Wickham Exercises: Functions

Elizabeth Brannon, Kesicia Dickinson, Shane Wery

2/21/2018

Functions

1. What function allows you to tell if an object is a function? What function allows you to tell if a function is a primitive function?

We could use str() or is.function to tell us if its a function. Similarly, we can use is.primitive to tell if it is a primitive function.

2. This code makes a list of all functions in the base package. Use it to answer the following questions:

```
objs <- mget(ls("package:base"), inherits = TRUE)
funs <- Filter(is.function, objs)
```

a Which base function has the most arguments?

```
require(plyr)
```

```
## Loading required package: plyr
```

```
funclength <- laply(funs, function(x)(length(formals(x)))) #creates the length of the function for the

funclength[which(funclength == max(funclength))] #tells us which length is longest
```

```
## [1] 22
```

b How many base functions have no arguments? What's special about those functions?

```
length(funclength[which(funclength == 0)])
```

```
## [1] 225
```

c How could you adapt the code to find all primitive functions?

```
primfunc <- Filter(is.primitive, objs)
primfunc
```

```
## $`-`
## function (e1, e2)  .Primitive("-")
##
## $`:`
## .Primitive(":")
##
## $`!`
## function (x)  .Primitive("!")
##
## $`!=`
## function (e1, e2)  .Primitive("!=")
##
## $`(`
## .Primitive("(")
##
## $`[`
## .Primitive("[")
##
## $`[[`
## .Primitive("[[")
##
## $`[[<-`
## .Primitive("[[<-")
##
## $`[<-`
## .Primitive("[<-")
##
## $`{`
## .Primitive("{")
##
## $`@`
## .Primitive("@")
##
## $`@<-`
## .Primitive("@<-")
##
## $`*`
## function (e1, e2)  .Primitive("*")
##
## $`/`
## function (e1, e2)  .Primitive("/")
##
## $`&`
## function (e1, e2)  .Primitive("&")
##
## $`&&`
## .Primitive("&&")
##
## $`%*%`
## function (x, y)  .Primitive("%*%")
##
## $`%/%`
## function (e1, e2)  .Primitive("%/%")
##
```

```
## $`%%`
## function (e1, e2)  .Primitive("%%")
##
## $`^`
## function (e1, e2)  .Primitive("^")
##
## $`+`
## function (e1, e2)  .Primitive("+")
##
## $`<`
## function (e1, e2)  .Primitive("<")
##
## $`<-`
## .Primitive("<-")
##
## $`<<-`
## .Primitive("<<-")
##
## $`<=`
## function (e1, e2)  .Primitive("<=")
##
## $`=`
## .Primitive("=")
##
## $`==`
## function (e1, e2)  .Primitive("==")
##
## $`>`
## function (e1, e2)  .Primitive(">")
##
## $`>=`
## function (e1, e2)  .Primitive(">=")
##
## $`|`
## function (e1, e2)  .Primitive("|")
##
## $`||`
## .Primitive("||")
##
## $`~`
## .Primitive("~")
##
## $`$`
## .Primitive("$")
##
## $`$<-`
## .Primitive("$<-")
##
## $abs
## function (x)  .Primitive("abs")
##
## $acos
## function (x)  .Primitive("acos")
##
```

```
## $acosh
## function (x)  .Primitive("acosh")
##
## $all
## function (..., na.rm = FALSE)  .Primitive("all")
##
## $any
## function (..., na.rm = FALSE)  .Primitive("any")
##
## $anyNA
## function (x, recursive = FALSE)  .Primitive("anyNA")
##
## $Arg
## function (z)  .Primitive("Arg")
##
## $as.call
## function (x)  .Primitive("as.call")
##
## $as.character
## function (x, ...)  .Primitive("as.character")
##
## $as.complex
## function (x, ...)  .Primitive("as.complex")
##
## $as.double
## function (x, ...)  .Primitive("as.double")
##
## $as.environment
## function (x)  .Primitive("as.environment")
##
## $as.integer
## function (x, ...)  .Primitive("as.integer")
##
## $as.logical
## function (x, ...)  .Primitive("as.logical")
##
## $as.numeric
## function (x, ...)  .Primitive("as.double")
##
## $as.raw
## function (x)  .Primitive("as.raw")
##
## $asin
## function (x)  .Primitive("asin")
##
## $asinh
## function (x)  .Primitive("asinh")
##
## $atan
## function (x)  .Primitive("atan")
##
## $atanh
## function (x)  .Primitive("atanh")
##
```

```
## $attr
## function (x, which, exact = FALSE)  .Primitive("attr")
##
## $`attr<-`
## function (x, which, value)  .Primitive("attr<-")
##
## $attributes
## function (obj)  .Primitive("attributes")
##
## $`attributes<-`
## function (obj, value)  .Primitive("attributes<-")
##
## $baseenv
## function ()  .Primitive("baseenv")
##
## $`break`
## .Primitive("break")
##
## $browser
## function (text = "", condition = NULL, expr = TRUE, skipCalls = 0L)  .Primitive("browser")
##
## $c
## function (...)  .Primitive("c")
##
## $call
## function (name, ...)  .Primitive("call")
##
## $ceiling
## function (x)  .Primitive("ceiling")
##
## $class
## function (x)  .Primitive("class")
##
## $`class<-`
## function (x, value)  .Primitive("class<-")
##
## $Conj
## function (z)  .Primitive("Conj")
##
## $cos
## function (x)  .Primitive("cos")
##
## $cosh
## function (x)  .Primitive("cosh")
##
## $cospi
## function (x)  .Primitive("cospi")
##
## $cummax
## function (x)  .Primitive("cummax")
##
## $cummin
## function (x)  .Primitive("cummin")
##
```

```
## $cumprod
## function (x)  .Primitive("cumprod")
##
## $cumsum
## function (x)  .Primitive("cumsum")
##
## $digamma
## function (x)  .Primitive("digamma")
##
## $dim
## function (x)  .Primitive("dim")
##
## $`dim<-`
## function (x, value)  .Primitive("dim<-")
##
## $dimnames
## function (x)  .Primitive("dimnames")
##
## $`dimnames<-`
## function (x, value)  .Primitive("dimnames<-")
##
## $emptyenv
## function ()  .Primitive("emptyenv")
##
## $enc2native
## function (x)  .Primitive("enc2native")
##
## $enc2utf8
## function (x)  .Primitive("enc2utf8")
##
## $`environment<-`
## function (fun, value)  .Primitive("environment<-")
##
## $exp
## function (x)  .Primitive("exp")
##
## $expm1
## function (x)  .Primitive("expm1")
##
## $expression
## function (...)  .Primitive("expression")
##
## $floor
## function (x)  .Primitive("floor")
##
## $`for`
## .Primitive("for")
##
## $forceAndCall
## function (n, FUN, ...)  .Primitive("forceAndCall")
##
## $`function`
## .Primitive("function")
##
```

```
## $gamma
## function (x)  .Primitive("gamma")
##
## $gc.time
## function (on = TRUE)  .Primitive("gc.time")
##
## $globalenv
## function ()  .Primitive("globalenv")
##
## $`if`
## .Primitive("if")
##
## $Im
## function (z)  .Primitive("Im")
##
## $interactive
## function ()  .Primitive("interactive")
##
## $invisible
## function (x)  .Primitive("invisible")
##
## $is.array
## function (x)  .Primitive("is.array")
##
## $is.atomic
## function (x)  .Primitive("is.atomic")
##
## $is.call
## function (x)  .Primitive("is.call")
##
## $is.character
## function (x)  .Primitive("is.character")
##
## $is.complex
## function (x)  .Primitive("is.complex")
##
## $is.double
## function (x)  .Primitive("is.double")
##
## $is.environment
## function (x)  .Primitive("is.environment")
##
## $is.expression
## function (x)  .Primitive("is.expression")
##
## $is.finite
## function (x)  .Primitive("is.finite")
##
## $is.function
## function (x)  .Primitive("is.function")
##
## $is.infinite
## function (x)  .Primitive("is.infinite")
##
```

```
## $is.integer
## function (x)  .Primitive("is.integer")
##
## $is.language
## function (x)  .Primitive("is.language")
##
## $is.list
## function (x)  .Primitive("is.list")
##
## $is.logical
## function (x)  .Primitive("is.logical")
##
## $is.matrix
## function (x)  .Primitive("is.matrix")
##
## $is.na
## function (x)  .Primitive("is.na")
##
## $is.name
## function (x)  .Primitive("is.symbol")
##
## $is.nan
## function (x)  .Primitive("is.nan")
##
## $is.null
## function (x)  .Primitive("is.null")
##
## $is.numeric
## function (x)  .Primitive("is.numeric")
##
## $is.object
## function (x)  .Primitive("is.object")
##
## $is.pairlist
## function (x)  .Primitive("is.pairlist")
##
## $is.raw
## function (x)  .Primitive("is.raw")
##
## $is.recursive
## function (x)  .Primitive("is.recursive")
##
## $is.single
## function (x)  .Primitive("is.single")
##
## $is.symbol
## function (x)  .Primitive("is.symbol")
##
## $isS4
## function (object)  .Primitive("isS4")
##
## $lazyLoadDBfetch
## function (key, file, compressed, hook)  .Primitive("lazyLoadDBfetch")
##
```

```
## $length
## function (x)  .Primitive("length")
##
## $`length<-`
## function (x, value)  .Primitive("length<-")
##
## $`levels<-`
## function (x, value)  .Primitive("levels<-")
##
## $lgamma
## function (x)  .Primitive("lgamma")
##
## $list
## function (...)  .Primitive("list")
##
## $log
## function (x, base = exp(1))  .Primitive("log")
##
## $log10
## function (x)  .Primitive("log10")
##
## $log1p
## function (x)  .Primitive("log1p")
##
## $log2
## function (x)  .Primitive("log2")
##
## $max
## function (..., na.rm = FALSE)  .Primitive("max")
##
## $min
## function (..., na.rm = FALSE)  .Primitive("min")
##
## $missing
## function (x)  .Primitive("missing")
##
## $Mod
## function (z)  .Primitive("Mod")
##
## $names
## function (x)  .Primitive("names")
##
## $`names<-`
## function (x, value)  .Primitive("names<-")
##
## $nargs
## function ()  .Primitive("nargs")
##
## $`next`
## .Primitive("next")
##
## $nzchar
## function (x, keepNA = FALSE)  .Primitive("nzchar")
##
```

```
## $oldClass
## function (x)  .Primitive("oldClass")
##
## $`oldClass<-`
## function (x, value)  .Primitive("oldClass<-")
##
## $on.exit
## function (expr = NULL, add = FALSE)  .Primitive("on.exit")
##
## $pos.to.env
## function (x)  .Primitive("pos.to.env")
##
## $proc.time
## function ()  .Primitive("proc.time")
##
## $prod
## function (..., na.rm = FALSE)  .Primitive("prod")
##
## $quote
## function (expr)  .Primitive("quote")
##
## $range
## function (..., na.rm = FALSE)  .Primitive("range")
##
## $Re
## function (z)  .Primitive("Re")
##
## $rep
## function (x, ...)  .Primitive("rep")
##
## $`repeat`
## .Primitive("repeat")
##
## $retracemem
## function (x, previous = NULL)  .Primitive("retracemem")
##
## $return
## .Primitive("return")
##
## $round
## function (x, digits = 0)  .Primitive("round")
##
## $seq_along
## function (along.with)  .Primitive("seq_along")
##
## $seq_len
## function (length.out)  .Primitive("seq_len")
##
## $seq.int
## function (from, to, by, length.out, along.with, ...)  .Primitive("seq.int")
##
## $sign
## function (x)  .Primitive("sign")
##
```

```
## $signif
## function (x, digits = 6)  .Primitive("signif")
##
## $sin
## function (x)  .Primitive("sin")
##
## $sinh
## function (x)  .Primitive("sinh")
##
## $sinpi
## function (x)  .Primitive("sinpi")
##
## $sqrt
## function (x)  .Primitive("sqrt")
##
## $standardGeneric
## function (f, fdef)  .Primitive("standardGeneric")
##
## $`storage.mode<-`
## function (x, value)  .Primitive("storage.mode<-")
##
## $substitute
## function (expr, env)  .Primitive("substitute")
##
## $sum
## function (..., na.rm = FALSE)  .Primitive("sum")
##
## $switch
## function (EXPR, ...)  .Primitive("switch")
##
## $tan
## function (x)  .Primitive("tan")
##
## $tanh
## function (x)  .Primitive("tanh")
##
## $tanpi
## function (x)  .Primitive("tanpi")
##
## $tracemem
## function (x)  .Primitive("tracemem")
##
## $trigamma
## function (x)  .Primitive("trigamma")
##
## $trunc
## function (x, ...)  .Primitive("trunc")
##
## $unclass
## function (x)  .Primitive("unclass")
##
## $untracemem
## function (x)  .Primitive("untracemem")
##
```

```
## $UseMethod
## function (generic, object)  .Primitive("UseMethod")
##
## $`while`
## .Primitive("while")
##
## $xtfrm
## function (x)  .Primitive("xtfrm")
```

