прогноз для Франції

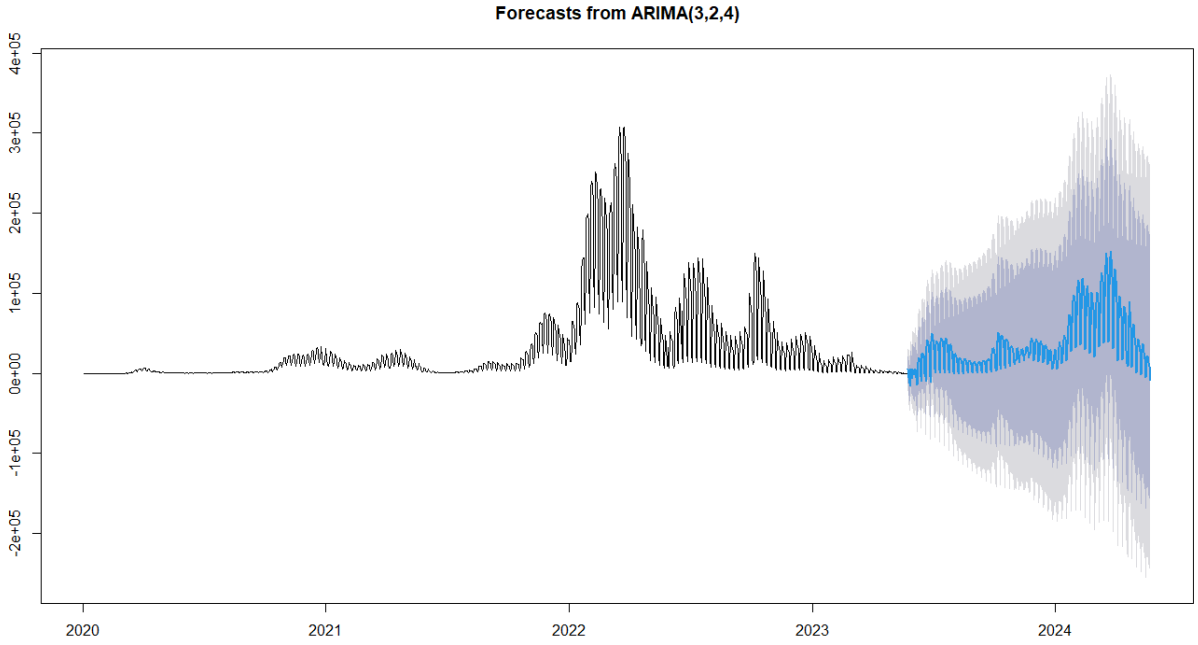

Рис. 8 – прогноз для Німеччини

Завдання 2.

Датасет взято з https://bank.gov.ua/ua/markets/exchangerate-chart.

Виправлення датасету було не потрібним, було лише змінено формат колонки дати з chr на Date. Маніпуляції з даними проводилися аналогічні з попереднім завданням, далі надаються лише результати роботи.

