Introduction to R

- # sign for comments
- Arithmetic operators: +, -, *, /, ^, %% (modulo)
- Variables allow you to store a value (e.g. 4) or an object (e.g. a function description)

```
> # Assign the value 42 to x
> x <- 42
> # Print out the value of x
> x
>
> # Assign a value to the variables my_apples and my_oranges
> my_apples <- 5
> my_oranges <- 6
>
> # Add these two variables together
> my_apples + my_oranges
>
> # Create the variable my_fruit
> my_fruit <- my_apples + my_oranges
```

- **Numerics** are decimals (e.g. 4.5)
- **Integers** are natural numbers
- **Logical** values are Boolean
- **Characters** are string values, (e.g.”this is a string”)

```
> # Set my_numeric to be 42
> my_numeric <- 42
>
> # Set my_character to be "universe"
> my_character <- "universe"
>
> # Set my_logical to be FALSE
> my_logical <- FALSE
```

- Check the data type of a variable by using: **class()**

```
> # Check class of my_numeric
> class(my_numeric)
“numeric”
> # Check class of my_character
```

```
> class(my_character)
“character”
> # Check class of my_logical
> class(my_logical)
“logical”
```

Vectors

- **Vectors** are 1-d arrays that hold numeric, character, or logical data (think python list but of one data type)
    - Create vectors using combine function **c()**

```
> numeric_vector <- c(1, 10, 49)
> character_vector <- c("a", "b", "c")
>
> # Complete the code for boolean_vector
> boolean_vector <- c(TRUE, FALSE, TRUE)
```

- You can name the elements of a vector using **names()** function

```
> vector_a <- c(“John Doe”, “poker player”
> names( vector_a) <- c(“Name”, “Profession”)
```

Exercise - Casino winnings

```
> # Poker winnings & Roulette winnings from Monday to Friday
> poker_vector <- c(140, -50, 20, -120, 240)
> roulette_vector <- c(-24, -50, 100, -350, 10)
>
> # A. Assign days as names of poker_vector, roulette_vectors
> names(poker_vector) <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday")
> names(roulette_vector) <- c("Monday", "Tuesday", "Wednesday", "Thursday",
      ( "Friday")
>
> # B. Assign days as names of poker_vector, roulette_vectors using created variable
> days_vector <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday")
> names(poker_vector) <- days_vector
> names(roulette_vector) <- days_vector
>
> roulette_vector
```

```
Monday   Tuesday Wednesday  Thursday    Friday
   -24         -50         100          -350         10
>
> total_daily <- roulette_vector+poker_vector
> total_daily
Monday   Tuesday Wednesday  Thursday    Friday
   116        -100         120          -470        250
```

- Adding vectors will take the element-wise sum:

```
> A_vector <- c(1, 2, 3)
> B_vector <- c(4, 5, 6)
> # Take the sum of A_vector and B_vector
> total_vector <- A_vector + B_vector
> total_vector
5 7 9
```

- To find the sum of the elements in a vector, use sum()

```
> total_poker <- sum(poker_vector)
> total_roulette <- sum(roulette_vector)
> # Check to see if poker winnings are greater than roulette winnings
> total_poker > total_roulette
TRUE
```

- Selecting element of a vector, matrix, data frame, etc: use **[ ]** (note: first element has index 1, not 0)

```
> # Define a new variable based on a selection
> poker_wednesday <- poker_vector[3]
> poker_wednesday
```

- Select multiple elements of vector, matrix, data frame, etc: use **[c( )]**

```
> # Define a new variable based on a selection of Tuesday, Wednesday, Thursday
> poker_midweek <- poker_vector[c(2,3,4)]
```

- Select multiple elements using splicing "**:**"

```
> # Define a new variable based on a selection of Tuesday to Friday
> roulette_selection_vector <-roulette_vector[2:5]
```

- Select multiple elements using their names

```
> # Select poker results for Monday, Tuesday and Wednesday
> poker_start <- poker_vector[c("Monday", "Tuesday", "Wednesday")]
```