3. What are the three important components of a function?

The three important components are the body(), formals(), and environment().

4. When does printing a function not show what environment it was created in?

Printing a function does not show what environment it is a primitive function. It will return null.

Lexical Scoping

1. What does the following code return? Why? What does each of the three c's mean?

```
c <- 10
c(c = c)
```

```
##  c
## 10
```

The first c is being assigned to 10. In the second line, the first c is concatnating the values, then the inside is assigning value to the vector, recreating the value of the new c as the value of the first original c in line one as 10.

2. What are the four principles that govern how R looks for values?

The four principles are name masking, functions vs. variables, a fresh start, and dynamic lookup.

#.3 What does the following function return? Make a prediction before running the code yourself. The function will run through each line in order, first doing 10^2, then adding 1 to that value (100+1), then multiplying 101 by 2, getting 202.

```
f <- function(x) {
  f <- function(x) {
    f <- function(x) {
      x ^ 2
    }
    f(x) + 1
  }
```

```
  f(x) * 2
}
f(10)
```

```
## [1] 202
```

Function Arguments

1. Clarify the following list of odd function calls:

```
x <- sample(replace = TRUE, 20, x = c(1:10, NA))
y <- runif(min = 0, max = 1, 20)
cor(m = "k", y = y, u = "p", x = x)
```

```
## [1] 0.01740777
```

x is saying to take a sample of 20 values that includes 1 to 10 and the rest NAs. y is saying to run values from the sample that range from 0 to 1 and include 20 values totaly. For cor, m is referring to the method where "k" = "kendall", x and y are referring to the operators above, and u is referring to the use, where "p" equals "pairwise".

