# Basic Course on R: Programming Structures 1 Practical

Elizabeth Ribble[*]

20-24 May 2019

## Contents

[*]emcclel3@msudenver.edu

# 1 Part A: `if()`, `else`, and `ifelse()` and Vectorization

1. Write a function `evenOrOdd()` involving `if()` and `else` that takes an argument `x` and returns `"Even"` or `"Odd"` depending on whether or not `x` is divisible by 2. (*Do not* use the `ifelse()` function).

2. Is your function `evenOrOdd()` *vectorized*? Check by passing it the vector:

```
w <- c(3, 6, 6, 4, 7, 9, 11, 6)
```

3. Another way to determine if each element of a vector is even or odd is to use the `ifelse()` function, which serves as a vectorized version `if()` and `else`. Use `ifelse()` to obtain `"Even"` or `"Odd"` for each element of `w`.

# 2 Part B: Loops

1. How many times will `"Frisbee Sailing"` be printed to the screen in each of the following sets of commands? Try to answer without using R.

```
a) i <- 5
   while(i < 1) {
     print("Frisbee Sailing")
     i <- i + 1
   }
```

```
b) i <- 0
   while(i < 5) {
     print("Frisbee Sailing")
   }
```

```
c) i <- 0
   while(i < 5) {
     print("Frisbee Sailing")
     i <- i + 1
   }
```

2. How many times will **"Masked Marvel"** be printed to the screen in the following set of commands? Try to answer without using R.

```r
i <- 1
repeat {
  if(i > 5) break
  print("Masked Marvel")
  i <- i + 1
}
```

3. The file **kennedys.txt** has a command to create a list containing two generations of the famous Kennedy family:

```r
Kennedys <- list(
    JosephJr = character(0),
    John = c("Caroline", "JohnJr", "Patrick"),
    Rosemary = character(0),
    Kathleen = character(0),
    Eunice = c("RobertIII", "Maria", "Timothy", "Mark", "Anthony"),
    Patricia = c("Christopher", "Sydney", "Victoria", "Robin"),
    Robert = c("Kathleen", "JosephII", "RobertJr", "David",
               "MaryC", "Michael", "MaryK", "Christopher",
               "Matthew", "Douglas", "Rory"),
    Jean = c("Stephen", "William", "Amanda", "Kym"),
    Edward = c("Kara", "EdwardJr", "Patrick")
)
```

Read in the file with the use of `source()` and type `ls()` to see if the list was created (type `Kennedys` to view the object).

Loop over the list of the first generation of Kennedys, keeping track of how many children each one has in a vector.

# 3  Part C: Logical Operators &, |, and !

1. What will be the result of the following:

```
(10 < 20 && 15 < 16) || 9 == 10
```

2. One of the following evaluates to `TRUE`, the other to `FALSE`. Which is which?

```
4 < 3 && (5 < 6 || 8 < 9)
(4 < 3 && 5 < 6) || 8 < 9
```

3. The data set below contains the systolic and diastolic blood pressure readings for 22 patients (and can be found in the file **BPressure.txt**).

| PatientID | Systolic | Diastolic |
|---|---|---|
| CK | 120 | 50 |
| SS | 96 | 60 |
| FR | 100 | 70 |
| CP | 120 | 75 |
| BL | 140 | 90 |
| ES | 120 | 70 |
| CP | 165 | 110 |
| JI | 110 | 40 |
| MC | 119 | 66 |
| FC | 125 | 76 |
| RW | 133 | 60 |
| KD | 108 | 54 |
| DS | 110 | 50 |
| JW | 130 | 80 |
| BH | 120 | 65 |
| JW | 134 | 80 |
| SB | 118 | 76 |
| NS | 122 | 78 |
| GS | 122 | 70 |
| AB | 122 | 78 |
| EC | 112 | 62 |
| HH | 122 | 82 |

a) Read the data from **BPressure.txt** into a data frame called `bp` using `read.table()`.

b) A person's blood pressure is classified as normal if the systolic level is below 120 and the diastolic level is below 80. Use square brackets `[ ]` to extract from `bp` the rows corresponding to patients with normal blood pressures.

c) Now use square brackets `[ ]` to extract the rows corresponding to patients whose blood pressures *aren't* normal.

4