- Calculate the average of a vector using **mean()**

```
> mean(poker_start)
36.66667
```

- Comparison operators:
    - These command operators return TRUE or FALSE

<	>	<=	>=	==	!=

```
> # Which days did you make money on poker?
> selection_vector <- poker_vector>0
>
> # Print out selection_vector
> selection_vector
Monday   Tuesday Wednesday  Thursday    Friday
  TRUE     FALSE      TRUE     FALSE      TRUE
```

- R will select only elements that are TRUE when a logical vector is passed in square brackets

```
> # Which days did you make money on poker?
> selection_vector <- poker_vector > 0
>
> # Select from poker_vector these days
> poker_winning_days <- poker_vector[selection_vector]
>
> poker_winning_days
Monday Wednesday    Friday
  140          20       240
```

Matrices

- A matrix is a collection of elements of the same data type, arranged into a fixed number of rows and columns
- Construct a matrix with **matrix()** function
    - **First argument** is the collection of elements to be arranged into rows and columns
    - Argument **byrow** indicates that the matrix is filled by rows, TRUE or FALSE (FALSE for matrix filled by columns)
    - Argument **nrow** indicates how many rows in the matrix

```
> matrix(1:9, byrow = TRUE, nrow = 3)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9

> # Box office Star Wars (in millions!)
> new_hope <- c(460.998, 314.4)
> empire_strikes <- c(290.475, 247.900)
> return_jedi <- c(309.306, 165.8)
>
> # Create box_office
> box_office <- c(new_hope, empire_strikes, return_jedi)
>
> # Construct star_wars_matrix
> star_wars_matrix <- matrix(box_office, byrow=TRUE, nrow=3)
      [,1]     [,2]
[1,] 460.998 314.4
[2,] 290.475 247.9
[3,] 309.306 165.8
```

- Adding row and column names
    - **rownames(**matrix**)** <- row_names_vector
    - **colnames(**matrix**)** <- col_names_vector

```
> # Vectors region and titles, used for naming
> region <- c("US", "non-US")
> titles <- c("A New Hope", "The Empire Strikes Back", "Return of the Jedi")
>
> # Name the columns with region
> colnames(star_wars_matrix) <- region
>
> # Name the rows with titles
> rownames(star_wars_matrix) <- titles
```

```
>
> # Print out star_wars_matrix
> Star_wars_matrix
                            US     non-US
A New Hope                 460.998   314.4
The Empire Strikes Back    290.475   247.9
Return of the Jedi         309.306   165.8
```

- **rowSums(**matrix**)** calculates the totals for each row of a matrix, creating a new vector as the result

```
> # Construct star_wars_matrix
> box_office <- c(460.998, 314.4, 290.475, 247.900, 309.306, 165.8)
> star_wars_matrix <- matrix(box_office, nrow = 3, byrow = TRUE,
          dimnames = list(c("A New Hope", "The Empire Strikes Back",
          "Return of the Jedi"),  c("US", "non-US")))
>
> # Calculate worldwide box office figures
> worldwide_vector <- rowSums(star_wars_matrix)
> worldwide_vector
>
> A New Hope   The Empire Strikes Back      Return of the Jedi
          775.398               538.375               475.106
```

- Adding columns to matrix using **cbind()**
    - Merges matrices and/or vectors together by column
    - Big_matrix = cbind(matrix1, matrix2, vector1...)

```
> # Bind the new variable worldwide_vector as a column to star_wars_matrix
> all_wars_matrix <- cbind(star_wars_matrix, worldwide_vector)

                            US       non-US     worldwide_vector
A New Hope                 460.998  314.4         775.398
The Empire Strikes Back    290.475  247.9         538.375
Return of the Jedi         309.306  165.8         475.106
```