2. What does this function return? Why? Which principle does it illustrate?

```
f1 <- function(x = {y <- 1; 2}, y = 0) {
  x + y
}
f1()
```

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

The function returns three. This is because we first assign x to 2 and y to 1, which r will pull before it pulls 0.

3. What does this function return? Why? Which principle does it illustrate?

```
f2 <- function(x = z) {
  z <- 100
  x
}
f2()
```

```
## [1] 100
```

This returns 100, for a similar reason as above. r will first pull the function that has been assigned to z, rather using the emtpy value given to x. This is name masking.

Special Calls

1. Create a list of all the replacement functions found in the base package. Which ones are primitive functions?

```
require(plyr)
library(plyr)
objs <- mget(ls("package:base"), inherits = TRUE)
prims <- Filter(is.primitive, objs)

replacfunc <- lapply(funs, is.primitive)
names(replacfunc)
```

```
##    [1] "-"
##    [2] "-.Date"
##    [3] "-.POSIXt"
##    [4] ":"
##    [5] "::"
##    [6] ":::"
##    [7] "!"
##    [8] "!.hexmode"
##    [9] "!.octmode"
##   [10] "!="
##   [11] "("
##   [12] "["
##   [13] "[.AsIs"
##   [14] "[.data.frame"
##   [15] "[.Date"
##   [16] "[.difftime"
##   [17] "[.Dlist"
##   [18] "[.factor"
##   [19] "[.hexmode"
##   [20] "[.listof"
##   [21] "[.noquote"
##   [22] "[.numeric_version"
##   [23] "[.octmode"
##   [24] "[.POSIXct"
##   [25] "[.POSIXlt"
##   [26] "[.simple.list"
##   [27] "[.table"
##   [28] "[.warnings"
##   [29] "[["
##   [30] "[[.data.frame"
##   [31] "[[.Date"
##   [32] "[[.factor"
##   [33] "[[.numeric_version"
##   [34] "[[.POSIXct"
##   [35] "[[<-"
##   [36] "[[<-.data.frame"
##   [37] "[[<-.factor"
##   [38] "[[<-.numeric_version"
##   [39] "[<-"
```

```
##   [40] "[<-.data.frame"
##   [41] "[<-.Date"
##   [42] "[<-.factor"
##   [43] "[<-.numeric_version"
##   [44] "[<-.POSIXct"
##   [45] "[<-.POSIXlt"
##   [46] "{"
##   [47] "@"
##   [48] "@<-"
##   [49] "*"
##   [50] "*.difftime"
##   [51] "/"
##   [52] "/.difftime"
##   [53] "&"
##   [54] "&.hexmode"
##   [55] "&.octmode"
##   [56] "&&"
##   [57] "%*%"
##   [58] "%/%"
##   [59] "%%"
##   [60] "%in%"
##   [61] "%o%"
##   [62] "%x%"
##   [63] "^"
##   [64] "+"
##   [65] "+.Date"
##   [66] "+.POSIXt"
##   [67] "<"
##   [68] "<-"
##   [69] "<<-"
##   [70] "<="
##   [71] "="
##   [72] "=="
##   [73] ">"
##   [74] ">="
##   [75] "|"
##   [76] "|.hexmode"
##   [77] "|.octmode"
##   [78] "||"
##   [79] "~"
##   [80] "$"
##   [81] "$.data.frame"
##   [82] "$.DLLInfo"
##   [83] "$.package_version"
##   [84] "$<-"
##   [85] "$<-.data.frame"
##   [86] "abbreviate"
##   [87] "abs"
##   [88] "acos"
##   [89] "acosh"
##   [90] "addNA"
##   [91] "addTaskCallback"
##   [92] "agrep"
##   [93] "agrepl"
```

```
##   [94] "alist"
##   [95] "all"
##   [96] "all.equal"
##   [97] "all.equal.character"
##   [98] "all.equal.default"
##   [99] "all.equal.environment"
##  [100] "all.equal.envRefClass"
##  [101] "all.equal.factor"
##  [102] "all.equal.formula"
##  [103] "all.equal.language"
##  [104] "all.equal.list"
##  [105] "all.equal.numeric"
##  [106] "all.equal.POSIXt"
##  [107] "all.equal.raw"
##  [108] "all.names"
##  [109] "all.vars"
##  [110] "any"
##  [111] "anyDuplicated"
##  [112] "anyDuplicated.array"
##  [113] "anyDuplicated.data.frame"
##  [114] "anyDuplicated.default"
##  [115] "anyDuplicated.matrix"
##  [116] "anyNA"
##  [117] "anyNA.numeric_version"
##  [118] "anyNA.POSIXlt"
##  [119] "aperm"
##  [120] "aperm.default"
##  [121] "aperm.table"
##  [122] "append"
##  [123] "apply"
##  [124] "Arg"
##  [125] "args"
##  [126] "array"
##  [127] "arrayInd"
##  [128] "as.array"
##  [129] "as.array.default"
##  [130] "as.call"
##  [131] "as.character"
##  [132] "as.character.condition"
##  [133] "as.character.Date"
##  [134] "as.character.default"
##  [135] "as.character.error"
##  [136] "as.character.factor"
##  [137] "as.character.hexmode"
##  [138] "as.character.numeric_version"
##  [139] "as.character.octmode"
##  [140] "as.character.POSIXt"
##  [141] "as.character.srcref"
##  [142] "as.complex"
##  [143] "as.data.frame"
##  [144] "as.data.frame.array"
##  [145] "as.data.frame.AsIs"
##  [146] "as.data.frame.character"
##  [147] "as.data.frame.complex"
```

```
##  [148] "as.data.frame.data.frame"
##  [149] "as.data.frame.Date"
##  [150] "as.data.frame.default"
##  [151] "as.data.frame.difftime"
##  [152] "as.data.frame.factor"
##  [153] "as.data.frame.integer"
##  [154] "as.data.frame.list"
##  [155] "as.data.frame.logical"
##  [156] "as.data.frame.matrix"
##  [157] "as.data.frame.model.matrix"
##  [158] "as.data.frame.noquote"
##  [159] "as.data.frame.numeric"
##  [160] "as.data.frame.numeric_version"
##  [161] "as.data.frame.ordered"
##  [162] "as.data.frame.POSIXct"
##  [163] "as.data.frame.POSIXlt"
##  [164] "as.data.frame.raw"
##  [165] "as.data.frame.table"
##  [166] "as.data.frame.ts"
##  [167] "as.data.frame.vector"
##  [168] "as.Date"
##  [169] "as.Date.character"
##  [170] "as.Date.date"
##  [171] "as.Date.dates"
##  [172] "as.Date.default"
##  [173] "as.Date.factor"
##  [174] "as.Date.numeric"
##  [175] "as.Date.POSIXct"
##  [176] "as.Date.POSIXlt"
##  [177] "as.difftime"
##  [178] "as.double"
##  [179] "as.double.difftime"
##  [180] "as.double.POSIXlt"
##  [181] "as.environment"
##  [182] "as.expression"
##  [183] "as.expression.default"
##  [184] "as.factor"
##  [185] "as.function"
##  [186] "as.function.default"
##  [187] "as.hexmode"
##  [188] "as.integer"
##  [189] "as.list"
##  [190] "as.list.data.frame"
##  [191] "as.list.Date"
##  [192] "as.list.default"
##  [193] "as.list.environment"
##  [194] "as.list.factor"
##  [195] "as.list.function"
##  [196] "as.list.numeric_version"
##  [197] "as.list.POSIXct"
##  [198] "as.logical"
##  [199] "as.logical.factor"
##  [200] "as.matrix"
##  [201] "as.matrix.data.frame"
```