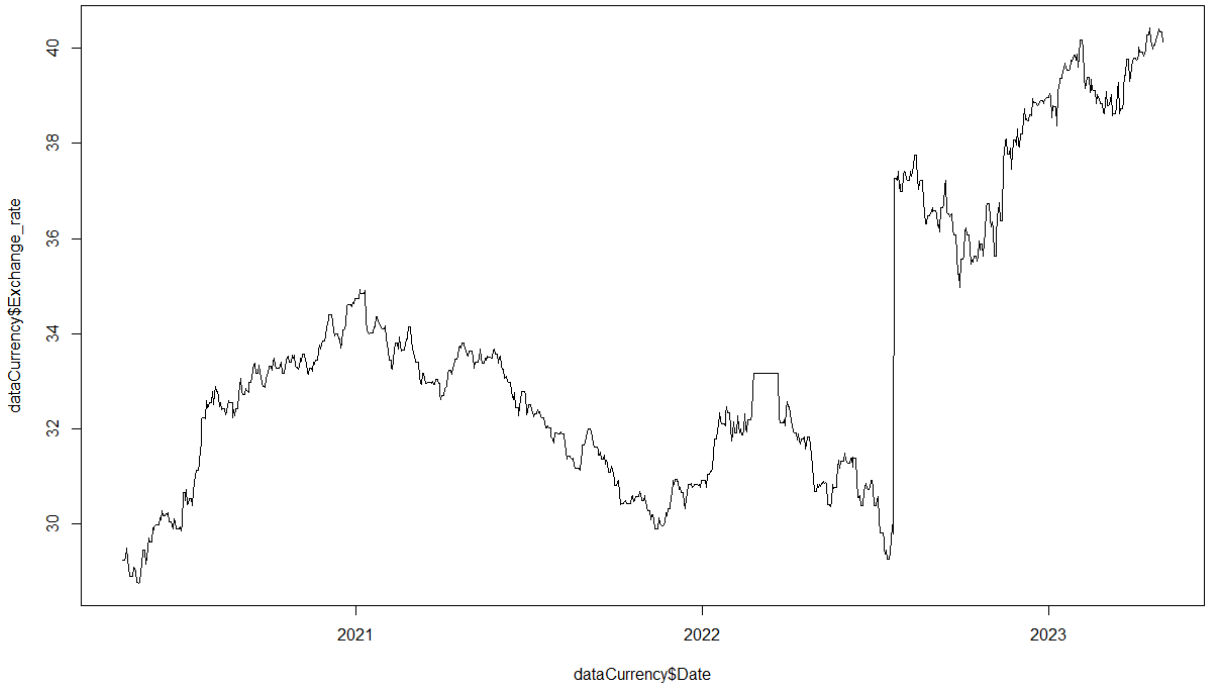

Рис. 9 – курс гривні до євро

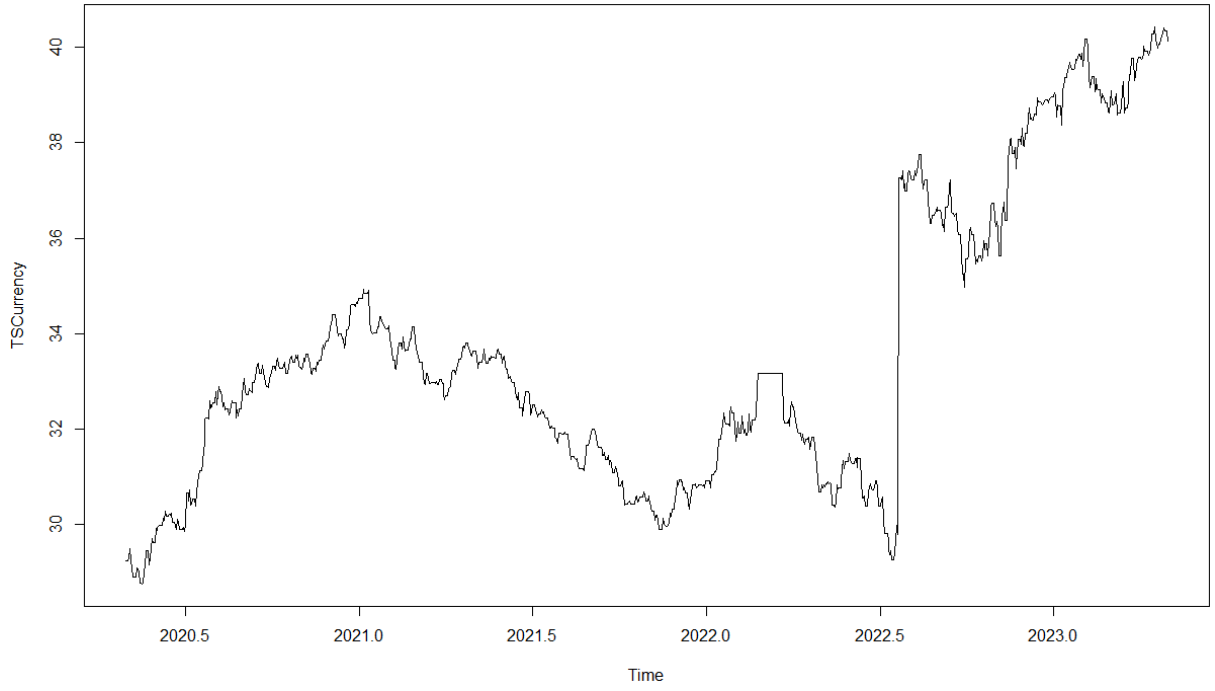

Рис. 10 – створення часового ряду

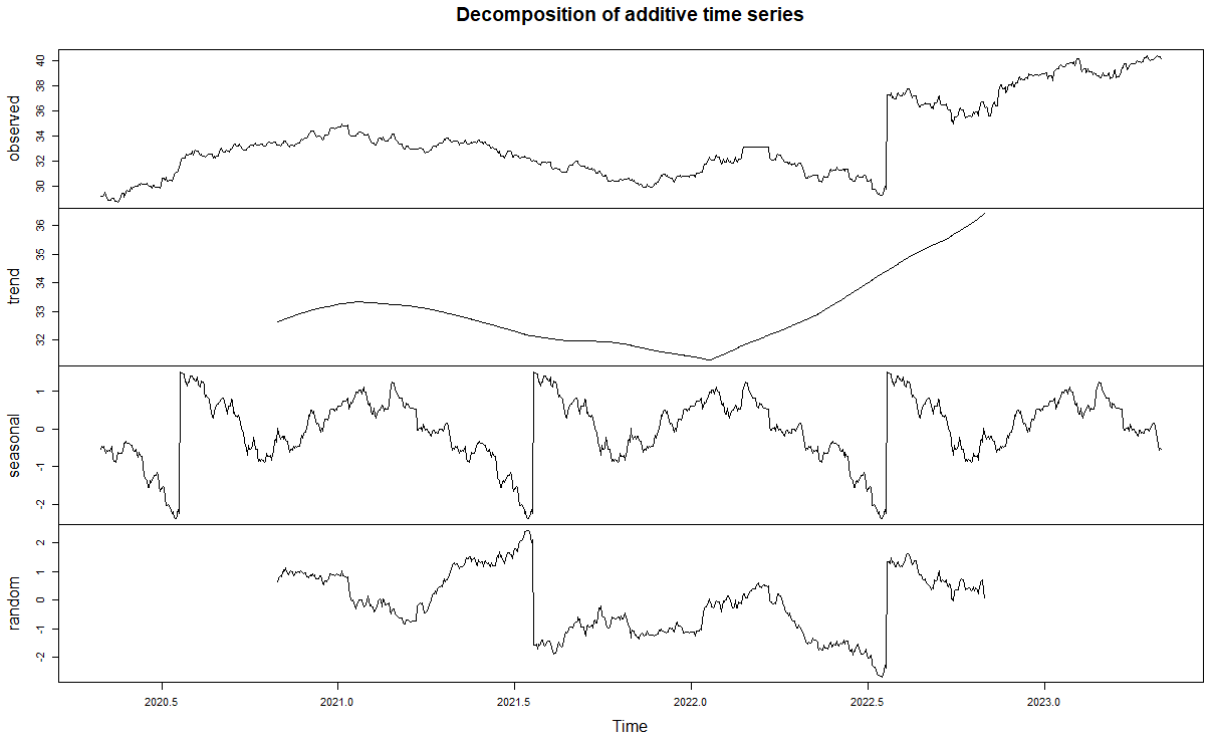

Рис. 11 – декомпозиція часового ряду

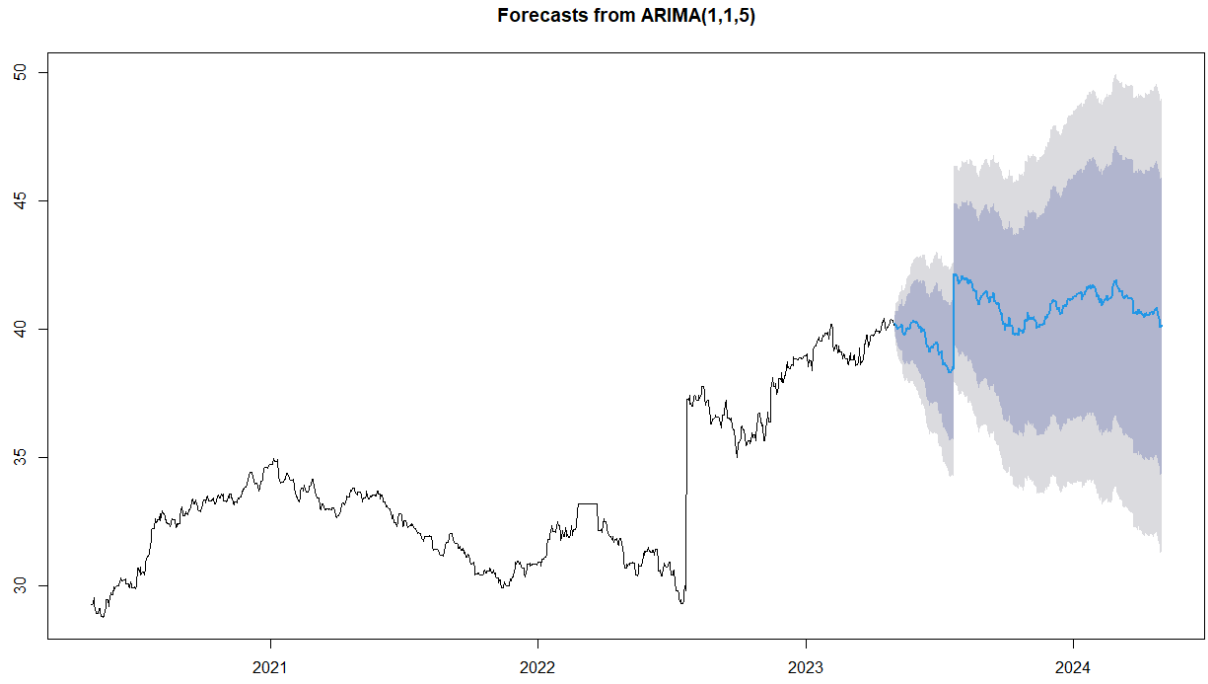

Рис. 12 – прогноз курсу

Виконання додаткового завдання.

Для виконання завдання було імпортовано файл seattleWeather_1948-2017.csv. Для виправлення було усі 3 відсутні значення RAIN <- FALSE і PRCP <- 0, адже в середньому недощових днів було більше і щоб не

порушувати цілісність часового виміру. Також було переведено дату у формат Date.

1. Градуси перевести в Цельсії.

```
# 1. Градуси перевести в Цельсії
dataweather$TMAX <- (dataweather$TMAX - 32) * 5 / 9
dataweather$TMIN <- (dataweather$TMIN - 32) * 5 / 9
```

Рис. 13 – переведення градусів у Цельсії

2. Чи є кореляція між температурою та опадами?

```
> cor(dataweather[2:4], use = "complete")
             PRCP       TMAX        TMIN
PRCP  1.00000000 -0.2267358 -0.06436549
TMAX -0.22673585  1.0000000  0.86067507
TMIN -0.06436549  0.8606751  1.00000000
```

Рис. 14 – перевірка кореляції між температурою та опадами

Зв’язок обернений, слабкий.

