Bios 6301: Assignment 8

Jeongwon Choi

Due Tuesday, 15 November, 1:00 PM

$5^{n=day}$ points taken off for each day late.

30 points total.

Submit a single knitr file (named `homework8.rmd`), along with a valid PDF output file. Inside the file, clearly indicate which parts of your responses go with which problems (you may use the original homework document as a template). Add your name as `author` to the file's metadata section. Raw R code/output or word processor files are not acceptable.

Failure to name file `homework8.rmd` or include author name may result in 5 points taken off.

Question 1

15 points

Install the `readxl` package and run the following

```
fn <- 'icd10.xlsx'
if(file.access(fn, mode = 4) == -1) {
    url <- "https://www.cdc.gov/nhsn/xls/icd10-pcs-pcm-nhsn-opc.xlsx"
    download.file(url, destfile = fn, mode = 'wb')
}
dat <- readxl::read_excel(fn, sheet = 2)
```

1. Show the class of `dat`. (1 point)

```
class(dat)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
# attr(dat, "class")
```

2. Show the methods available for objects of the given class (if there are multiple classes, show methods for all classes). (3 points)

```
methods( ,'tbl')
```

```
##  [1] $<-         [[<-        [<-         coerce      format      initialize
##  [7] Ops         print       show        slotsFromS3
## see '?methods' for accessing help and source code
```

```
methods( ,'tbl_df')
```

```
##  [1] $            $<-          [            [[           [[<-
##  [6] [<-          as.data.frame coerce      initialize   names<-
## [11] Ops          row.names<-   show        slotsFromS3  str
## see '?methods' for accessing help and source code
```

```
methods( ,'data.frame')
```

```
##  [1] $<-          [            [[           [[<-         [<-
##  [6] aggregate    anyDuplicated anyNA        as.data.frame as.list
## [11] as.matrix    as.vector    by           cbind        coerce
## [16] dim          dimnames     dimnames<-   droplevels   duplicated
## [21] edit         format       formula      head         initialize
## [26] is.na        Math         merge        na.exclude   na.omit
## [31] Ops          plot         print        prompt       rbind
## [36] row.names    row.names<-  rowsum       show         slotsFromS3
## [41] split        split<-      stack        str          subset
## [46] summary      Summary      t            tail         transform
## [51] type.convert unique       unstack      within       xtfrm
## see '?methods' for accessing help and source code
```

3. If you call `print(dat)`, what print method is being dispatched? (1 point)

```
print(dat)
```

```
## # A tibble: 9,726 × 4
##    `Procedure Code Category` `ICD-10-PCS Codes` Procedure Code Descrip…¹ Code …²
##    <chr>                    <chr>              <chr>                     <chr>
##  1 AAA                      04B00ZZ            Excision of Abdominal A… No cha…
##  2 AAA                      04B04ZZ            Excision of Abdominal A… No cha…
##  3 AAA                      04R007Z            Replacement of Abdomina… No cha…
##  4 AAA                      04R00JZ            Replacement of Abdomina… No cha…
##  5 AAA                      04R00KZ            Replacement of Abdomina… No cha…
##  6 AAA                      04R047Z            Replacement of Abdomina… No cha…
##  7 AAA                      04R04JZ            Replacement of Abdomina… No cha…
##  8 AAA                      04R04KZ            Replacement of Abdomina… No cha…
##  9 AMP                      0X600ZZ            Detachment at Right For… No cha…
## 10 AMP                      0X610ZZ            Detachment at Left Fore… No cha…
## # … with 9,716 more rows, and abbreviated variable names
## #   ¹`Procedure Code Descriptions`, ²`Code Status`
```

There are 3 classes in dat - tbl, tbl_df, and data.frame. UseMethod searches for a match sequentially, for this case, tbl, tbl_df, and so on. If this cannot find an appropriate type of function, default function is used. So in this case, tbl is dispatched.

4. Set the class of `dat` to be a data.frame. (1 point)

```
class(dat) <- c("data.frame")
#dat<- structure(list(), class = "data.frame")
class(dat)
```

```
## [1] "data.frame"
```

5. If you call `print(dat)` again, what print method is being dispatched? (1 point)

In this case, print method of data.frame is being dispatched.

