STAT 2200: Problem Set 4

Jack Ambery

Due: Monday, 3/18 at the beginning of class

- You may discuss this assignment with other students in the class, but you may not sit down and type it up with them or show them your code. You also may not discuss the assignment with anyone who isn't in our class, nor may you look up anything online.
- You must type up your homework using R Markdown. I want to see all of your code and output, and any answers you provide that aren't code must be typed above the respective `R` code chunk.
- Make sure none of your code runs off the page, otherwise you will lose points.
- You must print and turn in a PDF of your homework. I won't accept anything else (e.g., a Word document). All pages must be stapled together.
- This assignment is worth 100 points.

1. Create a sequence of 20 numbers as follows: the first number is 5, and each number thereafter is found by multiplying the current number by 1.5 and then subtracting 1. Print the resulting sequence.

```
arr <- c(5, rep(0, times = 19))
for (i in 1:19)
  arr[i+1] <- (arr[i] * 1.5) - 1

print(arr)
```

```
##  [1]    5.00000    6.50000    8.75000   12.12500   17.18750   24.78125
##  [7]   36.17188   53.25781   78.88672  117.33008  174.99512  261.49268
## [13]  391.23901  585.85852  877.78778 1315.68167 1972.52251 2957.78376
## [19] 4435.67564 6652.51346
```

2. Suppose you have the following vector of probabilities, each of which measures the probability a person has a particular disease: 0.25, 0.83, 0.76, 0.25, 0.33, 0.51, 0.67, 0.50, 0.54, 0.75, 0.71, 0.03, 0.38, 0.64, and 0.19. Create a new variable where a one indicates that the person is predicted to have the disease (based on a probability of 0.5 or above) and a zero indicates that the person is predicted not to have the disease (based on a probability under 0.5). Do this two different ways:

- first, using a loop, and
- second, using the `ifelse()` function.

Print the resulting vectors. Note that they should contain the exact same values!

```
probabilities <- c(0.25, 0.83, 0.76, 0.25, 0.33, 0.51, 0.67, 0.50, 0.54, 0.75, 0.71, 0.0
likelyhoods <- vector()

for (i in 1:length(probabilities))
  if (probabilities[i] >= 0.5) {
    likelyhoods[i] <- 1
  } else {
    likelyhoods[i] <- 0
  }
print('Likelyhood using loop:')
```

```
## [1] "Likelyhood using loop:"
```

```
print(likelyhoods)
```

```
##  [1] 0 1 1 0 0 1 1 1 1 1 1 0 0 1 0
```

```
likelyhoods2 <- ifelse(probabilities >= 0.5, 1, 0)
print('Likelyhood using ifelse:')
```

```
## [1] "Likelyhood using ifelse:"
```

```
print(likelyhoods2)
```

```
##  [1] 0 1 1 0 0 1 1 1 1 1 1 0 0 1 0
```