```
##  [202] "as.matrix.default"
##  [203] "as.matrix.noquote"
##  [204] "as.matrix.POSIXlt"
##  [205] "as.name"
##  [206] "as.null"
##  [207] "as.null.default"
##  [208] "as.numeric"
##  [209] "as.numeric_version"
##  [210] "as.octmode"
##  [211] "as.ordered"
##  [212] "as.package_version"
##  [213] "as.pairlist"
##  [214] "as.POSIXct"
##  [215] "as.POSIXct.date"
##  [216] "as.POSIXct.Date"
##  [217] "as.POSIXct.dates"
##  [218] "as.POSIXct.default"
##  [219] "as.POSIXct.numeric"
##  [220] "as.POSIXct.POSIXlt"
##  [221] "as.POSIXlt"
##  [222] "as.POSIXlt.character"
##  [223] "as.POSIXlt.date"
##  [224] "as.POSIXlt.Date"
##  [225] "as.POSIXlt.dates"
##  [226] "as.POSIXlt.default"
##  [227] "as.POSIXlt.factor"
##  [228] "as.POSIXlt.numeric"
##  [229] "as.POSIXlt.POSIXct"
##  [230] "as.qr"
##  [231] "as.raw"
##  [232] "as.single"
##  [233] "as.single.default"
##  [234] "as.symbol"
##  [235] "as.table"
##  [236] "as.table.default"
##  [237] "as.vector"
##  [238] "as.vector.factor"
##  [239] "asin"
##  [240] "asinh"
##  [241] "asNamespace"
##  [242] "asS3"
##  [243] "asS4"
##  [244] "assign"
##  [245] "atan"
##  [246] "atan2"
##  [247] "atanh"
##  [248] "attach"
##  [249] "attachNamespace"
##  [250] "attr"
##  [251] "attr.all.equal"
##  [252] "attr<-"
##  [253] "attributes"
##  [254] "attributes<-"
##  [255] "autoload"
```

```
##  [256] "autoloader"
##  [257] "backsolve"
##  [258] "baseenv"
##  [259] "basename"
##  [260] "besselI"
##  [261] "besselJ"
##  [262] "besselK"
##  [263] "besselY"
##  [264] "beta"
##  [265] "bindingIsActive"
##  [266] "bindingIsLocked"
##  [267] "bindtextdomain"
##  [268] "bitwAnd"
##  [269] "bitwNot"
##  [270] "bitwOr"
##  [271] "bitwShiftL"
##  [272] "bitwShiftR"
##  [273] "bitwXor"
##  [274] "body"
##  [275] "body<-"
##  [276] "bquote"
##  [277] "break"
##  [278] "browser"
##  [279] "browserCondition"
##  [280] "browserSetDebug"
##  [281] "browserText"
##  [282] "builtins"
##  [283] "by"
##  [284] "by.data.frame"
##  [285] "by.default"
##  [286] "bzfile"
##  [287] "c"
##  [288] "c.Date"
##  [289] "c.difftime"
##  [290] "c.noquote"
##  [291] "c.numeric_version"
##  [292] "c.POSIXct"
##  [293] "c.POSIXlt"
##  [294] "c.warnings"
##  [295] "call"
##  [296] "callCC"
##  [297] "capabilities"
##  [298] "casefold"
##  [299] "cat"
##  [300] "cbind"
##  [301] "cbind.data.frame"
##  [302] "ceiling"
##  [303] "char.expand"
##  [304] "character"
##  [305] "charmatch"
##  [306] "charToRaw"
##  [307] "chartr"
##  [308] "check_tzones"
##  [309] "chkDots"
```

```
##  [310] "chol"
##  [311] "chol.default"
##  [312] "chol2inv"
##  [313] "choose"
##  [314] "class"
##  [315] "class<-"
##  [316] "clearPushBack"
##  [317] "close"
##  [318] "close.connection"
##  [319] "close.srcfile"
##  [320] "close.srcfilealias"
##  [321] "closeAllConnections"
##  [322] "col"
##  [323] "colMeans"
##  [324] "colnames"
##  [325] "colnames<-"
##  [326] "colSums"
##  [327] "commandArgs"
##  [328] "comment"
##  [329] "comment<-"
##  [330] "complex"
##  [331] "computeRestarts"
##  [332] "conditionCall"
##  [333] "conditionCall.condition"
##  [334] "conditionMessage"
##  [335] "conditionMessage.condition"
##  [336] "conflicts"
##  [337] "Conj"
##  [338] "contributors"
##  [339] "cos"
##  [340] "cosh"
##  [341] "cospi"
##  [342] "crossprod"
##  [343] "Cstack_info"
##  [344] "cummax"
##  [345] "cummin"
##  [346] "cumprod"
##  [347] "cumsum"
##  [348] "curlGetHeaders"
##  [349] "cut"
##  [350] "cut.Date"
##  [351] "cut.default"
##  [352] "cut.POSIXt"
##  [353] "data.class"
##  [354] "data.frame"
##  [355] "data.matrix"
##  [356] "date"
##  [357] "debug"
##  [358] "debuggingState"
##  [359] "debugonce"
##  [360] "default.stringsAsFactors"
##  [361] "delayedAssign"
##  [362] "deparse"
##  [363] "det"
```

```
##  [364] "detach"
##  [365] "determinant"
##  [366] "determinant.matrix"
##  [367] "dget"
##  [368] "diag"
##  [369] "diag<-"
##  [370] "diff"
##  [371] "diff.Date"
##  [372] "diff.default"
##  [373] "diff.difftime"
##  [374] "diff.POSIXt"
##  [375] "difftime"
##  [376] "digamma"
##  [377] "dim"
##  [378] "dim.data.frame"
##  [379] "dim<-"
##  [380] "dimnames"
##  [381] "dimnames.data.frame"
##  [382] "dimnames<-"
##  [383] "dimnames<-.data.frame"
##  [384] "dir"
##  [385] "dir.create"
##  [386] "dir.exists"
##  [387] "dirname"
##  [388] "do.call"
##  [389] "dontCheck"
##  [390] "double"
##  [391] "dput"
##  [392] "dQuote"
##  [393] "drop"
##  [394] "droplevels"
##  [395] "droplevels.data.frame"
##  [396] "droplevels.factor"
##  [397] "dump"
##  [398] "duplicated"
##  [399] "duplicated.array"
##  [400] "duplicated.data.frame"
##  [401] "duplicated.default"
##  [402] "duplicated.matrix"
##  [403] "duplicated.numeric_version"
##  [404] "duplicated.POSIXlt"
##  [405] "duplicated.warnings"
##  [406] "dyn.load"
##  [407] "dyn.unload"
##  [408] "dynGet"
##  [409] "eapply"
##  [410] "eigen"
##  [411] "emptyenv"
##  [412] "enc2native"
##  [413] "enc2utf8"
##  [414] "encodeString"
##  [415] "Encoding"
##  [416] "Encoding<-"
##  [417] "endsWith"
```

```
##  [418] "enquote"
##  [419] "env.profile"
##  [420] "environment"
##  [421] "environment<-"
##  [422] "environmentIsLocked"
##  [423] "environmentName"
##  [424] "eval"
##  [425] "eval.parent"
##  [426] "evalq"
##  [427] "exists"
##  [428] "exp"
##  [429] "expand.grid"
##  [430] "expm1"
##  [431] "expression"
##  [432] "extSoftVersion"
##  [433] "factor"
##  [434] "factorial"
##  [435] "fifo"
##  [436] "file"
##  [437] "file.access"
##  [438] "file.append"
##  [439] "file.choose"
##  [440] "file.copy"
##  [441] "file.create"
##  [442] "file.exists"
##  [443] "file.info"
##  [444] "file.link"
##  [445] "file.mode"
##  [446] "file.mtime"
##  [447] "file.path"
##  [448] "file.remove"
##  [449] "file.rename"
##  [450] "file.show"
##  [451] "file.size"
##  [452] "file.symlink"
##  [453] "Filter"
##  [454] "Find"
##  [455] "find.package"
##  [456] "findInterval"
##  [457] "findPackageEnv"
##  [458] "findRestart"
##  [459] "floor"
##  [460] "flush"
##  [461] "flush.connection"
##  [462] "for"
##  [463] "force"
##  [464] "forceAndCall"
##  [465] "formals"
##  [466] "formals<-"
##  [467] "format"
##  [468] "format.AsIs"
##  [469] "format.data.frame"
##  [470] "format.Date"
##  [471] "format.default"
```

