Oppgaver 2 - løsning

October 6, 2025

1 Oppgaver 2

1.1 Hent inn pakken `tidyverse` som skal brukes i oppgavene med funksjonen `library()`

[ ]:
```
library(tidyverse)
```

1.2 Lag et objekt `teller` som inneholder den numeriske verdien 10 og et objekt `nevner` som inneholder den numeriske verdien 15. Bruk disse objektene til beregne andelen i prosent og lagre resultatet i objektet `andel`.

[ ]:
```
teller <- 10
nevner <- 15

andel <- teller/nevner*100
andel
```

1.3 Bruk objektet `andel` og rund av til én desimal med funksjonen `round()`.

[ ]:
```
round(andel, digits = 1)
```

1.4 Bruk objektet `andel` og rund av til null desimaler med funksjonen `round()` og gjør deretter om til et heltall med funksjonen `as.integer()`. Se om resultatet blir ulikt om du kun endrer variabeltypen til heltall uten å avrunde først.

[ ]:
```
as.integer(round(andel, digits = 0))
as.integer(andel)
```

1.5 Lag et objekt med navnet ditt (karakter) og et objekt med alderen din (numerisk) og skriv det ut i konsollen. Kall objektene for `navn` og `alder`

[ ]:
```
navn <- "Sindre"
navn

alder <- 31
```

```
alder
```

1.6 Bruk funksjonen `class()` på objektene `navn` og `alder`. Hva forteller denne?

[ ]:
```
class(navn)
class(alder)
```

1.7 Hva skjer hvis du ikke har fnutter ("") omkring navnet ditt når du definerer objektet `navn`?

[ ]:
```

```

1.8 Bruk funksjonen `nchar()` for å se hvor mange bokstaver navnet ditt innholder

[ ]:
```
nchar(navn)
```

1.9 Bruk funksjonen `substr()` for å hente ut den andre bokstaven i navnet ditt.

[ ]:
```
substr(navn, 2, 2)
```

1.10 Bruk funksjonen `substr()` for å hente ut de to første bokstavene i navnet ditt.

[ ]:
```
substr(navn, 1, 2)
```

1.11 Bruk funksjonen `substr()` for å hente ut det første sifferet i `alder` og lagre resultatet i et nytt objekt som heter `alder_1`.

[ ]:
```
alder_1 <- substr(alder, 1, 1)
alder_1
```

1.12 Lag en kode som øker verdien til objektet `alder_1` med 1 hver gang koden kjøres. Hint: sjekk variabeltypen til `alder_1`.

[ ]:
```
alder_1 <- as.numeric(alder_1)+1
alder_1
```

1.13 Bruk funksjonen `paste()` til å lime sammen objektene `navn` og `alder` til en setning (f.eks. "NAVN er ALDER år gammel"). Prøv deretter å endre fra funksjonen `paste()` til `paste0()` og se hvordan setningen endrer seg. Juster koden slik at den blir lik som den første. Lag til slutt den samme setningen med funksjonen `glue::glue()`

[ ]:
```
paste(navn, "er", alder, "år gammel")
paste0(navn, "er", alder, "år gammel")

paste0(navn, " er ", alder, " år gammel")
```

1.14 Opprett følgende objekter:

- Sett `navn` til en kort tekst som forklarer innholdet i et datasett.
- Sett `aargang` til et valgfritt år.
- Sett `versjon` til et helt tall.

Ifølge den nye navnestandarden på Dapla, skal en fil være på følgende form:

`flygende_objekter_p2019_v1.parquet`

Sett sammen `navn`, `aargang` og `versjon` og sett resultatet til `filnavn`, slik at filnavnet stemmer med navnestandarden. Print deretter innholdet i `filnavn` i konsollen for å kontrollere at det har blitt riktig satt sammen.

[ ]:
```
navn <- "flygende_objekter"
aargang <- 2019
versjon <- 2

filnavn <- paste0(navn, "_p", aargang, "_v", versjon, ".parquet")

print(filnavn)
```

1.15 Lag objektet `aargang` og erstatt alle årstall i denne filstien med verdien fra `aargang`:

`/data/prosjekt/2023/årsrapport_2023/backup_2023_rapport.csv`

Lagre filstien i objektet `filsti`. Sjekk at filstien blir oppdatert riktig når du endrer årstallet i `aargang` og kjører koden på nytt.

[ ]:
```
aargang <- 2023

# filsti <- paste0("/data/prosjekt/", aargang, "/årsrapport_", aargang, "/
  ↪backup_", aargang, "_rapport.csv")
filsti <- glue::glue("/data/prosjekt/{aargang}/årsrapport_{aargang}/
  ↪backup_{aargang}_rapport.csv")
filsti
```

1.16 Bruk funksjonen `gsub()` til å erstatte endelsen ".csv" med ".parquet" i objektet `filsti`

```
[ ]: gsub(".csv", ".parquet", filsti)
```

1.17 Norske kommunenummer er firesifrede på formen FFKK, der FF er fylkesnummeret og KK er et løpenummer innenfor fylket.