3. In this problem you will work with the *Cars2020* dataset, which you downloaded from http://www.lock5stat.com/datapage3e.html and used on Homework 3. Please work with the CSV again here. Create a new variable that classifies the fuel efficiencies of the cars in the dataset based on the following made-up criteria: a highway mpg under 20 is considered poor, a highway mpg between 20 and 25 (inclusive) is considered acceptable, a highway mpg above 25 and at most 35 is considered good, and a highway mpg above 35 is considered great. Make sure the dataset contains the new variable, and then print only the following for every car in the dataset: the make and model of the car, as well as the classified fuel efficiency.

```
Cars2020 <- read.csv("../datasets/Cars2020.csv")

fuelEff <- Cars2020$HwyMPG
fuelClass <- vector()

for (i in 1:length(fuelEff)) {
  if (fuelEff[i] < 20) {
    fuelClass[i] = "poor"
```

```
  } else if (fuelEff[i] >= 20 & fuelEff[i] <= 25) {
    fuelClass[i] = "acceptable"
  } else if (fuelEff[i] > 25 & fuelEff[i] <= 35) {
    fuelClass[i] = "good"
  } else { #fuelEff[i] > 35
    fuelClass[i] = "great"
  }
}
# adding it to Cars2020 dataframe
Cars2020$fuelClass <- fuelClass

#printing out specific columns of Cars2020
print(Cars2020[c("Make","Model","fuelClass")])
```

```
##           Make           Model  fuelClass
## 1        Acura             MDX       good
## 2        Acura             RLX      great
## 3         Audi              A3      great
## 4         Audi              A4      great
## 5         Audi              A6      great
## 6         Audi              A8       good
## 7         Audi              Q3       good
## 8         Audi              TT       good
## 9          BMW       3 Series      great
## 10         BMW       5 series      great
## 11         BMW       7 Series       good
## 12         BMW             X1      great
## 13         BMW             Z4      great
## 14       Buick        Enclave       good
## 15       Buick       Envision       good
## 16       Buick          Regal       good
## 17      Buick           Encore       good
## 18    Cadillac       Escalade acceptable
## 19    Cadillac            XT4       good
## 20    Cadillac            XT5       good
## 21    Cadillac            XT6       good
## 22   Chevrolet         Camaro       good
## 23   Chevrolet       Corvette       good
## 24   Chevrolet         Impala       good
## 25   Chevrolet         Malibu      great
## 26   Chevrolet          Spark      great
## 27   Chevrolet       Suburban acceptable
## 28   Chevrolet          Tahoe acceptable
## 29   Chevrolet       Traverse       good
```

```
## 30  Chevrolet          Blazer       good
## 31    Chrysler             300       good
## 32    Chrysler        Pacifica       good
## 33       Dodge         Durango acceptable
## 34       Dodge         Journey acceptable
## 35      Dodge       Challenger       good
## 36      Dodge          Charger       good
## 37      Dodge    Grand Caravan       good
## 38        Fiat            500L      great
## 39        Ford          Escape      great
## 40        Ford      Expedition acceptable
## 41        Ford        Explorer       good
## 42        Ford          Fusion       good
## 43        Ford         Mustang       good
## 44       Ford         EcoSport       good
## 45         GMC         Terrain       good
## 46        GMC           Acadia       good
## 47        GMC            Yukon acceptable
## 48       Honda        Passport       good
## 49      Honda           Accord      great
## 50      Honda             CR-V      great
## 51      Honda          Odyssey       good
## 52      Honda            Pilot       good
## 53     Hyundai          Accent      great
## 54     Hyundai          Elantra      great
## 55     Hyundai             Kona       good
## 56     Hyundai        Santa Fe       good
## 57     Hyundai          Sonata      great
## 58     Hyundai        Veloster      great
## 59    Infiniti             Q50       good
## 60    Infiniti            QX60       good
## 61      Jaguar          E-Pace       good
## 62      Jaguar              XF       good
## 63        Jeep         Compass       good
## 64        Jeep Grand Cherokee acceptable
## 65         Kia         Cadenza      great
## 66         Kia           Forte      great
## 67         Kia          Optima      great
## 68         Kia             Rio      great
## 69         Kia            Soul       good
## 70  Land Rover   Range Evoque       good
## 71       Lexus              GS       good
## 72       Lexus              LS       good
## 73       Lexus              RX       good
## 74     Lincoln         Aviator       good
```

```
## 75     Lincoln      Navigator acceptable
## 76       Mazda              3      great
## 77       Mazda              6      great
## 78       Mazda           CX-3      great
## 79       Mazda           CX-9       good
## 80       Mazda       Miata-MX5      great
## 81    Mercedes         C-Class       good
## 82    Mercedes             GLC       good
## 83  Mitsubishi         Mirage      great
## 84      Nissan         Altima      great
## 85      Nissan         Murano       good
## 86      Nissan     Pathfinder acceptable
## 87      Nissan          Rogue       good
## 88      Nissan         Sentra      great
## 89      Nissan          Versa      great
## 90      Porche            911       good
## 91      Porche        Cayenne       good
## 92      Subaru      Crosstrek      great
## 93      Subaru       Forester      great
## 94      Subaru        Impreza      great
## 95      Subaru        Outback       good
## 96     Subaru          Legacy      great
## 97      Toyota         Corolla      great
## 98      Toyoto         4Runner acceptable
## 99      Toyoto           Camry      great
## 100     Toyoto     Highlander       good
## 101     Toyoto           Rav4      great
## 102     Toyoto        Sequioa acceptable
## 103     Toyoto         Sienna       good
## 104     Toyoto          Yaris      great
## 105 Volkswagon           Golf      great
## 106 Volkswagon          Jetta      great
## 107 Volkswagon         Passat      great
## 108 Volkswagon         Tiguan       good
## 109      Volvo            S60      great
## 110      Volvo           XC90       good
```

4. Use a loop to fill a 6×8 (i.e., six rows and eight columns) matrix as follows: if the sum of the indices of an element's position in the matrix is even, then the new element is the product of the indices. On the other hand, if the sum of the indices is odd, then the new element is the sum of the indices plus three. Print the resulting matrix.

```r
table <- matrix(0, nrow = 6, ncol = 8)

for(i in 1:nrow(table)) {
```

```
  for (j in 1:ncol(table)) {
    if ((i + j) %% 2 == 0) { # if even
      table[i, j] <- i*j
    } else { # if odd
      table[i, j] <- i + j + 3
    }
  }
}
print(table)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1,]    1    6    3    8    5   10    7   12
## [2,]    6    4    8    8   10   12   12   16
## [3,]    3    8    9   10   15   12   21   14
## [4,]    8    8   10   16   12   24   14   32
## [5,]    5   10   15   12   25   14   35   16
## [6,]   10   12   12   24   14   36   16   48
```

5. Suppose I taught a course last semester that consisted of homework, two midterm exams, and a final exam, which are weighted as follows when calculating a student's final course average: homework is worth 35%, the midterms are each worth 20%, and the final exam is worth 25%. Also suppose I set up the course grading scheme as follows when determining final course letter grades:

- A = [90, 100]
- B = [80, 90)
- C = [70, 80)
- D = [60, 70)
- F = below 60

The grades for the students in the class can be found in the *CourseGrades* dataset on Blackboard. Determine the letter grade each student earned in the course. That letter grade should be added as a new variable to the data frame, and you must print the final data frame. Note that everything in this problem is made up.

```
grades <- read.csv("../datasets/CourseGrades.csv")
FinalGrade <- vector()
LetterGrade <- vector()

for (i in 1:nrow(grades)) {
  hwPercent <- grades[i, 2] * 0.35
  midterm1 <- grades[i, 3] * 0.2
  midterm2 <- grades[i, 4] * 0.2
  final <- grades[i, 5] * 0.25
```

```
  total <- hwPercent + midterm1 + midterm2 + final
  FinalGrade[i] <- total
}

for (i in 1:length(FinalGrade)) {
  if (FinalGrade[i] >= 90) {
    LetterGrade[i] = "A"
  } else if (FinalGrade[i] < 90 & FinalGrade[i] >= 80) {
    LetterGrade[i] = "B"
  } else if (FinalGrade[i] < 80 & FinalGrade[i] >= 70) {
    LetterGrade[i] = "C"
  } else if (FinalGrade[i] < 70 & FinalGrade[i] >= 60) {
    LetterGrade[i] = "D"
  } else { #FinalGrade[i] < 60
    LetterGrade[i] = "F"
  }
}

grades$LetterGrade <- LetterGrade
print(grades)
```

```
##     Student HW_Avg Exam1 Exam2 FinalExam LetterGrade
## 1      Bill  79.62    80    77        76           C
## 2   Cameron  86.11    83    83        95           B
## 3      Abby  94.53    94    92        86           A
## 4      Will  78.22    77    76        96           B
## 5     Janet  84.52    85    83        80           B
## 6     Chris  85.79    85    83        81           B
## 7     Laura  89.25    90    85        81           B
## 8  Patricia  83.56    81    80        74           C
## 9    George  96.91    97    92        76           A
## 10   Joseph  84.17    85    80        96           B
## 11  Diamond  87.51    87    84        78           B
## 12     Jill  90.89    87    88        91           B
## 13     T.J.  82.64    82    79        75           C
## 14      Amy  78.76    77    74        68           C
```

6. Suppose a friend of yours is currently a second-semester senior, and her GPAs for the previous (individual) semesters, in order (starting first semester freshman year), are as follows: 4.00, 3.50, 3.67, 3.33, 2.67, 3.42, and 3.56. Assuming she took the same number of credits each semester, write a loop that calculates and stores her cumulative GPA after each semester. Then print the resulting vector of cumulative GPAs.

```
gpas <- c(4.00, 3.50, 3.67, 3.33, 2.67, 3.42, 3.56)
cumulatives <- vector()
```

```r
for (i in 1:length(gpas)) {
  cumulatives[i] <- mean(gpas[1:i])
}
print("Cumulative GPAs:")
```

```
## [1] "Cumulative GPAs:"
```

```r
cumulatives
```

```
## [1] 4.000000 3.750000 3.723333 3.625000 3.434000 3.431667 3.450000
```

7. Returning to problem 6, suppose your friend was on an academic scholarship that requires the student to maintain a cumulative GPA of at least 3.6 in order to continue to receive financial support the following semester. If at any point the student's cumulative GPA drops below 3.6, they lose the scholarship and stop receiving the financial support permanently. Write a while loop that counts the number of semesters your friend received financial support from the scholarship during her college career. Print that number. Additionally, you should assume your friend received the benefits starting first semester freshman year.

```r
semesters = 1
cumulative = cumulatives[1]
while (cumulative >= 3.6) {
  cumulative = cumulatives[semesters + 2]
  semesters = semesters + 1
}
print("The student held the scholarship for this many semesters:")
```

```
## [1] "The student held the scholarship for this many semesters:"
```

```r
print(semesters)
```

```
## [1] 4
```

8. The Fibonacci sequence is a very special sequence that consists of the following numbers: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, etc. If you aren't familiar with this sequence and don't know how each number is found, please search on Google. It's pretty cool! Write code that calculates the first 40 numbers in the sequence, and then print them. Note that you may start with the first two numbers in the sequence only.

```r
fib <- c(0, 1)
for (i in 3:40) {
  fib[i] <- fib[i - 1] + fib[i - 2]
}
print("First 40 of Fibonacci:")
```

```
## [1] "First 40 of Fibonacci:"
```

```
fib
```

```
##  [1]        0        1        1        2        3        5        8       13
##  [9]       21       34       55       89      144      233      377      610
## [17]      987     1597     2584     4181     6765    10946    17711    28657
## [25]    46368    75025   121393   196418   317811   514229   832040  1346269
## [33]  2178309  3524578  5702887  9227465 14930352 24157817 39088169 63245986
```