```
##  [472] "format.difftime"
##  [473] "format.factor"
##  [474] "format.hexmode"
##  [475] "format.info"
##  [476] "format.libraryIQR"
##  [477] "format.numeric_version"
##  [478] "format.octmode"
##  [479] "format.packageInfo"
##  [480] "format.POSIXct"
##  [481] "format.POSIXlt"
##  [482] "format.pval"
##  [483] "format.summaryDefault"
##  [484] "formatC"
##  [485] "formatDL"
##  [486] "forwardsolve"
##  [487] "function"
##  [488] "gamma"
##  [489] "gc"
##  [490] "gc.time"
##  [491] "gcinfo"
##  [492] "gctorture"
##  [493] "gctorture2"
##  [494] "get"
##  [495] "get0"
##  [496] "getAllConnections"
##  [497] "getCallingDLL"
##  [498] "getCallingDLLe"
##  [499] "getConnection"
##  [500] "getDLLRegisteredRoutines"
##  [501] "getDLLRegisteredRoutines.character"
##  [502] "getDLLRegisteredRoutines.DLLInfo"
##  [503] "getElement"
##  [504] "geterrmessage"
##  [505] "getExportedValue"
##  [506] "getHook"
##  [507] "getLoadedDLLs"
##  [508] "getNamespace"
##  [509] "getNamespaceExports"
##  [510] "getNamespaceImports"
##  [511] "getNamespaceInfo"
##  [512] "getNamespaceName"
##  [513] "getNamespaceUsers"
##  [514] "getNamespaceVersion"
##  [515] "getNativeSymbolInfo"
##  [516] "getOption"
##  [517] "getRversion"
##  [518] "getSrcLines"
##  [519] "getTaskCallbackNames"
##  [520] "gettext"
##  [521] "gettextf"
##  [522] "getwd"
##  [523] "gl"
##  [524] "globalenv"
##  [525] "gregexpr"
```

```
##  [526] "grep"
##  [527] "grepl"
##  [528] "grepRaw"
##  [529] "grouping"
##  [530] "gsub"
##  [531] "gzcon"
##  [532] "gzfile"
##  [533] "I"
##  [534] "iconv"
##  [535] "iconvlist"
##  [536] "icuGetCollate"
##  [537] "icuSetCollate"
##  [538] "identical"
##  [539] "identity"
##  [540] "if"
##  [541] "ifelse"
##  [542] "Im"
##  [543] "importIntoEnv"
##  [544] "inherits"
##  [545] "integer"
##  [546] "interaction"
##  [547] "interactive"
##  [548] "intersect"
##  [549] "intToBits"
##  [550] "intToUtf8"
##  [551] "inverse.rle"
##  [552] "invisible"
##  [553] "invokeRestart"
##  [554] "invokeRestartInteractively"
##  [555] "is.array"
##  [556] "is.atomic"
##  [557] "is.call"
##  [558] "is.character"
##  [559] "is.complex"
##  [560] "is.data.frame"
##  [561] "is.double"
##  [562] "is.element"
##  [563] "is.environment"
##  [564] "is.expression"
##  [565] "is.factor"
##  [566] "is.finite"
##  [567] "is.function"
##  [568] "is.infinite"
##  [569] "is.integer"
##  [570] "is.language"
##  [571] "is.list"
##  [572] "is.loaded"
##  [573] "is.logical"
##  [574] "is.matrix"
##  [575] "is.na"
##  [576] "is.na.data.frame"
##  [577] "is.na.numeric_version"
##  [578] "is.na.POSIXlt"
##  [579] "is.na<-"
```

```
##  [580] "is.na<-.default"
##  [581] "is.na<-.factor"
##  [582] "is.na<-.numeric_version"
##  [583] "is.name"
##  [584] "is.nan"
##  [585] "is.null"
##  [586] "is.numeric"
##  [587] "is.numeric_version"
##  [588] "is.numeric.Date"
##  [589] "is.numeric.difftime"
##  [590] "is.numeric.POSIXt"
##  [591] "is.object"
##  [592] "is.ordered"
##  [593] "is.package_version"
##  [594] "is.pairlist"
##  [595] "is.primitive"
##  [596] "is.qr"
##  [597] "is.R"
##  [598] "is.raw"
##  [599] "is.recursive"
##  [600] "is.single"
##  [601] "is.symbol"
##  [602] "is.table"
##  [603] "is.unsorted"
##  [604] "is.vector"
##  [605] "isatty"
##  [606] "isBaseNamespace"
##  [607] "isdebugged"
##  [608] "isIncomplete"
##  [609] "isNamespace"
##  [610] "isNamespaceLoaded"
##  [611] "ISOdate"
##  [612] "ISOdatetime"
##  [613] "isOpen"
##  [614] "isRestart"
##  [615] "isS4"
##  [616] "isSeekable"
##  [617] "isSymmetric"
##  [618] "isSymmetric.matrix"
##  [619] "isTRUE"
##  [620] "jitter"
##  [621] "julian"
##  [622] "julian.Date"
##  [623] "julian.POSIXt"
##  [624] "kappa"
##  [625] "kappa.default"
##  [626] "kappa.lm"
##  [627] "kappa.qr"
##  [628] "kronecker"
##  [629] "l10n_info"
##  [630] "La_version"
##  [631] "La.svd"
##  [632] "labels"
##  [633] "labels.default"
```

```
##  [634] "lapply"
##  [635] "lazyLoad"
##  [636] "lazyLoadDBexec"
##  [637] "lazyLoadDBfetch"
##  [638] "lbeta"
##  [639] "lchoose"
##  [640] "length"
##  [641] "length.POSIXlt"
##  [642] "length<-"
##  [643] "length<-.factor"
##  [644] "lengths"
##  [645] "levels"
##  [646] "levels.default"
##  [647] "levels<-"
##  [648] "levels<-.factor"
##  [649] "lfactorial"
##  [650] "lgamma"
##  [651] "libcurlVersion"
##  [652] "library"
##  [653] "library.dynam"
##  [654] "library.dynam.unload"
##  [655] "licence"
##  [656] "license"
##  [657] "list"
##  [658] "list.dirs"
##  [659] "list.files"
##  [660] "list2env"
##  [661] "load"
##  [662] "loadedNamespaces"
##  [663] "loadingNamespaceInfo"
##  [664] "loadNamespace"
##  [665] "local"
##  [666] "lockBinding"
##  [667] "lockEnvironment"
##  [668] "log"
##  [669] "log10"
##  [670] "log1p"
##  [671] "log2"
##  [672] "logb"
##  [673] "logical"
##  [674] "lower.tri"
##  [675] "ls"
##  [676] "make.names"
##  [677] "make.unique"
##  [678] "makeActiveBinding"
##  [679] "Map"
##  [680] "mapply"
##  [681] "margin.table"
##  [682] "mat.or.vec"
##  [683] "match"
##  [684] "match.arg"
##  [685] "match.call"
##  [686] "match.fun"
##  [687] "Math.data.frame"
```

```