Define a new generic function `nUnique` with the code below.

```
nUnique <- function(x) {
    UseMethod('nUnique')
}
```

6. Write a default method for `nUnique` to count the number of unique values in an element. (2 points)

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.2.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
nUnique.default <- function(x, ...){
  n_distinct(x)
}
```

7. Check your function (2 points)

```
nUnique(letters) # should return 26
nUnique(sample(10, 100, replace = TRUE)) # should return 10 (probably)
```

8. Write a data.frame method for `nUnique` to operate on data.frame objects. This version should return counts for each column in a data.frame. (2 points)

```
nUnique.data.frame <- function(x, ...){
  l1 <- lapply(x, unique)
  l2 <- lapply(l1, length)
  unlist(l2)
    }
```

9. Check your function (2 points)

```
nUnique(dat)
```

Question 2

15 points

Programming with classes. The following function will generate random patient information.

```
makePatient <- function() {
  vowel <- grep("[aeiou]", letters)
  cons <- grep("[^aeiou]", letters)
  name <- paste(sample(LETTERS[cons], 1), sample(letters[vowel], 1), sample(letters[cons], 1), s
ep='')
  gender <- factor(sample(0:1, 1), levels=0:1, labels=c('female','male'))
  dob <- as.Date(sample(7500, 1), origin="1970-01-01")
  n <- sample(6, 1)
  doa <- as.Date(sample(1500, n), origin="2010-01-01")
  pulse <- round(rnorm(n, 80, 10))
  temp <- round(rnorm(n, 98.4, 0.3), 2)
  fluid <- round(runif(n), 2)
  list(name, gender, dob, doa, pulse, temp, fluid)
}
```

1. Create an S3 class `medicalRecord` for objects that are a list with the named elements `name`, `gender`, `date_of_birth`, `date_of_admission`, `pulse`, `temperature`, `fluid_intake`. Note that an individual patient may have multiple measurements for some measurements. Set the RNG seed to `8` and create a medical record by taking the output of `makePatient`. Print the medical record, and print the class of the medical record. (5 points)

```
# define class "medicalRecord"
medicalRecord <- setClass("medicalRecord", slots = c(name="character", gender="factor", date_of_
birth="Date", date_of_admission="Date", pulse="numeric", temperature="numeric", fluid_intake="nu
meric"),contain="list")

# set random seed and generate a patient data
set.seed(8)
mr <- makePatient()
# names(mp) <- c('name', 'gender', 'date_of_birth', 'date_of_admission', 'pulse', 'temperature',
'fluid_intake')

# assign medicalRecord class to the data
mr.new <- new("medicalRecord",name=mr[[1]],gender=mr[[2]],date_of_birth=mr[[3]],
              date_of_admission=mr[[4]], pulse=mr[[5]], temperature=mr[[6]],
              fluid_intake=mr[[7]])

# show the class of the medical record
print(class(mr.new))
```

```
## [1] "medicalRecord"
## attr(,"package")
## [1] ".GlobalEnv"
```

```
# print the medical record
print(mr.new)
```

```
## An object of class "medicalRecord"
## list()
## Slot "name":
## [1] "Yes"
##
## Slot "gender":
## [1] male
## Levels: female male
##
## Slot "date_of_birth":
## [1] "1977-05-03"
##
## Slot "date_of_admission":
## [1] "2013-06-09" "2013-07-02"
##
## Slot "pulse":
## [1] 79 78
##
## Slot "temperature":
## [1] 98.07 97.50
##
## Slot "fluid_intake":
## [1] 0.28 0.52
```

2. Write a `medicalRecord` method for the generic function `mean`, which returns averages for pulse, temperature and fluids. Also write a `medicalRecord` method for `print`, which employs some nice formatting, perhaps arranging measurements by date, and `plot`, that generates a composite plot of measurements over time. Call each function for the medical record created in part 1. (5 points)

```
mean.medicalRecord <- function(x){
  return(data.frame(pulseavg=mean(x@pulse), tempavg=mean(x@temperature), fluiavg=mean(x@fluid_in
take)))
}
```

```
print.medicalRecord <- function(x){
  cat(paste("Name:", x@name, "\n", "Gender:", x@gender,"\n", "Date of birth:", x@date_of_birth,
"\n"))

  cat("Measurement Date:\n")

  newd0 = data.frame(date_of_admission = x@date_of_admission, pulse = x@pulse, temperature = x@t
emperature, fluid_intake = x@fluid_intake)
  newd0 = newd0[order(newd0$date_of_admission), ]
  for(i in 1:length(newd0$date_of_admission)){
   print(paste0("Date: ", newd0$date_of_admission[i],  "|", "Pulse: ", newd0$pulse[i],"|", "Tem
p: ", newd0$temperature[i],"|", "Fluid_intake: ", newd0$fluid_intake[i]))
  }
}
```

