Inlämning 3

Johan Alenlöv

May 8, 2025

Inlämning

Utgå från laborationsmallen när du gör inlämningsuppgifterna. Det finns en mall för varje inlämningsuppgift, se nedan under Inlämningsuppgifter. Filen ni lämnar in ska **inte** innehålla något annat än de aktuella funktionerna, namn- och ID-variabler och ev. kommentarer.

För att lämna in ska ni gör följande:

- Lös uppgifterna enligt beskrivningarna i instruktionen.
- Se till att ni kan återskapa de testfall som visas i uppgiften.
- Se till att era funktioner klara alla tester i `markmyassignment`. Det rekommenderas att använda `markmyassignment` först när er funktion är hyfsat klar.
- På en av de schemalagda datorlaborationerna: kontakta en lärare/assistent och be om att få redovisa er inlämning.
- **Redovisning:**
  - Rensa den globala miljön (Environment i Rstudio) så att bara de variabler och funktioner som krävs för uppgiften finns kvar. Tips: Funktionen `rm()` tar bort objekt/variabler
  - Kör markmyassignment och visa att testerna går igenom.
  - Förklara kort hur ni har löst uppgifterna och svara på eventuella frågor från assistenten.
  - Om assistenten säger att ni är godkända, lämna då in er lösning på Inlämningar i kursrummet på Lisam.
- **Deadline** finns under Inlämningar i kursrummet på Lisam.

Tips!

Inlämningsuppgifterna innebär att konstruera funktioner. Ofta är det bra att bryta ned programmeringsuppgifter i färre små steg och testa att det fungerar i varje steg.

1. Lös uppgiften med vanlig kod direkt i RStudio (precis som i datorlaborationen ovan) utan att skapa en funktion.
2. Testa att du får samma resultat som testexemplen.
3. Implementera koden du skrivit i 1. ovan som en funktion.
4. Testa att du får samma resultat som i testexemplen, nu med funktionen.

Automatisk återkoppling med `markmyassignment`

Som ett komplement för att snabbt kunna få återkoppling på de olika arbetsuppgifterna finns paketet `markmyassignment`. Med detta är det möjligt att direkt få återkoppling på uppgifterna i laborationen, oavsett dator. Dock krävs internetanslutning.

Information om hur du installerar och använder `markmyassignment` för att få direkt återkoppling på dina laborationer finns att tillgå här.

Samma information finns också i R och går att läsa genom att först installera `markmyassignment`.

```
install.packages("markmyassignment")
```

Om du ska installera ett paket i PC-pularna så behöver du ange följande:

```
install.packages("markmyassignment",lib="mapp i din hemkatalog")
```

Tänk på att i sökvägar till mappar/filer i R i Windowssystem så används ``\\'', tex ``C:\\Users\\Josef''.

Därefter går det att läsa information om hur du använder `markmyassignment` med följande kommando i R:

```
vignette("markmyassignment")
```

Det går även att komma åt vignetten här. Till sist går det att komma åt hjälpfilerna och dokumentationen i `markmyassignment` på följande sätt:

```
help(package="markmyassignment")
```

Lycka till!

Chapter 1

Inlämningsuppgifter

Ladda nu ner laborationsmallen, som finns **här** och spara namn och liuID i respektive variabel. Spara filen som `inl2_[ditt liuID].R` Här är ett exempel `inl2_joswi123.R` .

För att använda `markmyassignment` i denna laboration ange:

```
library(markmyassignment)
```

```
Error in library(markmyassignment): there is no package called 'markmyassignment'
```

```
# eller
library(markmyassignment,lib="en mapp i din hemkatalog")
```

```
Error in library(markmyassignment, lib = "en mapp i din hemkatalog"): no library trees found
in 'lib.loc'
```

```
lab_path <-
"https://raw.githubusercontent.com/STIMALiU/KursRprgm2/main/Labs/Tests/inl3.yml"
set_assignment(lab_path)
```

```
Error in set_assignment(lab_path): could not find function "set_assignment"
```

1.1 Lösa inlämningsuppgifter

Här kommer lite tips till när ni ska lösa inlämningsuppgifter.