Lag et objekt `komm_nr` med et valgfritt kommunenummer og bruk R til å hente ut det korresponderende fylkesnummeret `fylke_nr` til kommunen. Print ut en passende tekst som inneholder `komm_nr` og `fylke_nr`. Oppdater `komm_nr` med ulike kommunenumre og sjekk at koden din stemmer.

```
[ ]: komm_nr <- "0301"
fylke_nr <- substr(komm_nr, 1, 2)

paste0("Kommunen ", komm_nr, " ligger i fylke ", fylke_nr)
```

1.18 Bruk funksjonen `stringr::str_pad()` til å omgjøre `komm_nr` og `fylke_nr` til å bestå av 8 tegn. Fyll inn med 0.

```
[ ]: stringr::str_pad(komm_nr, width = 8, "right", pad = "0")
stringr::str_pad(fylke_nr, width = 8, "right", pad = "0")
```

1.19 Bruk funksjonen `stringr::str_extract()` for å hente kommunenummeret fra denne tekststrengen: "Oslo har kommunenummer 0301"

```
[ ]: stringr::str_extract("Oslo har kommunenummer 0301", "[0-9]+")
```

1.20 Gjør det samme som i forrige oppgave, men lag en test om strengen inneholder tall (TRUE/FALSE)

```
[ ]: stringr::str_detect("Oslo har kommunenummer 0301", "[0-9]+")
```

1.21 Lag en karaktervektor som heter `handleliste` som inneholder de fem elementene: Banan, Eple, Melk, Brød og Tannpasta

```
[ ]: handleliste <- c("Banan", "Eple", "Melk", "Brød", "Tannpasta")
```

1.22 Lag en test for å sjekke hvor mange elementer som finnes i `handleliste`

```
[ ]: length(handleliste)
```

1.23 Hent ut det tredje elementet i `handleliste`.

```
[ ]: handleliste[3]
```

1.24 Lag en test for å sjekke om "Brød" finnes i `handleliste`.

```
[ ]: "Brød" %in% handleliste
```

1.25 Omgjør `handleliste` til en data frame og hent ut den andre raden.

```
[ ]: data.frame(handleliste)[2,]
```

1.26 Lag en heltallsvariabel som heter `aar` og sett den til et fødselsår. Lag en if-setning som sjekker om fødselsåret er før 1990. Lag en passende tekst som skrives til skjerm med `print()` dersom kritteriet er oppfylt. Test at logikken stemmer med ulike verdier for `aar`.

```
[ ]: aar <- 1993

if (aar < 1990){
  print("Fødselsåret er før 1990")
}
```

1.27 Kopier if-setningen din i cellen under og utvid den med en `else` (dersom betingelsen ikke er oppfylt). Skriv en passende melding til skjerm.

```
[ ]: if (aar < 1990){
  print("Fødselsåret er før 1990")
} else {
  print("Fødselsåret er IKKE før 1990")
}
```

1.28 Gjør det samme som i oppgaven over, men endre sjekken for om fødselsåret er på 1990-tallet.

```
[ ]: if (aar %in% 1990:1999){
  print("Fødselsåret er på 1990-tallet")
} else {
  print("Fødselsåret er IKKE på 1990-tallet")
}
```

1.29 Lag en ny variabel, `alder`, som er alderen til en person født i `aar` (se bort i fra fødselsdato). Bruk `alder` i en if-setning for å sjekke om en person med dette fødselsåret er myndig eller ikke. Oppdater `aar` med ulike verdier og se at logikken stemmer.

[ ]:
```
aar <- 2007

alder <- 2024-aar

if (alder >= 18){
  print("Myndig")
} else {
  print("Ikke myndig")
}
```

1.30 Lag et objekt med et årstall som heter `aargang` og bruk funksjonen `paste0()` til å lage en filsti til en mappe der du har filer med flere årganger. Dersom du ikke har et eget egnet eksempel lager du en fiktiv filsti. Endre årstallet og se at filstien oppdaterer seg riktig. Prøv gjerne også å lese inn filen som filstien peker til.

[ ]:
```
aargang <- 2024
filsti <- paste0("C:/Data/Prosjekt/Aargang_", aargang, "/resultater.csv")
filsti
```

1.31 Lag din egen data frame ved å bruke det du har lært om vektorer. Lag deg en miniversjon med fiktive data av et datasett du jobber med. Pass på at det er like mange elementer i hver vektor/kolonne.

[ ]:
```
# Eksempel:

# data <- data.frame(aargang = 2024,
#                    komm_nr = c('0301', '0301', '0301', '5001', '4601'),
#                    sykehus = c('Ullevål', 'Rikshospitalet', 'Aker', 'St.␣
 ↪Olavs', 'Haukeland'),
#                    plass = c(120, 150, 89, 41, 64))

# summary(data)
```