```
plot.medicalRecord <- function(x){
  newd = data.frame(date_of_admission=x@date_of_admission, pulse=x@pulse, temperature=x@temperat
ure, fluid_intake=x@fluid_intake)
  newd = newd[order(newd$date_of_admission), ]
 par(mfrow = c(1,3))
  plot(newd$date_of_admission, newd$pulse, type = "l", main = "pulse")
  plot(newd$date_of_admission, newd$tempearture, type = "l", main = "temperature")
  plot(newd$date_of_admission, newd$fluid_intake, type = "l", main = "fluid_intake")
}
```

3. Create a further class for a cohort (group) of patients, and write methods for `mean` and `print` which, when applied to a cohort, apply mean or print to each patient contained in the cohort. Hint: think of this as a "container" for patients. Reset the RNG seed to 8 and create a cohort of ten patients, then show the output for `mean` and `print`. (5 points)

```
# define class "cohort"
cohort <- setClass("cohort",contain="list") # this is a just list consisting of medical records

# define methods for 'mean' and 'print'
mean.cohort = function(arr){

  df.cohort <- data.frame()

  for (i in 1:length(arr)){
    this_df <- data.frame(avg.pulse=mean(arr[[i]]@pulse),
                          avg.temperature=mean(arr[[i]]@temperature),
                          avg.fluid_intake=mean(arr[[i]]@fluid_intake))
    df.cohort <- rbind(df.cohort, this_df)
  }
  return(df.cohort)
}
print.cohort = function(arr){

  for (i in 1:length(arr)){
    print(paste0('Patient ',i))
    print(arr[[i]]) # arr[[i]] has class of medicalRecord
  }
}

# set random seed
set.seed(8)

# generate a cohort of ten patients
patients <- lapply(1:10, function(d){return(makePatient())})

# transform each record into medicalRecord class
patients.mr <- lapply(patients,function(li){
  new("medicalRecord",name=li[[1]],gender=li[[2]],
date_of_birth=li[[3]],date_of_admission=li[[4]], pulse=li[[5]], temperature=li[[6]], fluid_intak
e=li[[7]])})

# assign cohort class to patient.mr
patients.cohort <- new('cohort',patients.mr)

# apply mean to cohort
mean(patients.cohort)
```

avg.pulse <dbl>	avg.temperature <dbl>	avg.fluid_intake <dbl>
78.50000	97.78500	0.4000000
86.33333	98.39667	0.4133333
77.00000	98.64750	0.5200000
83.16667	98.48500	0.2966667

avg.pulse <dbl>	avg.temperature <dbl>	avg.fluid_intake <dbl>
83.50000	98.45000	0.4525000
84.40000	98.48400	0.5220000
76.50000	98.38000	0.3975000
75.00000	98.36750	0.5225000
73.00000	98.36000	0.1500000
77.00000	98.54000	0.1500000

1-10 of 10 rows