- Inlämningsuppgifter utgår ifrån att ni har gjort **övningsuppgifterna** ovan först. Tänk att ni ska göra minst 70 % av övningsuppgifterna innan ni börjar med inlämningsuppgifterna.
- Se först till att ni förstår ”problemet” i uppgiften. Vad ska funktionen göra? Beskriv problemet för er själva. Vissa problem är det bra att bryta ner i mindre delproblem.
- Börja med att lösa problemet (eller det första delproblemet) med vanlig kod (dvs inte i en funktion) i ett R-script.
  - Lös ett delproblem i taget vid behov. Sätt sedan samman lösningarna på delproblemen. Under sök om koden fungerar som den ska.
- Sätt sedan in er kod i en funktion. Testa om funktionen kan återskapa de exempel som visas under beskrivningarna (testfallen). Se till att funktionen har rätt namn och rätt namn på argumenten.
- Ta er funktion och lägg in den i kodmallen, dvs i ett nytt R-script. Skapa variablerna `Namn` och `LiuId` med rätt innehåll. Spara filen med rätt namn.
- Kommentera bort all kod i er fil som inte är de aktuella funktionerna, namn- och ID-variabler och ev. kommentarer.
  - Detta minskar risken för fel.

- Testa nu att använda markmyassignment för att rätta er funktion.
  - Använd markmyassignment bara när er funktionen är ”hyfsat klar”, annars kommer ni att få många fel i markmyassignment, och det är svårt att veta var man ska börja kolla.
  - Det är vikitgt att funktionen rätt namn och rätt namn på argumenten, annars kommer ni att få många fel i markmyassignment.
- Om ni inte klarar alla tester i markmyassignment: undersök vilka felmeddelanden som ni får. Ta hjälp av lärare/labbassitenter vid behov om det är svårt att tolka.
- Innan ni lämnar in:
  - Se till att ni har löst uppgiften på det sätt som den är beskriven i uppgiftstexten. Ibland måste en viss metod användas för att lösa uppgiften. Ibland är vissa funktioner inte tillåtna i en viss uppgift. Så läs noga i varje uppgift vad som gäller.
  - Se till att ni klarar alla tester i markmyassignment

Dokumentation och kodstil

Från och med denna laboration och de resterade laborationerna i kursen så ska ni förutom att lösa angivna uppgifter också **kommentera** era funktioner och ha en **god kodstil** för att bli godkända.

- Kodstil:
  - Det viktiga är att koden ska vara **tydlig** och **läsbar**.
  - Följ någon av kodstilarna i Datorlaboration 4. Ni måste inte följa dessa exakt, men koden ska se bra ut och var konsekventa i den stil som ni väljer att använda.
  - Tänk särskilt på:
    - Enhetlighet och struktur
    - Ha lämplig indentering och avstånd
    - Ha bra variabelnamn
- Kommentarer:
  - Funktionshuvud: Använd mallen för `roxygen2` som ges i Datorlaboration 4. Ni ska ha med:
    - `@title` Här skriver ni funktionens namn
    - `@description` Förklara kort vad funktionen gör
    - `@param` Skriv först arguments namn, sen mellanslag och sen förklara kort argumentet. Upprepa detta för alla argument i funktionen. Ex: `x Numerisk vektor, används vid beräkning av medelvärde.`
    - `@return` Förklara vad funktionen returnerar
  - Kommenter i funktionen: Era lösningar ska innehålla lämpliga kommenater, där ni förklarar de övergripande dragen och de viktiga stegen i er kod. Ni behöver inte föklara alla detaljer. Kommentarerna ska berätta sådant för programmeraren som inte står i koden. Använd luft och kommentarer för att gruppera och strukturera er kod.
  - Tips: Tänk att det ska vara lätt att förstå er funktion långt senare, tex om ett år. Vilka kommentarer behövs då?

