Microsoft: DAT209x Programming in R for Data Science

Bookmarks

Bookmark

Create a list of dates using the following command:

```
set.seed(449)
my.dates<-as.Date(sample(18000:20000,20), origin = "1960-01-01")
```

When dates are used as explanatory variables in regression analysis, often one needs to transform the dates into numerical values.

For this, R has the function `julian()`, which converts the dates into so-called Julian dates, which is the number of days passed since a specific time point.

Question 1

(2/2 points)
Use the julian() function to convert my.dates into the corresponding number of days passed since January 1st, 1960. Obviously, you results should be between 18000 and 20000.

Which command could you use to do so?

○ my.days<-c(julian(my.dates))

- ○ my.days<-c(julian(my.dates,origin="1960-01-01"))
- ● my.days<-c(julian(my.dates,origin=as.Date("1960-01-01"))) ✓
- ○ my.days<-c(julian(my.dates,origin=c(1,1,1960)))

What is the third element of the my.days vector?

- ○ 18656
- ● 19713 ✓
- ○ 18235
- ○ 19331

EXPLANATION

Create a set of days passed since January 1st, 1960, using the following command:

```
set.seed(119)
my.days<-sample(18000:20000,20)
```

You want to construct a data frame that contains the weekday (ie. Monday, Tuesday,...), day in month, month in year and year, for the dates corresponding to my.days.

Question 2

(1/1 point)

The `weekdays()` function can extract the weekdays from the dates. The weekdays will appear in your locale. Thus, they will be expressed in your locale language. Let's say you want to present them in English, but your system locale is not English. First, save the current locale for time objects in old.locale. Then set the locale for time objects to be English. After the programming logic, set the locale for time objects back to the old settings.

```
#save t     nt locale for time objects in old.loca
old.locale<-Sys.getlocale(
#set th     for time objects to be English
Sys.setlocale("LC_TIME","English")
#progra      ogic below

#set th      for time objects back to the old sett
Sys.setlocale("LC_TIME",old.locale)
```

Question 3

(1/1 point)

You may benefit from installing the `chron` package and access the `month.day.year()` function.

Once you've installed and loaded the package `chron`, you can extract the day, month, and year of the dates using the `month.day.year()` function.

Which command could you use to do so?

- ○ my.days.structure<-month.day.year(my.days)
- ○ my.days.structure<-month.day.year(my.days,origin="1960-01-01")
- ○ my.days.structure<-month.day.year(my.days,origin=as.Date("1960-01-01"))
- [✓ my.days.structure<-month.day.year(my.days,origin=c(1,1,1960))] ✓

EXPLANATION

Question 4

(2/2 points)
You can then create a data frame combining the weekdays extracted using the `weekdays()` function and the `my.days.structure`.

Which command could you use to do so?

- ○ my.date.info<-c(Weekday=weekdays(my.dates),my.days.structure)
- ○ my.date.info<-rbind(Weekday=weekdays(my.dates),my.days.structure)
- ○ my.date.info<-cbind(Weekday=weekdays(my.dates),my.days.structure)
- [✓ my.date.info<-data.frame(Weekday=weekdays(my.dates),my.days.structure)] ✓

What day is the last row of the data frame?

- ○ Monday
- [✓ Tuesday] ✓
- ○ Wednesday

- ○ Thursday
- ○ Friday
- ○ Saturday
- ○ Sunday

EXPLANATION