```
# apply print to cohort
print(patients.cohort)
```

```
## [1] "Patient 1"
## Name: Yes
##  Gender: male
##  Date of birth: 1977-05-03
## Measurement Date:
## [1] "Date: 2013-06-09|Pulse: 79|Temp: 98.07|Fluid_intake: 0.28"
## [1] "Date: 2013-07-02|Pulse: 78|Temp: 97.5|Fluid_intake: 0.52"
## [1] "Patient 2"
## Name: Fal
##  Gender: male
##  Date of birth: 1988-05-24
## Measurement Date:
## [1] "Date: 2010-11-16|Pulse: 76|Temp: 98.23|Fluid_intake: 0.18"
## [1] "Date: 2013-03-24|Pulse: 87|Temp: 98.21|Fluid_intake: 0.1"
## [1] "Date: 2013-09-12|Pulse: 96|Temp: 98.75|Fluid_intake: 0.96"
## [1] "Patient 3"
## Name: Zog
##  Gender: male
##  Date of birth: 1988-12-14
## Measurement Date:
## [1] "Date: 2010-02-24|Pulse: 84|Temp: 98.54|Fluid_intake: 0.4"
## [1] "Date: 2013-03-25|Pulse: 69|Temp: 98.49|Fluid_intake: 0.81"
## [1] "Date: 2013-07-29|Pulse: 75|Temp: 98.82|Fluid_intake: 0.59"
## [1] "Date: 2013-10-27|Pulse: 80|Temp: 98.74|Fluid_intake: 0.28"
## [1] "Patient 4"
## Name: Yol
##  Gender: male
##  Date of birth: 1986-03-11
## Measurement Date:
## [1] "Date: 2010-02-22|Pulse: 84|Temp: 98.87|Fluid_intake: 0.39"
## [1] "Date: 2011-12-27|Pulse: 89|Temp: 98.27|Fluid_intake: 0.97"
## [1] "Date: 2012-03-10|Pulse: 87|Temp: 98.78|Fluid_intake: 0.12"
## [1] "Date: 2012-11-26|Pulse: 92|Temp: 98.26|Fluid_intake: 0.14"
## [1] "Date: 2013-03-24|Pulse: 78|Temp: 98.44|Fluid_intake: 0.13"
## [1] "Date: 2014-01-28|Pulse: 69|Temp: 98.29|Fluid_intake: 0.03"
## [1] "Patient 5"
## Name: Yak
##  Gender: female
##  Date of birth: 1983-09-15
## Measurement Date:
## [1] "Date: 2011-07-19|Pulse: 75|Temp: 98.58|Fluid_intake: 0.6"
## [1] "Date: 2012-04-07|Pulse: 88|Temp: 97.53|Fluid_intake: 0.29"
## [1] "Date: 2012-07-11|Pulse: 81|Temp: 99.11|Fluid_intake: 0.66"
## [1] "Date: 2012-08-30|Pulse: 90|Temp: 98.58|Fluid_intake: 0.26"
## [1] "Patient 6"
## Name: Gaf
##  Gender: female
##  Date of birth: 1978-04-27
## Measurement Date:
## [1] "Date: 2010-07-19|Pulse: 91|Temp: 98.01|Fluid_intake: 0.47"
## [1] "Date: 2011-05-03|Pulse: 90|Temp: 98.61|Fluid_intake: 0.36"
## [1] "Date: 2012-04-24|Pulse: 89|Temp: 98.32|Fluid_intake: 0.42"
```

```
## [1] "Date: 2012-08-06|Pulse: 77|Temp: 98.96|Fluid_intake: 0.74"
## [1] "Date: 2013-08-21|Pulse: 75|Temp: 98.52|Fluid_intake: 0.62"
## [1] "Patient 7"
## Name: Kuw
##  Gender: female
##  Date of birth: 1980-11-07
## Measurement Date:
## [1] "Date: 2010-10-03|Pulse: 82|Temp: 98.49|Fluid_intake: 0.12"
## [1] "Date: 2010-10-29|Pulse: 81|Temp: 98.17|Fluid_intake: 0.93"
## [1] "Date: 2011-09-16|Pulse: 72|Temp: 98.21|Fluid_intake: 0.29"
## [1] "Date: 2012-07-10|Pulse: 71|Temp: 98.65|Fluid_intake: 0.25"
## [1] "Patient 8"
## Name: Mav
##  Gender: female
##  Date of birth: 1989-07-16
## Measurement Date:
## [1] "Date: 2010-02-08|Pulse: 66|Temp: 97.95|Fluid_intake: 0.79"
## [1] "Date: 2010-04-19|Pulse: 88|Temp: 98|Fluid_intake: 0.5"
## [1] "Date: 2010-06-11|Pulse: 83|Temp: 98.45|Fluid_intake: 0.79"
## [1] "Date: 2012-03-02|Pulse: 63|Temp: 99.07|Fluid_intake: 0.01"
## [1] "Patient 9"
## Name: Fel
##  Gender: male
##  Date of birth: 1985-08-16
## Measurement Date:
## [1] "Date: 2010-09-26|Pulse: 81|Temp: 98.51|Fluid_intake: 0.24"
## [1] "Date: 2012-06-24|Pulse: 65|Temp: 98.21|Fluid_intake: 0.06"
## [1] "Patient 10"
## Name: Say
##  Gender: female
##  Date of birth: 1974-09-22
## Measurement Date:
## [1] "Date: 2010-03-14|Pulse: 77|Temp: 98.54|Fluid_intake: 0.15"
```