##  [688] "Math.Date"
##  [689] "Math.difftime"
##  [690] "Math.factor"
##  [691] "Math.POSIXt"
##  [692] "matrix"
##  [693] "max"
##  [694] "max.col"
##  [695] "mean"
##  [696] "mean.Date"
##  [697] "mean.default"
##  [698] "mean.difftime"
##  [699] "mean.POSIXct"
##  [700] "mean.POSIXlt"
##  [701] "mem.limits"
##  [702] "memCompress"
##  [703] "memDecompress"
##  [704] "memory.profile"
##  [705] "merge"
##  [706] "merge.data.frame"
##  [707] "merge.default"
##  [708] "message"
##  [709] "mget"
##  [710] "min"
##  [711] "missing"
##  [712] "Mod"
##  [713] "mode"
##  [714] "mode<-"
##  [715] "months"
##  [716] "months.Date"
##  [717] "months.POSIXt"
##  [718] "mostattributes<-"
##  [719] "names"
##  [720] "names.POSIXlt"
##  [721] "names<-"
##  [722] "names<-.POSIXlt"
##  [723] "namespaceExport"
##  [724] "namespaceImport"
##  [725] "namespaceImportClasses"
##  [726] "namespaceImportFrom"
##  [727] "namespaceImportMethods"
##  [728] "nargs"
##  [729] "nchar"
##  [730] "ncol"
##  [731] "NCOL"
##  [732] "Negate"
##  [733] "new.env"
##  [734] "next"
##  [735] "NextMethod"
##  [736] "ngettext"
##  [737] "nlevels"
##  [738] "noquote"
##  [739] "norm"
##  [740] "normalizePath"
##  [741] "nrow"
```

```
##  [742] "NROW"
##  [743] "numeric"
##  [744] "numeric_version"
##  [745] "nzchar"
##  [746] "objects"
##  [747] "oldClass"
##  [748] "oldClass<-"
##  [749] "OlsonNames"
##  [750] "on.exit"
##  [751] "open"
##  [752] "open.connection"
##  [753] "open.srcfile"
##  [754] "open.srcfilealias"
##  [755] "open.srcfilecopy"
##  [756] "Ops.data.frame"
##  [757] "Ops.Date"
##  [758] "Ops.difftime"
##  [759] "Ops.factor"
##  [760] "Ops.numeric_version"
##  [761] "Ops.ordered"
##  [762] "Ops.POSIXt"
##  [763] "options"
##  [764] "order"
##  [765] "ordered"
##  [766] "outer"
##  [767] "package_version"
##  [768] "packageEvent"
##  [769] "packageHasNamespace"
##  [770] "packageStartupMessage"
##  [771] "packBits"
##  [772] "pairlist"
##  [773] "parent.env"
##  [774] "parent.env<-"
##  [775] "parent.frame"
##  [776] "parse"
##  [777] "parseNamespaceFile"
##  [778] "paste"
##  [779] "paste0"
##  [780] "path.expand"
##  [781] "path.package"
##  [782] "pcre_config"
##  [783] "pipe"
##  [784] "pmatch"
##  [785] "pmax"
##  [786] "pmax.int"
##  [787] "pmin"
##  [788] "pmin.int"
##  [789] "polyroot"
##  [790] "pos.to.env"
##  [791] "Position"
##  [792] "pretty"
##  [793] "pretty.default"
##  [794] "prettyNum"
##  [795] "print"
```

```
##  [796] "print.AsIs"
##  [797] "print.by"
##  [798] "print.condition"
##  [799] "print.connection"
##  [800] "print.data.frame"
##  [801] "print.Date"
##  [802] "print.default"
##  [803] "print.difftime"
##  [804] "print.Dlist"
##  [805] "print.DLLInfo"
##  [806] "print.DLLInfoList"
##  [807] "print.DLLRegisteredRoutines"
##  [808] "print.factor"
##  [809] "print.function"
##  [810] "print.hexmode"
##  [811] "print.libraryIQR"
##  [812] "print.listof"
##  [813] "print.NativeRoutineList"
##  [814] "print.noquote"
##  [815] "print.numeric_version"
##  [816] "print.octmode"
##  [817] "print.packageInfo"
##  [818] "print.POSIXct"
##  [819] "print.POSIXlt"
##  [820] "print.proc_time"
##  [821] "print.restart"
##  [822] "print.rle"
##  [823] "print.simple.list"
##  [824] "print.srcfile"
##  [825] "print.srcref"
##  [826] "print.summary.table"
##  [827] "print.summaryDefault"
##  [828] "print.table"
##  [829] "print.warnings"
##  [830] "prmatrix"
##  [831] "proc.time"
##  [832] "prod"
##  [833] "prop.table"
##  [834] "provideDimnames"
##  [835] "psigamma"
##  [836] "pushBack"
##  [837] "pushBackLength"
##  [838] "q"
##  [839] "qr"
##  [840] "qr.coef"
##  [841] "qr.default"
##  [842] "qr.fitted"
##  [843] "qr.Q"
##  [844] "qr.qty"
##  [845] "qr.qy"
##  [846] "qr.R"
##  [847] "qr.resid"
##  [848] "qr.solve"
##  [849] "qr.X"
```

```
##  [850] "quarters"
##  [851] "quarters.Date"
##  [852] "quarters.POSIXt"
##  [853] "quit"
##  [854] "quote"
##  [855] "R_system_version"
##  [856] "R.home"
##  [857] "R.Version"
##  [858] "range"
##  [859] "range.default"
##  [860] "rank"
##  [861] "rapply"
##  [862] "raw"
##  [863] "rawConnection"
##  [864] "rawConnectionValue"
##  [865] "rawShift"
##  [866] "rawToBits"
##  [867] "rawToChar"
##  [868] "rbind"
##  [869] "rbind.data.frame"
##  [870] "rcond"
##  [871] "Re"
##  [872] "read.dcf"
##  [873] "readBin"
##  [874] "readChar"
##  [875] "readline"
##  [876] "readLines"
##  [877] "readRDS"
##  [878] "readRenviron"
##  [879] "Recall"
##  [880] "Reduce"
##  [881] "reg.finalizer"
##  [882] "regexec"
##  [883] "regexpr"
##  [884] "registerS3method"
##  [885] "registerS3methods"
##  [886] "regmatches"
##  [887] "regmatches<-"
##  [888] "remove"
##  [889] "removeTaskCallback"
##  [890] "rep"
##  [891] "rep_len"
##  [892] "rep.Date"
##  [893] "rep.factor"
##  [894] "rep.int"
##  [895] "rep.numeric_version"
##  [896] "rep.POSIXct"
##  [897] "rep.POSIXlt"
##  [898] "repeat"
##  [899] "replace"
##  [900] "replicate"
##  [901] "require"
##  [902] "requireNamespace"
##  [903] "restartDescription"
```

```
##  [904] "restartFormals"
##  [905] "retracemem"
##  [906] "return"
##  [907] "returnValue"
##  [908] "rev"
##  [909] "rev.default"
##  [910] "rle"
##  [911] "rm"
##  [912] "RNGkind"
##  [913] "RNGversion"
##  [914] "round"
##  [915] "round.Date"
##  [916] "round.POSIXt"
##  [917] "row"
##  [918] "row.names"
##  [919] "row.names.data.frame"
##  [920] "row.names.default"
##  [921] "row.names<-"
##  [922] "row.names<-.data.frame"
##  [923] "row.names<-.default"
##  [924] "rowMeans"
##  [925] "rownames"
##  [926] "rownames<-"
##  [927] "rowsum"
##  [928] "rowsum.data.frame"
##  [929] "rowsum.default"
##  [930] "rowSums"
##  [931] "sample"
##  [932] "sample.int"
##  [933] "sapply"
##  [934] "save"
##  [935] "save.image"
##  [936] "saveRDS"
##  [937] "scale"
##  [938] "scale.default"
##  [939] "scan"
##  [940] "search"
##  [941] "searchpaths"
##  [942] "seek"
##  [943] "seek.connection"
##  [944] "seq"
##  [945] "seq_along"
##  [946] "seq_len"
##  [947] "seq.Date"
##  [948] "seq.default"
##  [949] "seq.int"
##  [950] "seq.POSIXt"
##  [951] "sequence"
##  [952] "serialize"
##  [953] "set.seed"
##  [954] "setdiff"
##  [955] "setequal"
##  [956] "setHook"
##  [957] "setNamespaceInfo"
```

