Stats 20, F23 – Homework 2

Code

JACQUELINE NGUYEN – 606069033

04/17/24

Question 1

Hide

```
mixed1 <- c(TRUE, FALSE, FALSE, TRUE, 4, 0, 3)
mixed1
```

```
## [1] 1 0 0 1 4 0 3
```

Hide

```
mixed2 <- c(TRUE, FALSE, FALSE, TRUE, 4, 0, "3")
mixed2
```

```
## [1] "TRUE"  "FALSE" "FALSE" "TRUE"  "4"     "0"     "3"
```

Hide

```
mixed3 <- c(c(TRUE, FALSE, FALSE, TRUE, 4, 0), "3")
mixed3
```

```
## [1] "1" "0" "0" "1" "4" "0" "3"
```

(a)

mixed1 and mixed2 produced different results because of the way it has been constructed and the mode hierarchy. In mixed2, all of the modes are taken into account at once meaning that since character is the least restrictive, all the items are changed to it. But for the second, we first group logical with numerical and since numerical is less restrictive, the logical items became numerical, then it is combines with characters, making everything in the vector characters.

(b)

Hide

```
mixed4 <- c(c(TRUE, FALSE), c(FALSE, TRUE, 4, 0),  "3")
mixed4
```

```
## [1] "TRUE"  "FALSE" "0"     "1"     "4"     "0"     "3"
```

Question 2

Hide

```
as.numeric(mixed2)
```

```
## Warning: NAs introduced by coercion
```

```
## [1] NA NA NA NA  4  0  3
```

Hide

```
as.numeric(mixed3)
```

```
## [1] 1 0 0 1 4 0 3
```

Hide

```
as.logical(mixed2)
```

```
## [1]  TRUE FALSE FALSE  TRUE    NA    NA    NA
```

Hide

```
as.logical(mixed3)
```

```
## [1] NA NA NA NA NA NA NA
```

Hide

```
# as.logical("waffles")
```

(a)

as.numeric converts character vectors into numeric ones. Since for mixed2 the, there are some logical objects in the list which can not be converted into a character, this yields NA values. Whereas in mixed3, thre logical values are first converted to numeric ones and then converted to characters. So therefore, there are no NA values

(b)

as.logical changes a numerical object to logical values (zero becomes FALSE, all other values become TRUE). Because the last three values in mixed2 are not numerical, it cannot convert it and therefore the values for them get NA. For mixed3, the sub-vector in the vector transforms the logical objects into numerical ones(due to the mode herarchy) and the last object is a character and turns everything else to a character as well- therefore, all the values become NA

(c)

Hide

```
mixed5 <- c(c(TRUE, FALSE, FALSE, TRUE), as.logical(c( 4, 0, as.numeric("3"))))
mixed5
```

```
## [1]  TRUE FALSE FALSE  TRUE  TRUE FALSE  TRUE
```

Question 3

(a)

Hide

```
seq(1,25)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
```

Hide

```
1:25
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
```

Hide

```
seq_len(25)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
```

Hide

```
seq_along(1:25)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
```

(b)

Hide

```
seq(8,2, by=-.5)
```

```
##  [1] 8.0 7.5 7.0 6.5 6.0 5.5 5.0 4.5 4.0 3.5 3.0 2.5 2.0
```

Hide

```
rep(c(-1,0,3,5),5)
```

```
##  [1] -1  0  3  5 -1  0  3  5 -1  0  3  5 -1  0  3  5 -1  0  3  5
```

Hide

```
rep(5:1,5)
```

```
##  [1] 5 4 3 2 1 5 4 3 2 1 5 4 3 2 1 5 4 3 2 1 5 4 3 2 1
```

Hide

```
rep(5:1,c(5,5,5,5,5))
```

```
##  [1] 5 5 5 5 5 4 4 4 4 4 3 3 3 3 3 2 2 2 2 2 1 1 1 1 1
```

Hide

```
rep(10:6,c(1:5))
```

```
##  [1] 10  9  9  8  8  8  7  7  7  7  6  6  6  6  6
```

(c)

this sequence cannot be created like the other ones because it decrements the 10,9,8,7,6 by one everytime it repeats and is followed by repetition the new decremented number sequence. We cannot do this we just seq and repitition because it is not a set operation available given just rep() and seq().

one way to generate the sequence below is like this:

Hide

```
rep(seq(10,6),5) - rep(0:4, rep(5,5))
```

```
##  [1] 10  9  8  7  6  9  8  7  6  5  8  7  6  5  4  7  6  5  4  3  6  5  4  3  2
```

Question 4

Using at most two lines of code, compute sin (kπ), for k = 0, 1, 2, . . . , 100. Explain your result. Does the output match what you would expect mathematically?