3. Скласти прогноз опадів на 2018 рік, оцінити точність прогнозу

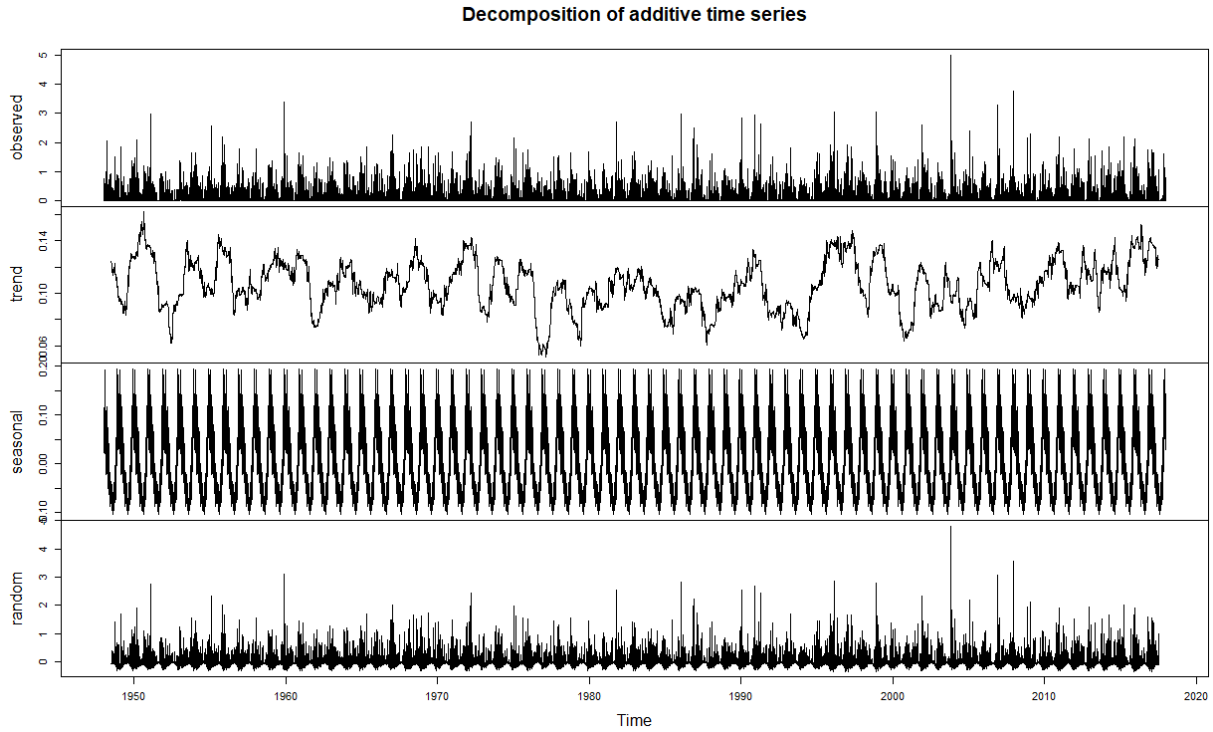

Рис. 15 – декомпозиція часового ряду

Тож, сезонна компонента є.

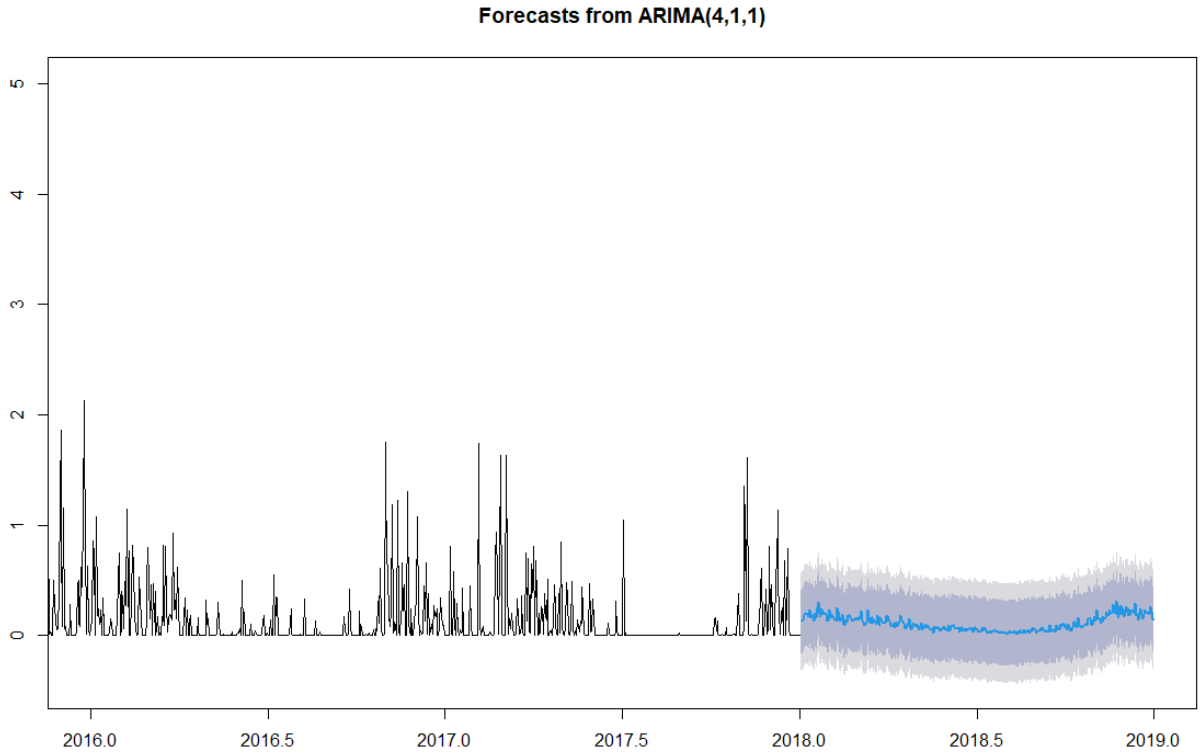

Рис. 16 – прогноз опадів

Висновок.

