Week-6: Code-along

Chua Jieh Yih Audra

2023-09-18

II. Code to edit and execute using the Code-along-6.Rmd file

A. `for` loop

1. Simple `for` loop (Slide #6)

```
# Enter code here
for (x in c(3, 6, 9)) {
  print(x)
  }
```

```
## [1] 3
## [1] 6
## [1] 9
```

2. `for` loops structure (Slide #7)

```
# Left-hand side code: for loop for passing values
for (x in 1:8) {
  print(x)
  }
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
```

```
# Right-hand side code: for loop for passing indices
for (x in 1:8) {
  y <- seq(from=100,to=200,by=5)
  print(y[x])
  }
```

```
## [1] 100
## [1] 105
## [1] 110
## [1] 115
## [1] 120
## [1] 125
## [1] 130
## [1] 135
```

3. Example: find sample means (Slide #9)

```
# Enter code here
sample_sizes <- c(5, 10, 15, 20, 25000)
# pre-allocatting space to store output
sample_means <- double(length(sample_sizes))
#loop
for (i in seq_along(sample_sizes)) {
  sample_means[i] <- mean(rnorm(sample_sizes[i]))
  }
sample_means
```

```
## [1] -0.265780775  0.394769877 -0.072731082  0.534396828 -0.005698245
```

4. Alternate ways to pre-allocate space (Slide #12)

```
# Example 1 for data_type=double
sample_means <- vector("double", length = 5)
# Example 2 for data_type=double
sample_means <- double(5)
# Example 3 for data_type=double
sample_means <- rep(0, length(sample_sizes))
```

*Added the others for my own reference

```
# Initialisation of data_list
data_list <- vector("list", length = 5)
```

5. Review: Vectorized operations (Slide #18)

```
# Example: bad idea!
a <- 7:11
b <- 8:12
# Vector of all zeros of length 5
out <- rep(0L, 5)

# Loop along the length of vector a
for (i in seq_along(a)) {
  # Each entry of out is the sum of the corresponding vector... numbers
  out[i] <- a[i] + b[i]
  }
out
```

```
## [1] 15 17 19 21 23
```

```
# Taking advantage of vectorization
a <- 7:11
b <- 8:12
out <- a + b
out
```

```
## [1] 15 17 19 21 23
```

B. Functionals

6. `for` loops vs Functionals (Slides #23 and #24)

```
# Slide 23
sample_sizes <- c(5, 10, 15, 20, 25000)

# Create a functional- function inside a function
sample_summary <- function(sample_sizes, fun) {
  # Initialise a vector of the same size as sample_sizes
  out <- vector("double", length(sample_sizes))
    # Run the for loop for as long as the length of sample_sizes
    for (i in seq_along(sample_sizes)) {
    # Perform operations indicated by fun
    out[i] <- fun(rnorm(sample_sizes[i]))
    }
  return(out)
  }
```

```
# Slide 24
#Compute mean
sample_summary(sample_sizes,mean)
```

```
## [1] -0.28979221  0.45379797  0.43804603  0.04465742 -0.01383041
```

```
# Compute median
sample_summary(sample_sizes,median)
```

```
## [1] -0.233356950 -0.484412679 -0.492414602  0.221082745  0.004578289
```

```
# Compute sd
sample_summary(sample_sizes,sd)
```

```
## [1] 1.3294526 0.6863330 0.8648391 0.9396210 1.0009055
```

C. `while` loop

7. `while` loop (Slides #27)

```
# Left-hand side code: for loop
for(i in 1:5){
  print(i)
  }
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
```

```
# Right-hand side code: while loop
i <- 1
while (i <= 5) {
  # body
  print(i)
  i <- i + 1
  }
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
```