```
##  [958] "setSessionTimeLimit"
##  [959] "setTimeLimit"
##  [960] "setwd"
##  [961] "showConnections"
##  [962] "shQuote"
##  [963] "sign"
##  [964] "signalCondition"
##  [965] "signif"
##  [966] "simpleCondition"
##  [967] "simpleError"
##  [968] "simpleMessage"
##  [969] "simpleWarning"
##  [970] "simplify2array"
##  [971] "sin"
##  [972] "single"
##  [973] "sinh"
##  [974] "sink"
##  [975] "sink.number"
##  [976] "sinpi"
##  [977] "slice.index"
##  [978] "socketConnection"
##  [979] "socketSelect"
##  [980] "solve"
##  [981] "solve.default"
##  [982] "solve.qr"
##  [983] "sort"
##  [984] "sort.default"
##  [985] "sort.int"
##  [986] "sort.list"
##  [987] "sort.POSIXlt"
##  [988] "source"
##  [989] "split"
##  [990] "split.data.frame"
##  [991] "split.Date"
##  [992] "split.default"
##  [993] "split.POSIXct"
##  [994] "split<-"
##  [995] "split<-.data.frame"
##  [996] "split<-.default"
##  [997] "sprintf"
##  [998] "sqrt"
##  [999] "sQuote"
## [1000] "srcfile"
## [1001] "srcfilealias"
## [1002] "srcfilecopy"
## [1003] "srcref"
## [1004] "standardGeneric"
## [1005] "startsWith"
## [1006] "stderr"
## [1007] "stdin"
## [1008] "stdout"
## [1009] "stop"
## [1010] "stopifnot"
## [1011] "storage.mode"
```

```
## [1012] "storage.mode<-"
## [1013] "strftime"
## [1014] "strptime"
## [1015] "strrep"
## [1016] "strsplit"
## [1017] "strtoi"
## [1018] "strtrim"
## [1019] "structure"
## [1020] "strwrap"
## [1021] "sub"
## [1022] "subset"
## [1023] "subset.data.frame"
## [1024] "subset.default"
## [1025] "subset.matrix"
## [1026] "substitute"
## [1027] "substr"
## [1028] "substr<-"
## [1029] "substring"
## [1030] "substring<-"
## [1031] "sum"
## [1032] "summary"
## [1033] "summary.connection"
## [1034] "summary.data.frame"
## [1035] "Summary.data.frame"
## [1036] "summary.Date"
## [1037] "Summary.Date"
## [1038] "summary.default"
## [1039] "Summary.difftime"
## [1040] "summary.factor"
## [1041] "Summary.factor"
## [1042] "summary.matrix"
## [1043] "Summary.numeric_version"
## [1044] "Summary.ordered"
## [1045] "summary.POSIXct"
## [1046] "Summary.POSIXct"
## [1047] "summary.POSIXlt"
## [1048] "Summary.POSIXlt"
## [1049] "summary.proc_time"
## [1050] "summary.srcfile"
## [1051] "summary.srcref"
## [1052] "summary.table"
## [1053] "suppressMessages"
## [1054] "suppressPackageStartupMessages"
## [1055] "suppressWarnings"
## [1056] "svd"
## [1057] "sweep"
## [1058] "switch"
## [1059] "sys.call"
## [1060] "sys.calls"
## [1061] "Sys.chmod"
## [1062] "Sys.Date"
## [1063] "sys.frame"
## [1064] "sys.frames"
## [1065] "sys.function"
```

```
## [1066] "Sys.getenv"
## [1067] "Sys.getlocale"
## [1068] "Sys.getpid"
## [1069] "Sys.glob"
## [1070] "Sys.info"
## [1071] "sys.load.image"
## [1072] "Sys.localeconv"
## [1073] "sys.nframe"
## [1074] "sys.on.exit"
## [1075] "sys.parent"
## [1076] "sys.parents"
## [1077] "Sys.readlink"
## [1078] "sys.save.image"
## [1079] "Sys.setenv"
## [1080] "Sys.setFileTime"
## [1081] "Sys.setlocale"
## [1082] "Sys.sleep"
## [1083] "sys.source"
## [1084] "sys.status"
## [1085] "Sys.time"
## [1086] "Sys.timezone"
## [1087] "Sys.umask"
## [1088] "Sys.unsetenv"
## [1089] "Sys.which"
## [1090] "system"
## [1091] "system.file"
## [1092] "system.time"
## [1093] "system2"
## [1094] "t"
## [1095] "t.data.frame"
## [1096] "t.default"
## [1097] "table"
## [1098] "tabulate"
## [1099] "tan"
## [1100] "tanh"
## [1101] "tanpi"
## [1102] "tapply"
## [1103] "taskCallbackManager"
## [1104] "tcrossprod"
## [1105] "tempdir"
## [1106] "tempfile"
## [1107] "testPlatformEquivalence"
## [1108] "textConnection"
## [1109] "textConnectionValue"
## [1110] "tolower"
## [1111] "topenv"
## [1112] "toString"
## [1113] "toString.default"
## [1114] "toupper"
## [1115] "trace"
## [1116] "traceback"
## [1117] "tracemem"
## [1118] "tracingState"
## [1119] "transform"
```

```
## [1120] "transform.data.frame"
## [1121] "transform.default"
## [1122] "trigamma"
## [1123] "trimws"
## [1124] "trunc"
## [1125] "trunc.Date"
## [1126] "trunc.POSIXt"
## [1127] "truncate"
## [1128] "truncate.connection"
## [1129] "try"
## [1130] "tryCatch"
## [1131] "typeof"
## [1132] "unclass"
## [1133] "undebug"
## [1134] "union"
## [1135] "unique"
## [1136] "unique.array"
## [1137] "unique.data.frame"
## [1138] "unique.default"
## [1139] "unique.matrix"
## [1140] "unique.numeric_version"
## [1141] "unique.POSIXlt"
## [1142] "unique.warnings"
## [1143] "units"
## [1144] "units.difftime"
## [1145] "units<-"
## [1146] "units<-.difftime"
## [1147] "unix.time"
## [1148] "unlink"
## [1149] "unlist"
## [1150] "unloadNamespace"
## [1151] "unlockBinding"
## [1152] "unname"
## [1153] "unserialize"
## [1154] "unsplit"
## [1155] "untrace"
## [1156] "untracemem"
## [1157] "unz"
## [1158] "upper.tri"
## [1159] "url"
## [1160] "UseMethod"
## [1161] "utf8ToInt"
## [1162] "validEnc"
## [1163] "validUTF8"
## [1164] "vapply"
## [1165] "vector"
## [1166] "Vectorize"
## [1167] "warning"
## [1168] "warnings"
## [1169] "weekdays"
## [1170] "weekdays.Date"
## [1171] "weekdays.POSIXt"
## [1172] "which"
## [1173] "which.max"
```

```
## [1174] "which.min"
## [1175] "while"
## [1176] "with"
## [1177] "with.default"
## [1178] "withCallingHandlers"
## [1179] "within"
## [1180] "within.data.frame"
## [1181] "within.list"
## [1182] "withRestarts"
## [1183] "withVisible"
## [1184] "write"
## [1185] "write.dcf"
## [1186] "writeBin"
## [1187] "writeChar"
## [1188] "writeLines"
## [1189] "xor"
## [1190] "xor.hexmode"
## [1191] "xor.octmode"
## [1192] "xpdrows.data.frame"
## [1193] "xtfrm"
## [1194] "xtfrm.AsIs"
## [1195] "xtfrm.Date"
## [1196] "xtfrm.default"
## [1197] "xtfrm.difftime"
## [1198] "xtfrm.factor"
## [1199] "xtfrm.numeric_version"
## [1200] "xtfrm.POSIXct"
## [1201] "xtfrm.POSIXlt"
## [1202] "xtfrm.Surv"
## [1203] "xzfile"
## [1204] "zapsmall"
```