Hide

```
k <- 0:100
result <- sin(k * pi)
result
```

```
##   [1]  0.000000e+00  1.224647e-16 -2.449294e-16  3.673940e-16 -4.898587e-16
##   [6]  6.123234e-16 -7.347881e-16  8.572528e-16 -9.797174e-16  1.102182e-15
##  [11] -1.224647e-15  4.899825e-15 -1.469576e-15 -1.960673e-15 -1.714506e-15
##  [16]  5.389684e-15 -1.959435e-15 -1.470814e-15 -2.204364e-15  5.879543e-15
##  [21] -2.449294e-15 -9.809554e-16 -9.799650e-15  6.369401e-15 -2.939152e-15
##  [26] -4.910967e-16  3.921346e-15  6.859260e-15 -3.429011e-15 -1.237961e-18
##  [31] -1.077937e-14  7.349119e-15 -3.918870e-15  4.886208e-16  2.941628e-15
##  [36]  7.838977e-15 -4.408728e-15  9.784795e-16 -1.175909e-14  8.328836e-15
##  [41] -4.898587e-15  1.567919e-14  1.961911e-15  8.818695e-15 -1.959930e-14
##  [46]  1.958197e-15 -1.273880e-14 -4.902301e-15 -5.878305e-15  1.665891e-14
##  [51]  9.821934e-16  9.798412e-15  7.842691e-15  2.937914e-15 -1.371852e-14
##  [56] -3.922584e-15 -6.858022e-15  1.763863e-14  2.475923e-18  1.077813e-14
##  [61] -2.155874e-14  3.917632e-15 -1.469824e-14 -2.942866e-15 -7.837740e-15
##  [66]  1.861835e-14 -9.772415e-16  1.175785e-14  5.883256e-15  4.897349e-15
##  [71] -1.567795e-14 -1.963149e-15 -8.817457e-15  1.959806e-14 -1.956959e-15
##  [76]  1.273756e-14 -2.351817e-14  5.877067e-15 -1.665767e-14 -9.834313e-16
##  [81] -9.797174e-15  2.057778e-14 -3.135839e-14 -1.470443e-14  3.923822e-15
##  [86]  6.856784e-15 -1.763739e-14  2.841800e-14 -3.919860e-14 -6.864212e-15
##  [91] -3.916394e-15  1.469700e-14 -2.547761e-14  3.625821e-14  9.804602e-15
##  [96]  9.760036e-16 -1.175661e-14  2.253721e-14 -3.331782e-14 -1.274499e-14
## [101]  1.964387e-15
```

the answer is supposed to be 0 for each element but here in the code, it gives a series of small numbers extremely close to 0. This occurs because r is not 100% accurate since it can only store so many digits and is based 2.

Question 5

(a)

Hide

```
my_skew <- function(x) {
  n <- length(x)
  num <- sqrt(n) * sum((x-mean(x))^3)
  denom <- sum((x-mean(x))^2)^(3/2)
  num/denom
}
```

(b)

the skew is slightly to the left.