För att bli godkänd på inlämningsuppgifterna måste ni följa ovanstående instruktioner för kommentarer och kodstil.

1.2 classroom()

Skapa en konstruktor som skapar ett klassrumsobjekt med funktionen classroom(). Funktionen ska ha två argument, seats and whiteboards, som anger hur många platser och hur många tavlor som finns i rummet. Denna information ska sparas i en lista. Objektet som returneras ska vara av klassen classroom. Funktionen ska också kontrollera att seats är strikt större än 0 och att whiteboard är större eller lika med 0. Ni kan välja felmeddelande själva.

Implementera sedan en specifik metod för klassen classroom till generiska funktionen print(). Ni ska alltså skapa funktionen print.classroom(). Beroende på antalet sittplatser ska följande skrivas ut:

- 10 eller färre sittplatser: A group room for [seats] students with [whiteboard] whiteboards.
- 11 till 50 sittplatser: A classroom for [seats] students with [whiteboard] whiteboards.
- fler än 50 sittplatser: A lecture hall for [seats] students with [whiteboard] whiteboards.
- Om antalet whiteboards är noll ska istället för siffran 0, ordet no användas.

```
is.function(print.classroom)

[1] TRUE

cr1 <- classroom(10,0)
print(cr1)

A group room for 10 students with no whiteboards.

print.classroom(cr1)

A group room for 10 students with no whiteboards.

cr1[[1]]

[1] 10

cr1[[2]]

[1] 0

classroom(40,4)

A classroom for 40 students with 4 whiteboards.

cr2 <- classroom(150,12)
print(cr2)

A lecture hall for 150 students with 12 whiteboards.

class(cr2)

[1] "classroom"

classroom(0,10)

Error in classroom(0, 10): Error in classroom() : Not a positive value

classroom(15,-1)

Error in classroom(15, -1): Error in classroom() : Not a non-negative value
```

1.3 `give_blood()`

För blodgivare finns vissa regler för när hen får ge blod. Här står det vilka regler som gäller. Ni ska skriva en funktion som ska hjälpa en blodgivare att veta när hen får ge blod, utifrån några av reglerna. Funktionen ska heta `give_blood()` och ha argumenten:

- `lasttime`: ett datum som anger senaste gången blodgivaren gav blod, default ska vara idag. tips: today()
- `holiday`: ska vara antingen: 1) ett interval-objekt som anger start- och slutdatum för en utlandsresa. Startdatum är det datum som personen lämnar Sverige och slutdatum är det datum som personen kommer hem till Sverige. 2) Defaultvärde ska vara "`hemma`", vilket indikerar att det inte blir någon resa.
- `sex`: antar värdet "f" för kvinna och "m" för man
- `type_of_travel`: "`malaria`" indikerar resa till ett land där det finns malaria och "`other`" indikerar resa till ett land utan malaria. Ska vara NULL om `holiday` har värdet "hemma".

Alla datum ska vara på formen **"year-month-day"**. Funktionen ska givet argumenten räkna ut ett datum när blodgivaren får ge blod igen och returnera datumet. Vi utgår ifrån att blodgivaren vill ge blod så **ofta** som möjligt. Funktionen ska följa följande regler:

- Minsta tid mellan blodgivningstillfällen: kvinnor 4 månader, män 3 månader, båda anger relativ tid. Efter exakt denna tid kan personen ge blod.
- Om personen varit i ett land där det inte finns malaria ska hen vänta (vara i karantän) 4 veckor (relativ tid) efter slutdatum i argumentet `holiday` innan hen får ge blod.
- Om personen varit i ett land där det finns malaria ska hen vänta (vara i karantän) 6 månader (relativ tid) efter slutdatum i argumentet `holiday` innan hen får ge blod.
- När det gäller karantänen får personen inte ge blod under karantänen utan det är första dagen efter karantänen personen får ge blod.
- Vi utgår ifrån att blodgivningscentralen bara är öppen på vardagar (måndag till fredag), så givet de tidigare reglerna så ska den första möjliga vardagen väljas.