При виконанні лабораторної роботи було ознайомлено з методами моделювання часових послідовностей та прогнозування за допомогою моделі ARIMA. При виконанні роботи було проаналізовані різні часові послідовності і було зроблено прогнозування для них.

Додаток А. Код мовою програмування R

```
library(dplyr)
library(forecast)
library(plm)

# завдання 1 - часовий ряд для статистики захворювань на Covid в двох сусідніх країнах

# завантаження даних
dataCovid <- read.csv("owid-covid-data.csv", sep=",", header = TRUE, dec = '.')

# вибір колонок лише про країну, дату і зареєстровано випадків за день
dataCovid <- select(dataCovid, location, date, new_cases)

# дослідження даних
colSums(is.na(dataCovid))/nrow(dataCovid)*100
str(dataCovid)
summary(dataCovid)

# виправлення даних, перетворення колонки у формат дати
dataCovid$new_cases[is.na(dataCovid$new_cases)] <- 0
dataCovid$date <- as.Date(dataCovid$date)


# виокремлення даних для Франції і Німеччини

dataCovidFrance <- dataCovid[dataCovid$location == "France",]
dataCovidFrance$location <- NULL

dataCovidGermany <- dataCovid[dataCovid$location == "Germany",]
dataCovidGermany$location <- NULL

plot(dataCovidFrance$date,dataCovidFrance$new_cases, type='l', xlab = "Date", ylab = "New Cases")
lines(dataCovidGermany$date,dataCovidGermany$new_cases, type='l', col="blue")
```

```
# створення часових рядів

TSFrance <- ts(dataCovidFrance$new_cases, frequency = 365, start = c(2020,3))
str(TSFrance)

TSGermany <- ts(dataCovidGermany$new_cases, frequency = 365, start = c(2020,3))
str(TSGermany)

plot(TSFrance, xlab = "Date", ylab = "New Cases")
lines(TSGermany, col = "blue")



# декомпозиція на тренд (T), сезону (S) та випадкову (R) складові

decFrance <- decompose(TSFrance)
plot(decFrance)

decGermany <- decompose(TSGermany)
plot(decGermany)



# визначення автокореляції, при значенні > 0.8 вона є
acf(TSFrance, lag.max = 20)
acf(TSFrance-decFrance$seasonal, lag.max = 20)
acf(TSGermany, lag.max = 200)
acf(TSGermany-decGermany$seasonal, lag.max = 20)

# p < 0.05, нульова гіпотеза про відсутність автокореляції відхиляється
Box.test(TSFrance, lag=20, type="Ljung-Box")
Box.test(TSGermany, lag=20, type="Ljung-Box")

# Зображення приростів
plot(diff(TSFrance))
```

```
hist(diff(TSFrance), breaks = 30)

plot(diff(TSGermany))

hist(diff(TSGermany), breaks = 30)


# Прогноз, модель ARIMA

# order= c(p,q,d) - p - скільки попередніх значень враховувати для прогнозу (по автокореляції, або підбором від 1)

#                 q - скільки попередніх значень прогнозу враховувати для прогнозу (підбором від 1)

#                 d - скільки похідних враховувати для прогнозу (до 5)

#                     підбір спочатку d, потім p, потім q

# AIC(model) інформаційний критерій Акаіке для моделі (чим менше, тим краще)


#France


# перед тим, як моделювати, треба прибрати сезонну компоненту і не забути її потім додати до прогнозу

TSFranceSeasonal <- TSFrance-decFrance$seasonal


for (i in 0:5) # вибрали 4

{

  print(i)

  print(AIC(arima(TSFranceSeasonal, order = c(0,0,i))))

}


for (i in 0:5) # вибрали 3

{

  print(i)

  print(AIC(arima(TSFranceSeasonal, order = c(i,0,4))))

}


for (i in 1:4) # вибрали 1

{

  print(i)

  print(AIC(arima(TSFranceSeasonal, order = c(3,i,4))))

}
```

```
forecFrance<-forecast(arima(TSFranceSeasonal, order = c(3,1,4)), h=365)
plot(forecFrance)

