Spring 2022 courses

DONE ds quiz 4

Settings

- After the first play, the quiz will be opened for unlimited play
- Let me know if you have any comments or corrections

Which program executes an R program?

Tip: an R program or script has the file type .R

TRUE:

- Rscript
- R CMD Batch

FALSE:

- Rterm
- Rgui
- R

In the code example, which variables are local?

```
oddcount <- function(x=0) {
  k <- 0
  for (n in x) {
    if (n %% 2 == 1) k <- k+1
  }
  return(k)
}
```

TRUE:

- `n`
- `k`

FALSE:

- `x`
- `n in x`

Feedback: variables that are defined inside a function are not known outside of it. `x` is an argument variable that must exist outside for the function to be used. If they were, they'd be global variables. However, if you try to print x it will not be known either. To make a variable global from within a function, you can use the super-assignment operator `<<-`.

Be the interpreter!

The function `oddcount` is defined in the code block blow. What is the output of the last command, `oddcount()`?

```
oddcount <- function(x=0) {
  k <- 0
  for (n in x) {
    if (n %% 2 == 1) k <- k+1
  }
  return(k)
}
oddcount()
```

TRUE:

- `0`

FALSE:

- `1`
- `ERROR`
- `<bytecode: 0x00000000051a7cf8>`

Feedback: If no argument is given, the default value `0` is returned. The `bytecode` is the byte-compiled version of the function. These versions run faster than the non-compiled versions. `oddcount` is not byte-compiled, but the built-in functions like `mean()` are. Test it by entering `mean` without an argument. You may know the bytecode concept from Java whose bytecode is executed by the Java virtual machine.

Argument list with `args()`

`ggplot` is a function in the `ggplot2` package, which is not built-in. When will `args(ggplot)` return the list of arguments?

TRUE:

- When `ggplot2` is installed and loaded

FALSE:

- When `ggplot2` is installed
- When `dev.list() == NULL`
- Never, because `args()` only works for built-in functions like `mean`
- `[ ]`

  `args(ggplot)`

  ```
  library(ggplot2)
  args(ggplot)
  ```

  ```
  function (data = NULL, mapping = aes(), ..., environment = parent.frame())
  NULL
  ```

Complete the code `???` to return the output:

```
speed <- c("medium", "slow", "fast", "fast", "fast")
f_speed <- factor(speed,
                  ordered = TRUE,
                  levels = ???)
summary(f_speed)
```

Output:

```
slow medium   fast
   1      1      3
```

TRUE:

- `c("slow","medium","fast")`

FALSE:

- `c("fast","medium","slow")`
- `c("slow","medium")`
- `c("fast","fast","fast")`

Feedback: you can create ordered factors with `factor()` by setting the `ordered` and `level` args. `summary()` is a generic function. It was introduced in the "Introduction to R" DataCamp course.

```
speed <- c("medium", "slow", "fast", "fast", "fast")
f_speed <- factor(speed,
                  ordered = TRUE,
                  levels = c("slow","medium","fast"))
summary(f_speed)
```

Be the interpreter!

The function call below should print: `"Hello, Marcus Birkenkrahe !"` But it returns this error instead:

```
: Error: unexpected symbol in "hello2(fname="Marcus" lname"
```

Can you fix the code below?

```
hello2 <- function(fname, lname) {
  print(paste("Hello, ", fname, lname,"!"))
}
hello2(fname="Marcus" lname="Birkenkrahe")
```

```
Error: unexpected symbol in "hello2(fname="Marcus" lname"
```

TRUE:

- The function call is missing a comma between the two arguments

FALSE:

- User-defined functions can only have one argument
- The arguments `"Marcus"` and `"Birkenkrahe"` should not be named in the function call
- The function call to `hello2()` contains the unexpected symbol `=`

Which command below returns the average of the vector `foo`?

```
foo <- c(1,3,4,100,NA,5,NA)
```

TRUE:

- `mean(foo,na.rm=TRUE)`

FALSE:

- `mean(foo,rm.na=TRUE)`
- `average(foo)`
- `mean(foo)`

Assign the output to the correct command!

Tip: Nile is a time series of 100 elements.

`Nile[Nile > 1200 & Nile < 1250]`	`1210 1230 1210 1220`
`which(Nile > 1200 & Nile < 1250)`	`4 8 22 26`

```
Nile[Nile > 1200 & Nile < 1250]
which(Nile > 1200 & Nile < 1250)
```

```
[1] 1210 1230 1210 1220
[1]  4  8 22 26
```

Which plotting command belongs to which plot type?

`plot(data)`	Scatter plot
`hist(data)`	Histogram
`boxplot(data)`	Boxplot
`plot(density(data))`	Density distribution plot

```
plot(mtcars$mpg ~ mtcars$wt)
```

scatter.png

Extract column vectors

`ToothGrowth` contains the factor vector `supp`, which indicates if a test subject received Vitamin C (`"VC"`) or Orange Juice (`"OJ"`). The data frame looks like this:

```
'data.frame':      60 obs. of  3 variables:
 $ len : num  4.2 11.5 7.3 5.8 6.4 10 11.2 11.2 5.2 7 ...
 $ supp: Factor w/ 2 levels "OJ","VC": 2 2 2 2 2 2 2 2 2 2 ...
 $ dose: num  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 ...
```

Which command confirms that 30 subjects received Vitamin C?

TRUE:

- `sum(ToothGrowth$supp=="VC")`

- `sum(ToothGrowth[,2,]=="VC")`
- `length(ToothGrowth[ToothGrowth=="VC"])`
- `length(which(ToothGrowth=="VC"))`

```
str(ToothGrowth)
sum(ToothGrowth$supp=="VC")
sum(ToothGrowth[,2,]=="VC")
length(ToothGrowth[ToothGrowth=="VC"])
length(which(ToothGrowth=="VC"))
```

```
'data.frame':	60 obs. of  3 variables:
 $ len : num  4.2 11.5 7.3 5.8 6.4 10 11.2 11.2 5.2 7 ...
 $ supp: Factor w/ 2 levels "OJ","VC": 2 2 2 2 2 2 2 2 2 2 ...
 $ dose: num  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 ...
[1] 30
[1] 30
[1] 30
[1] 30
```

Author: Marcus Birkenkrahe

Created: 2022-03-14 Mon 13:24

Validate