R Scripting

Exercises for unit 4 - Functions, OO concepts

Marcus Wurzer

06 10 2022

Please solve the following problems!

1. Try to answer quiz questions 1. to 5. at the beginning of the "Functions" chapter of the coursebook: http://adv-r.had.co.nz/Functions.html
2. Try to answer quiz questions 1. to 4. at the beginning of the "OO field guide" chapter of the coursebook: http://adv-r.had.co.nz/OO-essentials.html
3. Coursebook exercises:

   a. Implement `arg_max()`. It should take a function and a vector of inputs, and return the elements of the input where the function returns the highest value. For example, `arg_max(-10:5, function(x) x ^ 2)` should return -10. `arg_max(-5:5, function(x) x ^ 2)` should return `c(-5, 5)`. Also implement the matching `arg_min()` function.

   ```
   arg_max <- function(x, f) {
     x[f(x) == max(f(x))]
   }

   arg_min <- function(x, f) {
     x[f(x) == min(f(x))]
   }

   arg_max(-10:5, function(x) x ^ 2)
   ```

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

   ```
   arg_max(-5:5, function(x) x ^ 2)
   ```

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

   b. 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?

   You can test objects with `is.function` and `is.primitive`.

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

   `body()`, `formals()` and `environment()`.

   There is one exception to the rule that functions have three components. Primitive functions, like `sum()`, call C code directly with `.Primitive()` and contain no R code. Therefore their `formals()`, `body()`, and `environment()` are all NULL.

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

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

A named vector `c`, which first field has the value 10 and the name "c". The first "c" is the `c()` function, the second is the name of the first entry and the third is the value of the first entry.

e. Clarify the following list of odd function calls:

```
x <- sample(replace = TRUE, 20, x = c(1:10, NA))
# -> sample(x = c(1:10, NA), size = 20, replace = TRUE)
y <- runif(min = 0, max = 1, 20)
# -> runif(n = 20, min = 0, max = 1)
cor(m = "k", y = y, u = "p", x = x)
# -> cor(x = x, y = y, use = "pairwise.complete.obs", method = "pearson")
```

f. What classes have a method for the `Math` group generic in base R?

```
methods("Math")
```

```
## [1] Math,nonStructure-method Math,structure-method    Math.data.frame
## [4] Math.Date                Math.difftime            Math.factor
## [7] Math.POSIXt              Math.quosure*
## see '?methods' for accessing help and source code
```

4. Verify that the objects `cos`, `median`, and `read.table` are all functions.

```
str(cos)
```

```
## function (x)
```

```
str(median)
```

```
## function (x, na.rm = FALSE, ...)
```

```
str(read.table)
```

```
## function (file, header = FALSE, sep = "", quote = "\"'", dec = ".", numerals = c("allow.loss",
##     "warn.loss", "no.loss"), row.names, col.names, as.is = !stringsAsFactors,
##     na.strings = "NA", colClasses = NA, nrows = -1, skip = 0, check.names = TRUE,
##     fill = !blank.lines.skip, strip.white = FALSE, blank.lines.skip = TRUE,
##     comment.char = "#", allowEscapes = FALSE, flush = FALSE, stringsAsFactors = FALSE,
##     fileEncoding = "", encoding = "unknown", text, skipNul = FALSE)
```

. . . or just type the name of the function, e.g:

```
cos
```

```
## function (x)  .Primitive("cos")
```

5. Suppose payments of R Euros are deposited annually into a bank account which earns constant interest i per year. What is the accumulated value of the account at the end of n years, supposing deposits are made at the end of the year?

   The total amount at the end of the year is

$$R(1+i)^{n-1} + ... + R(1+i) + R = R\frac{(1+i)^{n-1}}{i}$$

   a. Write an R function to calculate the amount of an annuity. The user should be able to change term (in years), annual interest rate, and annual deposit. Do also give reasonable default values to the arguments if meaningful.
   b. Use the function you wrote to calculate the amount of an annuity. Compute accumulated amounts after 10 years, with periodic payments of €400, but with a vector of interest rates ranging from 0.01 through 0.2, by increments of 0.01.

```
# a.
annuAm <- function(n, R = 1, i = 0.01) {
  R * ((1 + i) ^ n - 1) / i
}

# b.
annuAm(10, 400, seq(0.01, 0.2, 0.01))
```

```
##  [1]  4184.885  4379.888  4585.552  4802.443  5031.157  5272.318  5526.579
##  [8]  5794.625  6077.172  6374.970  6688.804  7019.494  7367.900  7734.918
## [15]  8121.487  8528.588  8957.243  9408.523  9883.545 10383.473
```

6. Suppose you deposit a certain amount of money (P) in a bank where the interest rate is $i.r$, and interest is compounded, If you leave the money in the bank for n interest conversion periods, it will accumulate to

$$P(1+i.r)^n$$

   a. Write an R function which computes this amount. Your function should have three arguments (with defaults, if meaningful).
   b. Often, one interest conversion period is equal to 1 month. In this case, how much will you have in the bank at the end of 30 months, if you deposit €1,000, and the interest rate is 1% per month?
   c. Often, the interest rate is quoted as a nominal annual rate, compounded monthly. To obtain the monthly rate, the nominal annual rate is simply divided by 12. More generally, if there are m interest conversion periods in a year, the nominal annual rate is divided by m to obtain the effective interest rate for the appropriate conversion period (e.g., if the compounding is daily, the nominal annual rate is divided by 365). Use `fix()` to fix the function you defined above so that a nominal annual interest rate j and m, the number of conversion periods per year are part of the argument. The effective rate $i.r$ is assigned j/m in the body of the function. (You may delete $i.r$ from the argument list.)
   d. Use the fixed function to compute the amount accumulated from €1,000 at an annual rate of 12%, compounded daily. Compare this with the amount accumulated if the compounding is monthly or yearly.

```
# a.
cint <- function(P, n, i.r = 0.01) {
  P * (1 + i.r) ^ n
}

# b.
cint(1000, 30)
```

```
## [1] 1347.849
```

```
# c.
# using fix(cint):
cint <- function(P, j, m, n) {
  i.r <- j / m
  P * (1 + i.r) ^ (n * m)
}

# d.
cint(1000, 0.12, c(365, 12, 1), 1)
```

```
## [1] 1127.475 1126.825 1120.000
```

7. Suppose you wish to take out a mortgage on a house. You want to know what your periodic payments will be. If P is the initial amount mortgaged, $i.r$ is the effective interest rate, and n is the length of the mortgage, then the periodic payment R is given by

$$R = \frac{Pi.r}{1 - (1 + i.r)^{-n}}$$

   a. Construct a function which employs this formula.
   b. Calculate your monthly payments, if the initial amount is €100,000, the interest rate is 1%, and the number of interest conversion periods is 300.

```
# a.
mort <-  function(P, i.r, n) {
  P * i.r / (1 - (1 + i.r) ^ (-n))
}

# b.
mort(1e5, 0.01, 300)
```

```
## [1] 1053.224
```