~~~~
---
title: "Week-3: Code-along"
author: "Misra Anandita"
date: "`r Sys.Date()`"
output:
  html_document: null
  df_print: paged
  pdf_document: default
  word_document: default
pdf_document: default
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```


# I. Code to edit and execute

**To be submitted on canvas before attending the tutorial**


### Loading packages

```{r, eval=TRUE,echo=TRUE}
# Load package tidyverse
library("tidyverse")



```


### Assigning values to variables

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Example a.: execute this example
x <- 'A'
x
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Complete the code for Example b and execute it
x <- "Apple"
x
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Complete the code for Example c and execute it
x <- FALSE
x
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Complete the code for Example d and execute it
x <- 5L
x
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Complete the code for Example e and execute it
x <- 5
x

```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Complete the code for Example f and execute it
x <- 1i
~~~~

~~~
x
```

### Checking the type of variables

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Example a.: execute this example
x <- 'A'
typeof(x)
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Complete the code for Example b and execute it
x <- "Apple"
typeof(x)
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Complete the code for Example c and execute it
x <- FALSE
typeof(x)
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Complete the code for Example d and execute it
x <- 5L
typeof(x)
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Complete the code for Example e and execute it
x <- 5
typeof(x)
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Complete the code for Example f and execute it
x <- 1i
typeof(x)
```

### Need for data types
```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# import the cat-lovers data from the csv file you downloaded from canvas
cat_lovers <- read_csv("cat-lovers.csv")
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Compute the mean of the number of cats: execute this command
mean(cat_lovers$number_of_cats)
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Get more information about the mean() command using ? operator
?mean
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Convert the variable number_of_cats using as.integer()
as.integer(cat_lovers$number_of_cats)
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Display the elements of the column number_of_cats
cat_lovers$number_of_cats
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
~~~

~~~
# Display the elements of the column number_of_cats after converting it using
as.numeric()
as.numeric(cat_lovers$number_of_cats)
cat_lovers$number_of_cats
```

### Create an empty vector

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Empty vector
x <- vector()
# Type of the empty vector
typeof(x)
```

### Create vectors of type logical

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 1
x<-vector("logical",length=5)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 2
x<-logical(5)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 3
x<-c(TRUE,FALSE,TRUE,FALSE,TRUE)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

### Create vectors of type character

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 1
x<-vector("character",length=5)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 2
x<-character(5)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 3
x<-c('A','b','r','q')
# Display the contents of x
~~~

~~~
print(x)
# Display the type of x
print(typeof(x))
```

### Create vectors of type integer

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 1
x<-vector("integer",length=5)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 2
x<-integer(5)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))

```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 3
x<-c(1L,2L,3L,4L,5L)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 4
x<-seq(from=1,to=5,by=0.1)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 5
x<-1:5
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

### Create vectors of type double

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 1
x<-vector("double", length=5)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 2
x<-double(5)
# Display the contents of x
~~~

~~~
print(x)
# Display the type of x
print(typeof(x))
```

```{r,warning=TRUE,message=FALSE,eval=TRUE,echo=TRUE}
# Method 3
x<-c(1.787, 0.63573, 2.3890)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

### Implicit coercion

#### Example 1

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Create a vector
x<-c(1.8)
# Check the type of x
typeof(x)
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Add a character to the vector
x<-c(x,'a')
# Check the type of x
typeof(x)

```

#### Example 2

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Create a vector
x<-c(TRUE)
# Check the type of x
typeof(x)
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Add a number to the vector
x<-c(x,2)
# Check the type of x
typeof(x)
```

#### Example 3

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Create a vector
x<-c('a')
# Check the type of x
typeof(x)
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Add a logical value to the vector
x<-c(x, TRUE)
# Check the type of x
typeof(x)
```


#### Example 4
~~~

~~~
```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Create a vector
x<-c(1L)
# Check the type of x
typeof(x)
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Add a number to the vector
x<-c(x,2)
# Check the type of x
typeof(x)
```

### Explicit coercion

#### Example 1

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Create a vector
x<-c(1L)
# Check the type of x
typeof(x)
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Convert the vector to type character
x<-as.character(x)
# Check the type of x
typeof(x)
```

#### Example 2

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Create a vector
x<-c('A')
# Check the type of x
typeof(x)
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Convert the vector to type double
x<-as.numeric(x)
# Check the type of x
typeof(x)
```

### Accessing elements of the vector

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Create a vector
x <- c(1,10,9,8,1,3,5)
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Access one element with index 3
x[3]
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Access elements with consecutive indices, 2 to 4: 2,3,4
x[2:4]
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Access elements with non-consecutive indices, 1,3,5
x[c(1,3,5)]
~~~

~~~
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Access elements using logical vector
x[c(TRUE,FALSE,FALSE,TRUE,FALSE,FALSE,TRUE)]
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Access elements using the conditional operator <
x[x<10]
```

### Examining vectors

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Display the length of the vector
print(length(x))
# Display the type of the vector
print(typeof(x))
# Display the structure of the vector
print(str(x))
```

### Lists

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Initialise a named list
my_pie = list(type="key lime", diameter=7, is.vegetarian=TRUE)
# display the list
my_pie
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Print the names of the list
names(my_pie)
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Retrieve the element named type
my_pie$type
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Retrieve a truncated list
my_pie["type"]
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Retrieve the element named type
my_pie[["type"]]
```

#### Exploring data-sets

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Install package
# Load the package
library(openintro)
# Load package
library(tidyverse)
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Catch a glimpse of the data-set: see how the rows are stacked one below another
glimpse(loans_full_schema)
```
~~~

~~~
```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Selecting numeric variables
loans <- loans_full_schema %>% # <-- pipe operator
  select(paid_total, term, interest_rate,
         annual_income,paid_late_fees,debt_to_income)
# View the columns stacked one below another
glimpse(loans)
```

```{r,warning=TRUE,message=TRUE,eval=TRUE,echo=TRUE}
# Selecting categoric variables
loans <- loans_full_schema %>%
  select(grade,state,homeownership,disbursement_method ) # type the chosen columns as in
the lecture slide
# View the columns stacked one below another
glimpse(loans)
```
~~~