Nedan följer ett förslag på lösningsordning:

1. Undersök om personen varit hemma, på resa i land med malaria eller i land utan malaria. Addera eventuell tilläggstid till slutdatum och spara som `extraTime`. Tänk på att ta hänsym till fallet då personen inte reser, tex genom att sätta `extraTime` till samma datum som `lasttime`.
   Tips: `int_end(), months(), weeks()`
2. Givet om den är en man eller kvinna räkna ut när personen tidigast får ge blod, spara det datumet i variablen `suggestion`. **Tips:** `months()`
3. Kolla om `suggestion` inträffar efter `extraTime`, om så är fallet ange `suggestion` som förslag för blodgivnig. Om så inte är fallet, ange dagen efter `extraTime` som förslag.
4. Kontrollera att den angivna dagen är en vardag, om inte ange nästa vardag som förslag.
   Tips! `?wday(), ?days()`
5. Returnera förslaget som en text-sträng på formen:
   ``` `year=[året] month=[månaden] day=[dagen] weekday=[veckodagen]' '. ```
   Tex om föreslaget datum är 2014-02-21 så ska strängen bli:
   ``` `year=2014 month=Feb day=19 weekday=Friday' '. ```
   Tips: `year(), month(), day(), paste()`

Tips! Det kan vara så att weekday() returnerar veckodagarna på svenska. För att returnera veckodagar på engelska finns följande tips:

```
Sys.setlocale("LC_TIME", "English")
# eller
Sys.setlocale("LC_TIME", "C")
```

Om ni har svårt att ändra språk (locale) så finns följande tips:

```r
# månad:
# lägg denna funktion utanför give_blood() i filen ni lämnar in
wrap_month<-function(x){ month.name[x] }

library(lubridate)
# exempel på datum:
x <- ymd("2012-03-28")
print(month(x))
print(month.name)

# ha denna kod i give_blood().
month_name_eng<-substr(x = wrap_month(month(x,label = FALSE)),start = 1,stop = 3)

# veckodag:
week_name <- c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday")
print(wday(x))
week_name[wday(x)]
```

Kolla om funktionen uppfyller testfallen nedan:

```r
library(lubridate)
```

```
Attaching package: 'lubridate'
The following object is masked _by_ '.GlobalEnv':
      leap_year
The following objects are masked from 'package:base':
      date, intersect, setdiff, union
```

```r
Sys.setlocale("LC_TIME", "C")
```

```
[1] "C"
```

```r
# Test 1:
day1<-ymd("2014-02-24")
give_blood(lasttime=day1,holiday="hemma",sex="m",type_of_travel=NULL)
```

```
[1] "year=2014 month=May day=26 weekday=Monday"
```

```r
give_blood(lasttime=day1,holiday="hemma",sex="f",type_of_travel=NULL)
```

```
[1] "year=2014 month=Jun day=24 weekday=Tuesday"
```

```r
# Test 2:
day2<-ymd("2014-03-23")
day3<-ymd("2014-04-24")
holiday1<-interval(day2,day3)
give_blood(lasttime=day1,holiday=holiday1,sex="m",type_of_travel="malaria")
```

```
[1] "year=2014 month=Oct day=27 weekday=Monday"
```

```r
give_blood(lasttime=day1,holiday=holiday1,sex="f",type_of_travel="malaria")
```

```
[1] "year=2014 month=Oct day=27 weekday=Monday"
```

```r
# Test 3:
day4<-ymd("2014-04-13")
day5<-ymd("2014-05-23")
holiday2<-interval(day4, day5)
give_blood(lasttime=day1,holiday=holiday2,sex="m",type_of_travel="other")
```

```
[1] "year=2014 month=Jun day=23 weekday=Monday"
```

```r
give_blood(lasttime=day1,holiday=holiday2,sex="f",type_of_travel="other")
```

```
[1] "year=2014 month=Jun day=24 weekday=Tuesday"
```

Grattis! Nu är du klar!