Week 2 - Coursera Data Science

Martin Aragoneses

July 22, 2015

`if ... else` statements:

```
if(1>2) {
  print(1)
} else if(2>1) {
  print(2)
} else {
  print("bye")
}
```

```
## [1] 2
```

`for` loops:

```
x <- c("a", "b", "c", "d")

for(i in seq_along(x)) {
  print(x[i])
}
```

```
## [1] "a"
## [1] "b"
## [1] "c"
## [1] "d"
```

```
x <- matrix(1:6, 2, 3)

for(i in seq_len(nrow(x))) {
  for (j in seq_len(ncol(x))) {
    print(x[i, j])
  }
}
```

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

`while` loops:

```
count = 0
while(count<10) {
  print(count)
  count <- count + 1
}
```

```
## [1] 0
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
## [1] 9
```

```
# Random Walk
z <- 5

while(z >= 3 && z<= 10) {
  print(z)
  coin <- rbinom(1, 1, 0.5)  # flip a fair coin

  if(coin == 1) {    # random walk
    z <- z + 1
  } else {
    z <- z - 1
  }
}
```

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

repear, next, break:

repeat until you find a break

next skips an iteration in a loop

Your first R function:

```
above <- function(x, n = 10) {
  use <- x > n
  x[use]
}
```

```
columnMean <- function(x, removeNA = TRUE) {
  ## x is a data frame or a matrix
  ## we are going to go through each column
  ## and calculate its mean

  nc <- ncol(x)
  ## Initialize an empty (numeric) vector with as many elements as columns in the data frame/matrix
  means <- numeric(nc)
  for(i in 1:nc) {
    # calculate the mean for each column i of all the individuals and fill it in the i-th index of the
    means[i] <- mean(x[, i], na.rm = removeNA)
  }
  means

}
```

Lexical Scoping:

```
make.power <- function(n) {
  pow <- function(x) {
    x^n
  }
  pow
}

cube <- make.power(3)
square <- make.power(2)
cube(3)
```

```
## [1] 27
```

```
square(4)
```

```
## [1] 16
```

Lexical vs. Dynamic Scoping (which is what other languages use)

```
y <- 10

f <- function(x) {
  y <- 2
  y^2 + g(x)
}

g <- function(x) {
  x*y
}

f(3)
```

```
## [1] 34
```