- Adding rows to matrix with **rbind**()

```
> star_wars_matrix2
                          US    non-US
The Phantom Menace      474.5  552.5
Attack of the Clones    310.7  338.7
Revenge of the Sith      380.3  468.5
```

```
> # Combine both Star Wars trilogies in one matrix
> all_wars_matrix <- rbind(star_wars_matrix, star_wars_matrix2)
> all_wars_matrix
                            US     non-US
A New Hope                 461.0  314.4
The Empire Strikes Back    290.5  247.9
Return of the Jedi         309.3  165.8
The Phantom Menace         474.5  552.5
Attack of the Clones       310.7  338.7
Revenge of the Sith        380.3  468.5
```

- **colSums()** to calculate totals for each column of a matrix

```
> # Total revenue for US and non-US
> total_revenue_vector <- colSums(all_wars_matrix)
>
> # Print out total_revenue_vector
> total_revenue_vector
  US       non-US
2226.3   2087.8
```

- Selection of matrix elements using brackets
    - My_matrix[1,2] selects element at the first row and second column
    - My_matrix[1:3, 2:4] results in a matrix with the data on the rows 1, 2, 3 and columns 2, 3, 4
    - My_matrix[,1] selects all elements of the first column
    - My_matrix[1,] selects all elements of the first row

```
> # Select the non-US revenue for all movies
> non_us_all <- all_wars_matrix[,2]
>
> # Average non-US revenue
> mean(non_us_all)
[1] 347.9667
> # Select the non-US revenue for first two movies
> non_us_some <- all_wars_matrix[1:2, 2]
>
> # Average non-US revenue for first two movies
> mean(non_us_some)
[1] 281.15
```

- Arithmetic on matrices
    - 2 * my_matrix will multiply each element of the matrix by 2

- my_matrix/10 will divide each element of the matrix by 10
- My_matrix1 * my_matrix2 will create a matrix where each element is a product of the corresponding elements in the two matrices
    - Not the classic multiplication of matrices

```
> # Estimate the visitors, assuming each ticket is $5
> visitors <- all_wars_matrix/5
>
> # Print the estimate to the console
> visitors
                           US non-US
A New Hope                 92.20  62.88
The Empire Strikes Back 58.10  49.58
Return of the Jedi         61.86  33.16
The Phantom Menace     94.90 110.50
Attack of the Clones       62.14  67.74
Revenge of the Sith       76.06  93.70

> ticket_prices_matrix
                           US non-US
A New Hope                  5.0    5.0
The Empire Strikes Back   6.0    6.0
Return of the Jedi          7.0    7.0
The Phantom Menace       4.0    4.0
Attack of the Clones        4.5    4.5
Revenge of the Sith         4.9    4.9

> # Estimated number of visitors
> visitors <- all_wars_matrix/ticket_prices_matrix
>
> # US visitors
> us_visitors <- visitors[,1]
>
> # Average number of US visitors
> mean(us_visitors)
[1] 75.01401
```

Factors

- Factor is a data type used to store categorical variables (limited number of categories)
- To create a factor, use function **factor()**
- Factor levels are also known as the number of categories

```
> # Sex vector
> sex_vector <- c("Male", "Female", "Female", "Male", "Male")
>
```

```
> # Convert sex_vector to a factor
> factor_sex_vector <- factor(sex_vector)
>
> # Print out factor_sex_vector
> factor_sex_vector
[1] Male Female Female Male Male
Levels: Female Male
```

- Nominal categorical variables: categorical variable without an implied order
    - Ex: categories are “Elephant”, “Giraffe”, and “Cow”
- Ordinal categorical variables: categorical variables with natural ordering
    - Ex: categories are “Low”, “Medium”, and “High”
- Changing the names of factor levels using **levels()**
    - levels(factor_vector) <- c(“name1”, “name2”,...)

```
> # Code to build factor_survey_vector
> survey_vector <- c("M", "F", "F", "M", "M")
> factor_survey_vector <- factor(survey_vector)
>
> # Specify the levels of factor_survey_vector
> levels(factor_survey_vector) <-c("Female", "Male")
>
> factor_survey_vector
[1] Male Female Female Male Male
Levels: Female Male
```

- Using **summary()** to see a quick overview of the contents of a variable

```
> summary(factor_survey_vector)
Female   Male
    2      3
```