# останнє значення сезонної компоненти в даних, щоб додати сезонну компоненту до прогнозу
last<-decFrance$seasonal[length(decFrance$seasonal)]

# знайти позицію останнього значення
which(decFrance$figure==last)

# починаючи з наступного значення, додати повторюваний фрагмент до прогнозу
season<-c(decFrance$figure[137:365],decFrance$figure[1:136])
forecFrance$mean<-forecFrance$mean + season
forecFrance$x<-forecFrance$x + decFrance$seasonal
forecFrance$lower<-forecFrance$lower+season
forecFrance$upper<-forecFrance$upper+season

# намалювати прогноз
plot(forecFrance)
plot(forecFrance, xlim=c(2023, 2025))

# Germany

# перед тим, як моделювати, треба прибрати сезонну компоненту і не забути її потім додати до прогнозу
TSGermanySeasonal <- TSGermany-decGermany$seasonal

for (i in 0:5) # вибрали 3
{
  print(i)
  print(AIC(arima(TSGermanySeasonal, order = c(0,0,i))))
}

for (i in 0:5) # вибрали 5
{
  print(i)
```

```
  print(AIC(arima(TSGermanySeasonal, order = c(i,0,3))))
}

for (i in 1:4) # вибрали 2
{
  print(i)
  print(AIC(arima(TSGermanySeasonal, order = c(5,i,3))))
}

forecGermany<-forecast(arima(TSGermanySeasonal, order = c(3,2,4)), h=365)
plot(forecGermany)

# останнє значення сезонної компоненти в даних, щоб додати сезонну компоненту до прогнозу
last<-decGermany$seasonal[length(decGermany$seasonal)]

# знайти позицію останнього значення
which(decGermany$figure==last)

# починаючи з наступного значення, додати повторюваний фрагмент до прогнозу
season<-c(decGermany$figure[143:365],decGermany$figure[1:142])
forecGermany$mean<-forecGermany$mean + season
forecGermany$x<-forecGermany$x + decGermany$seasonal
forecGermany$lower<-forecGermany$lower+season
forecGermany$upper<-forecGermany$upper+season

# намалювати прогноз
plot(forecGermany)
plot(forecGermany, xlim=c(2023, 2025))



# завдання 2 - часовий ряд для курсу гривня/долар або гривня/євро за останні 3 роки

# завантаження даних
```

```
dataCurrency <- read.csv("uah_to_euro.csv", sep=",", header = TRUE, dec = '.')

# вибір колонок лише про дату і курс
dataCurrency <- select(dataCurrency, Дата, Офіційний.курс.гривні..грн)
names(dataCurrency)[names(dataCurrency) == "Дата"] <- "Date"
names(dataCurrency)[names(dataCurrency) == "Офіційний.курс.гривні..грн"] <- "Exchange_rate"

# дослідження даних
colSums(is.na(dataCurrency))/nrow(dataCurrency)*100
str(dataCurrency)
summary(dataCurrency)

# перетворення колонки у формат дати
dataCurrency$Date <- as.Date(dataCurrency$Date, format = "%d.%m.%Y")

# виведення даних
plot(dataCurrency$Date,dataCurrency$Exchange_rate, type='l')

# створення часового ряду
TSCurrency <- ts(dataCurrency$Exchange_rate, frequency = 365, start = c(2020,121))
str(TSCurrency)
plot(TSCurrency)

# декомпозиція на тренд (T), сезону (S) та випадкову (R) складові
decCurrency <- decompose(TSCurrency)
plot(decCurrency)

# визначення автокореляції, при значенні > 0.8 вона є
acf(TSCurrency, lag.max = 20)
acf(TSCurrency-decCurrency$seasonal, lag.max = 20)

# p < 0.05, нульова гіпотеза про відсутність автокореляції відхиляється
Box.test(TSCurrency, lag=20, type="Ljung-Box")
```

```
# Зображення приростів

plot(diff(TSCurrency))

hist(diff(TSCurrency), breaks = 30)  # залишки моделі мають бути розподілені нормально - як перевірити та подивитись


# перед тим, як моделювати, треба прибрати сезонну компоненту і не забути її потім додати до прогнозу

TSCurrencySeasonal <- TSCurrency-decCurrency$seasonal


