Weighted Dice Very Basics

R Objects

Assignment

```
die1 <- c(1, 2, 3, 4, 5, 6)
die2 <- 1:6
die1 == die2
```

```
## [1] TRUE TRUE TRUE TRUE TRUE TRUE
```

```
die <- die2
ls()
```

```
## [1] "die"  "die1" "die2"
```

Vectorized Operations

Check how the mathematical operations are regarded as functions, and how to bring up help pages

```
die - 1
```

```
## [1] 0 1 2 3 4 5
```

```
die / 2
```

```
## [1] 0.5 1.0 1.5 2.0 2.5 3.0
```

```
die * die
```

```
## [1]  1  4  9 16 25 36
```

```
1:2
```

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

```
1:4
```

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

```
die + 1:2
```

```
## [1] 2 4 4 6 6 8
```

```
die + 1:4
```

```
## Warning in die + 1:4: longer object length is not a multiple of shorter
## object length
```

```
## [1] 2 4 6 8 6 8
```

```
die %*% die
```

```
##      [,1]
## [1,]   91
```

```
die %o% die
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    1    2    3    4    5    6
## [2,]    2    4    6    8   10   12
## [3,]    3    6    9   12   15   18
## [4,]    4    8   12   16   20   24
## [5,]    5   10   15   20   25   30
## [6,]    6   12   18   24   30   36
```

```
outer(die, die)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    1    2    3    4    5    6
## [2,]    2    4    6    8   10   12
## [3,]    3    6    9   12   15   18
## [4,]    4    8   12   16   20   24
## [5,]    5   10   15   20   25   30
## [6,]    6   12   18   24   30   36
```

```
# help("%*%")
# help("%o%")
# ?`%*%`
# ?outer
```

Functions

```
round(3.1415)
```

```
## [1] 3
```

```
round(3.1415, digits = 2)
```

```
## [1] 3.14
```

```
factorial(3)
```

```
## [1] 6
```

```
mean(1:6)
```

```
## [1] 3.5
```

```
mean(die)
```

```
## [1] 3.5
```

```
sd(die)
```

```
## [1] 1.870829
```

```
round(mean(die), digits = 2)
```

```
## [1] 3.5
```

Sampling

`x` for the population, `size` for the sample size. Note how the arguments are recognized.

```
# ?sample
sample(die)
```

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

```
sample(x = 1:4, size = 2)
```

```
## [1] 1 4
```

```
sample(x = die, size = 1)
```

```
## [1] 5
```

```
sample(x = die, size = 1)
```

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

```
sample(x = die, size = 1)
```

```
## [1] 6
```

```
sample(die, size = 1)
```

```
## [1] 5
```

```
# round(3.1415, corners = 2)
args(round)
```

```
## function (x, digits = 0)
## NULL
```

```
round(3.1415, digits = 2)
```

```
## [1] 3.14
```

```
sample(die, 1)
```

```
## [1] 6
```

```
sample(size = 1, x = die)
```

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

```
# ?sample
```

Sample with Replacement

```
sample(die, size = 2)
```

```
## [1] 6 4
```

```
sample(die, size = 2, replace = TRUE)
```

```
## [1] 4 4
```

```
sample(die, size = 2, replace = TRUE)
```

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

```
dice <- sample(die, size = 2, replace = TRUE)
dice
```

```
## [1] 1 5
```

```
sum(dice)
```

```
## [1] 6
```

Writing Your Own Functions

The Function Constructor

Simulate sum of two tosses of a die, or the sum of two dice thrown. The difference between `roll()` and `roll`.

```
roll <- function() {
  die <- 1:6
  dice <- sample(die, size = 2, replace = TRUE)
  sum(dice)
}
roll()
```

```
## [1] 5
```

```
roll
```

```
## function() {
##   die <- 1:6
##   dice <- sample(die, size = 2, replace = TRUE)
##   sum(dice)
## }
```

Arguments

How to implement an input variable, Why we need to set up a default value.

```
roll2 <- function(bones) {
  dice <- sample(bones, size = 2, replace = TRUE)
  sum(dice)
}
roll2(bones = 1:4)
```

```
## [1] 4
```

```
roll2(bones = 1:5)
```

```
## [1] 10
```

```
roll2(1:20)
```

```
## [1] 31
```

```
# roll2()
```

Default Value

```
roll2 <- function(bones = 1:6) {
  dice <- sample(bones, size = 2, replace = TRUE)
  sum(dice)
}
roll2()
```

```
## [1] 8
```

Dump and Source

```
dump(list = c("roll", "roll2"), file = "./roll.R")
rm(list = ls())
ls()
```

```
## character(0)
```

```
source("./roll.R")
ls()
```

```
## [1] "roll"  "roll2"
```