Basic Operations

John Eric Humphries, Oliver Browne*

March 31, 2014

1 Basic Programming Principles

- KISS - Keep it Simple Stupid
- DRY - Don't Repeat Yourself
- Abstraction Principle
- Google's R Style guide

2 Some basic R code

Basic Types

```
# Different types in R
x <- 1
y <- TRUE
z <- "the quick brown fox jumped over the lazy dog"
typeof(x)

## [1] "double"

typeof(y)

## [1] "logical"

typeof(z)

## [1] "character"
```

*email: obrowne@uchicago.edu

```r
x <- sqrt(as.complex(-1))
x
```

```
## [1] 0+1i
```

```r
typeof(x)
```

```
## [1] "complex"
```

Defining Vectors

```r
# This is how you create vectors
x <- c(1:10)
x
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```r
y <- c(15:6)
y
```

```
##  [1] 15 14 13 12 11 10  9  8  7  6
```

```r
typeof(x)
```

```
## [1] "integer"
```

```r
length(x)
```

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

Chopping Vectors

```r
# Chopping Vectors
y[3]
```

```
## [1] 13
```

```r
y[-3]
```

```
## [1] 15 14 12 11 10  9  8  7  6
```

```r
y[x < 7 & x >= 4]
```

```
## [1] 12 11 10
```

```r
y[1:5]
```

```
## [1] 15 14 13 12 11

y[-(1:5)]

## [1] 10  9  8  7  6

sort(y)

##  [1]  6  7  8  9 10 11 12 13 14 15
```

Operations on Vectors

```
# Element Wise Operations
x + y

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

y - x

##  [1] 14 12 10  8  6  4  2  0 -2 -4

x/y

##  [1] 0.06667 0.14286 0.23077 0.33333 0.45455 0.60000 0.77778 1.00000 1.28571 1.66667

x * y

##  [1] 15 28 39 48 55 60 63 64 63 60

# Vector Outer Product
x %o% y

##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]   15   14   13   12   11   10    9    8    7     6
##  [2,]   30   28   26   24   22   20   18   16   14    12
##  [3,]   45   42   39   36   33   30   27   24   21    18
##  [4,]   60   56   52   48   44   40   36   32   28    24
##  [5,]   75   70   65   60   55   50   45   40   35    30
##  [6,]   90   84   78   72   66   60   54   48   42    36
##  [7,]  105   98   91   84   77   70   63   56   49    42
##  [8,]  120  112  104   96   88   80   72   64   56    48
##  [9,]  135  126  117  108   99   90   81   72   63    54
## [10,]  150  140  130  120  110  100   90   80   70    60
```

Basic Functions

```r
# Some basic functions
max(x)

## [1] 10

which.max(x)

## [1] 10

range(x)

## [1]  1 10

mean(x)

## [1] 5.5

median(x)

## [1] 5.5

quantile(x)

##    0%   25%   50%   75%  100%
##  1.00  3.25  5.50  7.75 10.00

cumsum(x)

##  [1]  1  3  6 10 15 21 28 36 45 55

(x > y)

##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE

which(x > y)

## [1]  9 10
```

Defining Matrices

```r
# Defining Matricies
z <- matrix(c(1, 2, 3, 4, 5, 6), nrow = 2)
x <- matrix(c(4, 5, 6, 7, 8, 9), nrow = 2)
x

##      [,1] [,2] [,3]
```

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

t(x)

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

nrow(x)

## [1] 2

ncol(x)

## [1] 3
```

Matrix Operations

```
# Elementwise Matrix Operations
x + z

##      [,1] [,2] [,3]
## [1,]    5    9   13
## [2,]    7   11   15

x * z

##      [,1] [,2] [,3]
## [1,]    4   18   40
## [2,]   10   28   54

x/z

##      [,1] [,2] [,3]
## [1,]  4.0 2.00  1.6
## [2,]  2.5 1.75  1.5

# Matrix Multiplication
x %*% t(z)

##      [,1] [,2]
## [1,]   62   80
## [2,]   71   92
```

Matrix Inversion

```
# Matrix Invesion
x <- matrix(1:4, nrow = 2)
xinv <- solve(x)
x %*% xinv

##      [,1] [,2]
## [1,]    1    0
## [2,]    0    1
```

Random Number Generation

```
# Generating Random Generation
set.seed(1234)
`?`(rnorm)
`?`(runif)
rnorm(1)

## [1] -1.207

x <- rnorm(1000)
summary(x)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -3.400  -0.673  -0.040  -0.027   0.616   3.200

mean(x)

## [1] -0.0266

var(x)

## [1] 0.9947

x2 <- rnorm(1000, 5, 3)
mean(x2)

## [1] 5.044

var(x2)

## [1] 8.66
```

Defining Functions

```r
# Defining Functions
addxy <- function(x, y) {
    # Adds x and y Inuput: x,y Output x+y
    return(x + y)
}
addxy(3, 4)

## [1] 7
```