MTH208: Worksheet 2

Introduction to R cont…

Let's familiarize ourselves with some basic data structures in `R`.

The following store **homogeneous** elements (all of the same type):

Structure	Description	Example
`vector`	1D collection of elements	`c(1, 2, 3)`
`matrix`	2D rectangular array	`matrix(1:6, nrow = 2)`
`array`	N-dimensional generalization of matrix	`array(1:8, dim = c(2, 2, 2))`
`factor`	Categorical variable	`factor(c("yes", "no", "yes"))`

These can hold **different data types**:

Structure	Description	Example
`list`	Collection of elements of any type	`list(name = "Akash", scores = c(85, 90))`
`data.frame`	Tabular data with columns of varying types	`data.frame(name, score)`

Now, let's play with these structures.

1. Try the following -

```r
# Character vector
v <- c("low", "medium", "high", "low")

# Factor
f <- factor(v, levels = c("low", "medium", "high"), ordered = TRUE)

# Output
```

```r
v
f
summary(v)
summary(f)

# Finally
summary(factor(c("yes", "no", "yes")))
```

2. Some exercises in array search; try the following -

   - By index:

     ```r
     x <- c(10, 20, 30, 40)
     x[2]        # returns 20
     x[c(1, 3)] # returns 10 and 30
     ```

   - By logical conditions to filter elements:

     ```r
     x[x > 25]         # returns 30, 40
     which(x > 25)     # returns indices: 3 4
     which(x == 20)    # returns 2
     ```

   - By names:

     ```r
     names(x) <- c("a", "b", "c", "d")
     x["b"]            # returns 20
     x[c("a", "c")]    # returns 10, 30
     ```

   - Using `%in%`

     ```r
     x %in% c(20, 30)      # returns FALSE  TRUE  TRUE FALSE
     x[x %in% c(20, 30)]  # returns 20, 30
     ```

   - Using `match()`

     ```r
     match(30, x)   # returns 3
     ```

   - Using `grep()`

     ```r
     names <- c("Aditi", "Bhanu", "Charu")
     grep("h", names)          # returns indices: 2 3
     names[grep("h", names)]  # returns: "Bhanu" "Charu"
     ```

   - Search in multidimensional array

```r
m <- matrix(1:9, nrow = 3)
m[2, 3]                        # element at 2nd row, 3rd column
which(m > 5, arr.ind = TRUE)   # get positions
```

3. Let's play with a data frame -

   - Create one!

   ```r
   # Define vectors
   Name <- c("Adi", "Bhanu", "Charu", "Dev", "Esha")
   Age <- c(25, 30, 22, 28, 24)
   Gender <- c("Male", "Female", "Female", "Male", "Female")

   # Create data frame
   df <- data.frame(Name, Age, Gender)

   # View the data frame
   print(df)
   ```

   - Use $ operator (access a column by name):

   ```r
   df$column_name   # e.g., df$Age
   ```

   - Use double brackets `[[]]` (access a column by name or position):

   ```r
   df[["Age"]]
   df[[2]]         # Second column
   ```

   - Use single brackets `[,]` (subset by rows and columns):

   ```r
   df[1, ]         # First row
   df[, 2]         # Second column
   df[1, 2]        # Value in first row, second column
   df[1:5, "Age"]  # Rows 1 to 5 of column "Age"
   ```

   - Filter based on condition:

   ```r
   df[df$Age > 25, ]                # Rows where Age > 25
   df[df$Gender == "Male", "Name"]  # Names of all male rows
   ```

   - Filter using the `subset()` function:

   ```r
   subset(df, Age > 25)
   subset(df, Gender == "Female", select = Name)
   ```

- Get row numbers matching conditions using `which()` :

  ```r
  which(df$Age == 30)      # Returns row indices
  df[which(df$Age == 30), ]
  ```

- Search for pattern (strings):

  ```r
  df[grep("ha", df$Name), ]     # Rows where "ha" appears in Name
  ```

We have familiarized ourselves with some starter exercises in `R`. Now let's try to do some visual and exploratory analyses

1. Recall the seating chart for this course

   ```r
   seat <- read.csv("seating.csv")
   ```

   MSc students have 9-digit roll numbers and BS students have 6-digit roll numbers. Write `R` code to calculate the number of MSc students enrolled in this course, and the number of BS students enrolled in this course.

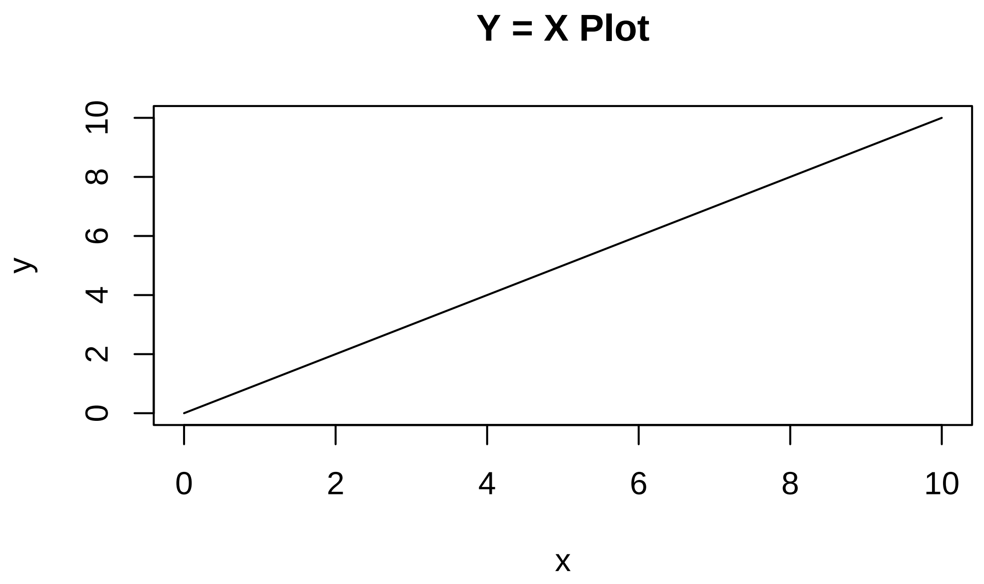
2. An "all-rounder" in cricket is a player who performs well in both batting and bowling. The dataset `battingbowling.csv` in your repository contains the batting and bowling ODI averages of selected male players. Load the dataset using the following command (remember that your working directory should be the directory that contains this file).

   ```r
   cricket <- read.csv("battingbowling.csv")
   ```

   (A high batting average is good, a high bowling average is bad.) Let's say a decent batter is someone with a batting average higher than 25 and a decent bowler is someone with a bowling average below 40.

   1. Create a sub-dataset of all all-rounders using the above criterion.
   2. Which team has the most all-rounders?
   3. Which team has the least all-rounders.

3. The `plot()` function can be used to make a variety of plots in `R`. Do `?plot` on the console to learn how the syntax for plots works. Reproduce the following $y = x$ plot given below.



4. For $n = 1, ..., 1000$, make a plot of n versus $f(n)$ where

$$f(n) = \left(1 + \frac{1}{n}\right)^n$$

Using `abline()`, draw a horizontal line, in red, at the value e.