for (i in 0:5) # вибрали 5

{

 print(i)

 print(AIC(arima(TSCurrencySeasonal, order = c(0,0,i))))

}


for (i in 0:2) # вибрали 1

{

 print(i)

 print(AIC(arima(TSCurrencySeasonal, order = c(i,0,5))))

}


for (i in 0:4) # вибрали 1

{

 print(i)

 print(AIC(arima(TSCurrencySeasonal, order = c(1,i,5))))

}


forecCurrency<-forecast(arima(TSCurrencySeasonal, order = c(1,1,5)), h=365)


# щоб додати сезонну компоненту до прогнозу треба подивитись, яке було останнє значення сезонної компоненти в наявних даних

last<-decCurrency$seasonal[length(decCurrency$seasonal)]


# знайти це значення в повторюваному фрагменті сезонної компоненти

which(decCurrency$figure==last)
```

```
# починаючи з наступного значення, додати повторюваний фрагмент до прогнозу
season<-c(decCurrency$figure[2:365],decCurrency$figure[1])
forecCurrency$mean<-forecCurrency$mean + season
forecCurrency$x<-forecCurrency$x + decCurrency$seasonal
forecCurrency$lower<-forecCurrency$lower+season
forecCurrency$upper<-forecCurrency$upper+season

# намалювати прогноз
plot(forecCurrency)
plot(forecCurrency, xlim=c(2023, 2025))



# додаткове завдання

# завантаження даних
dataWeather <- read.csv("seattleWeather_1948-2017.csv", sep=",", header = TRUE, dec = '.')
str(dataWeather)
summary(dataWeather)

# заповнення NA
dataWeather[is.na(dataWeather$RAIN),]
dataWeather$RAIN[is.na(dataWeather$RAIN)] <- FALSE
dataWeather$PRCP[is.na(dataWeather$PRCP)] <- 0

# переведення дати
dataWeather$DATE <- as.Date(dataWeather$DATE, format = "%Y-%m-%d")

# 1. Градуси перевести в Цельсії
dataWeather$TMAX <- (dataWeather$TMAX - 32) * 5 / 9
dataWeather$TMIN <- (dataWeather$TMIN - 32) * 5 / 9

# 2. Чи є кореляція між температурою та опадами
cor(dataWeather[2:4], use = "complete") # зв'язок слабкий обернений
```

```
# 3. Скласти прогноз опадів на 2018 рік. Оцінити точність прогнозу

# Чи є сезонна компонента в кількості опадів в Сіетлі?
TSWeather<-ts(dataWeather$PRCP, frequency = 365, start = c(1948,1))
plot(TSWeather)
decWeather <- decompose(TSWeather) # є сезонна компонента
plot(decWeather)

# Скласти прогноз опадів на 2018 рік
acf(TSWeather-decWeather$seasonal, lag.max = 21)  # оцінка автокореляції, p=4

# перед тим, як моделювати, треба прибрати сезонну компоненту і не забути її потім додати до прогнозу
TSWeatherSeasonal <- TSWeather-decWeather$seasonal

for (i in 0:5) # вибрали 1
{
  print(i)
  print(AIC(arima(TSWeatherSeasonal, order = c(4,0,i))))
}

for (i in 1:3) # вибрали 1
{
  print(i)
  print(AIC(arima(TSWeatherSeasonal, order = c(4,i,1))))
}

forecWeather<-forecast(arima(TSWeatherSeasonal, order = c(4,1,1)), h=365) # order= c(p=4,q=1,d=1)
forecWeather$model$aic

# останнє значення сезонної компоненти в даних, щоб додати сезонну компоненту до прогнозу
last<-decWeather$seasonal[length(decWeather$seasonal)]

# знайти позицію останнього значення
```

```
which(decWeather$figure==last)


# починаючи з наступного значення, додати повторюваний фрагмент до прогнозу
season<-c(decWeather$figure[2:365],decWeather$figure[1])
forecWeather$mean<-forecWeather$mean + season
forecWeather$x<-forecWeather$x + decWeather$seasonal
forecWeather$lower<-forecWeather$lower+season
forecWeather$upper<-forecWeather$upper+season


# намалювати прогноз
plot(forecWeather)
plot(forecWeather, xlim=c(2016,2019))


# оцінка точності прогнозу неможлива за допомогою даного датасету - відсутні фактичні значення для порівняння
```