Week-6: Code-along

Esther Kho Yining

'2024-02-20

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)

```r
# Enter code here
sample_sizes <- c(5, 10, 15, 20, 25000)
sample_means <- double(length(sample_sizes))

for(i in seq_along(sample_sizes)){
  sample_means[i] <- mean(rnorm(sample_sizes[i]))
}

sample_means
```

```
## [1] -0.033159245 -0.388699557  0.155253282 -0.001055524  0.007190612
```

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

```r
# Example 3 for data_type=double
sample_means <- rep(0, length(sample_sizes))
```

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

5. Review: Vectorized operations (Slide #18)

```r
# Example: bad idea!
a <- 7:11
b <- 8:12
out <- rep(0L, 5)

for(i in seq_along(a)){
  out[i] <- a[i] + b[i]
}

out
```

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

```r
# 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)

```r
# Slide 23
sample_sizes <- c(5, 10, 15, 20, 25000)
sample_summary <- function(sample_sizes, fun){
  out <- vector("double", length(sample_sizes))
  for (i in seq_along(sample_sizes)) {
    out[i] <- fun(rnorm(sample_sizes[i]))
  }
  return(out)
}
```

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

```
## [1]  0.41429644  0.31224371  0.12668940 -0.36337944  0.00229969
```

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

```
## [1]  1.26920193  0.16281814  0.70877278 -0.22022143  0.01618724
```

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

```
## [1] 0.8764353 0.8386719 0.9945842 0.8484311 0.9977185
```

C. `while` loop

7. `while` loop (Slides #27)

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

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

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

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