2. What are valid names for user-created infix functions?

You use %% to create infix functions. Anything surrounded by %% will be an infix function.

3. Create an infix xor() operator.

```
`%xor%` <- function(x,y){
  (x|y)
}

x %xor% y
```

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

4. Create infix versions of the set functions intersect(), union(), and setdiff().

```
x1 <- seq(1:10)
y1 <- seq(5:15)


`%intersect%` <- function(x1, y1)
{
  intersect(x1, y1)
}
x1 %intersect% y1
```

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

```
`%union%` <- function(x1, y1)
{
  union(x, y)
}
x1 %union% y1
```

```
##  [1]  3.00000000  4.00000000          NA  5.00000000  7.00000000
##  [6]  2.00000000 10.00000000  9.00000000  8.00000000  0.44111331
## [11]  0.98774387  0.93423053  0.67208203  0.36052373  0.52471441
## [16]  0.60802662  0.19523074  0.63037713  0.08207641  0.89238004
## [21]  0.90457966  0.84889546  0.44403310  0.94307838  0.93634955
## [26]  0.83448290  0.45736176  0.14605886  0.24691841
```

```
`%setdiff%` <- function(x1, y1)
{
  setdiff(x1, y1)
}
x1 %setdiff% y1
```

```
## integer(0)
```

5. Create a replacement function that modifies a random location in a vector.

```
x <- seq(1:10)
`randomsample` <- function(x, value) {
  xlength <- 1:length(x)
  place <- sample(xlength,1)
  x[place] <-
  print(x)
}
randomsample(x)
```

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

```
## Warning in x[place] <- print(x): number of items to replace is not a
## multiple of replacement length
```

Return Calls

1. How does the chdir parameter of source() compare to in_dir()? Why might you prefer one approach to the other?

They both allow you to edit the working directory by pulling out a specific source but not changing the working directory for the rest of your script.

2. What function undoes the action of library()? How do you save and restore the values of options() and par()?

You can use detach() to undue the library actions. Options() and par() both allow you to access the global environment. They work simpliarly to setwd().

3. Write a function that opens a graphics device, runs the supplied code, and closes the graphics device (always, regardless of whether or not the plotting code worked).

4. We can use on.exit() to implement a simple version of capture.output().

```
capture.output2 <- function(code) {
  temp <- tempfile()
  on.exit(file.remove(temp), add = TRUE)

  sink(temp)
  on.exit(sink(), add = TRUE)

  force(code)
  readLines(temp)
}
capture.output2(cat("a", "b", "c", sep = "\n"))
```

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

```
body(capture.output)
```

```
## {
##     args <- substitute(list(...))[-1L]
##     type <- match.arg(type)
##     rval <- NULL
##     closeit <- TRUE
##     if (is.null(file))
##         file <- textConnection("rval", "w", local = TRUE)
##     else if (is.character(file))
##         file <- file(file, if (append)
##             "a"
```

```
##         else "w")
##     else if (inherits(file, "connection")) {
##         if (!isOpen(file))
##             open(file, if (append)
##                 "a"
##             else "w")
##         else closeit <- FALSE
##     }
##     else stop("'file' must be NULL, a character string or a connection")
##     sink(file, type = type, split = split)
##     on.exit({
##         sink(type = type, split = split)
##         if (closeit) close(file)
##     })
##     pf <- parent.frame()
##     evalVis <- function(expr) withVisible(eval(expr, pf))
##     for (i in seq_along(args)) {
##         expr <- args[[i]]
##         tmp <- switch(mode(expr), expression = lapply(expr, evalVis),
##             call = , name = list(evalVis(expr)), stop("bad argument"))
##         for (item in tmp) if (item$visible)
##             print(item$value)
##     }
##     on.exit()
##     sink(type = type, split = split)
##     if (closeit)
##         close(file)
##     if (is.null(rval))
##         invisible(NULL)
##     else rval
## }
```

```
body(capture.output2)
```

```
## {
##     temp <- tempfile()
##     on.exit(file.remove(temp), add = TRUE)
##     sink(temp)
##     on.exit(sink(), add = TRUE)
##     force(code)
##     readLines(temp)
## }
```

Compare capture.output() to capture.output2(). How do the functions differ? What features have I removed to make the key ideas easier to see? How have I rewritten the key ideas to be easier to understand?

From looking at the body of the functions, it is clear that capture.output2() is much shorter than capture.output() is. It removes many of the if else and for loops that the function was going through.