Hide

```
running_times <- c(51, 40, 57, 34, 47, 50, 50, 56, 41, 38)
my_skew(running_times)
```

```
## [1] -0.1615082
```

(c)

Hide

```
my_skew_one <- function(x) {
  n_1 <- length(x)
  num_1 <- sqrt(n_1) * (sum(x^3) - (3*mean(x)) * sum(x^2) + (2*n_1*mean(x)^3))
  denom_1 <- sum(x^2) - (n_1*mean(x)^2)
  num_1/((denom_1)^(3/2))
}

running_times <- c(51, 40, 57, 34, 47, 50, 50, 56, 41, 38)
my_skew_one(running_times)
```

```
## [1] -0.1615082
```

(d)

Hide

```
new_running_times <- running_times * 1e10
my_skew_one(new_running_times)
```

```
## [1] -0.1615082
```

Hide

```
my_skew(new_running_times)
```

```
## [1] -0.1615082
```

Hide

```
my_skew(running_times) - my_skew_one(running_times)
```

```
## [1] -4.15501e-14
```

the answer should not differ because if you mulitply all values in the vector by the same constant, the skew should not change since the distribution of the numbers are the same.

(e)

Hide

```
my_skew(running_times) - my_skew_one(running_times)
```

```
## [1] -4.15501e-14
```

the answer should be 0 but due to rounding errors it is not because of lack of precision.

Question 6

(a)

Hide

```
my_length <- function(x) {
  max(0, seq_along(x))
}
```

(b)

Hide

```
my_length(numeric(0))
```

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

Hide

```
my_length(seq(1, 99, by = 2))
```

```
## [1] 50
```

Hide

```
my_length(c("waffles", "friends", "work"))
```

```
## [1] 3
```

Question 7

(a)

Hide

```
my_rev <- function(x) {
   x[seq(length(x),0)]
  #x[length(x) - seq_along(x) + 1]
}
```

(b)

Hide

```
my_rev(numeric(0))
```

```
## numeric(0)
```

Hide

```
my_rev(seq(1, 99, by = 2))
```

```
##  [1] 99 97 95 93 91 89 87 85 83 81 79 77 75 73 71 69 67 65 63 61 59 57 55 53 51
## [26] 49 47 45 43 41 39 37 35 33 31 29 27 25 23 21 19 17 15 13 11  9  7  5  3  1
```

Hide

```
my_rev(c("waffles", "friends", "work"))
```

```
## [1] "work"    "friends" "waffles"
```

Question 8

(a)

Hide

```
front <- function(x,n) {
  x[0: min(n, length(x))]
}

front(numeric(0),6)
```

```
## numeric(0)
```

Hide

```
front(seq(1, 99, by = 2),6)
```

```
## [1]  1  3  5  7  9 11
```

Hide

```
front(c("waffles", "friends", "work"),6)
```

```
## [1] "waffles" "friends" "work"
```

Hide

```
front(numeric(0),0)
```

```
## numeric(0)
```

Hide

```
front(seq(1, 99, by = 2),0)
```

```
## numeric(0)
```

Hide

```
front(c("waffles", "friends", "work"),0)
```

```
## character(0)
```

(b)

Hide

```
back <- function(x,n) {
  rev_vect <- my_rev(x)
  x <- front(rev_vect, n)
  y <- my_rev(x)
  y

}

back(numeric(0),6)
```

```
## numeric(0)
```

Hide

```
back(seq(1, 99, by = 2),6)
```

```
## [1] 89 91 93 95 97 99
```

Hide

```
back(c("waffles", "friends", "work"),6)
```

```
## [1] "waffles" "friends" "work"
```

Hide

```
back(numeric(0),0)
```

```
## numeric(0)
```

Hide

```
back(seq(1, 99, by = 2),0)
```

```
## numeric(0)
```

Hide

```
back(c("waffles", "friends", "work"),0)
```

```
## character(0)
```

(c)

Hide

```
insert_at <- function(x, values, at) {
  beginning <- front(x,at - 1)
  ending <- back(x, max(0,length(x) + 1 - at))
  new_vect <- c(beginning,values,ending)
  new_vect
}
```

(d)

Hide

```
insert_at(1:5, c(0, 0, 0), at = 1)
```

```
## [1] 0 0 0 1 2 3 4 5
```

Hide

```
insert_at(1:5, c(0, 0, 0), at = 3)
```

```
## [1] 1 2 0 0 0 3 4 5
```

Hide

```
insert_at(1:5, c(0, 0, 0), at = 5)
```

```
## [1] 1 2 3 4 0 0 0 5
```

Hide

```
insert_at(1:5, c(0, 0, 0), at = 6)
```

```
## [1] 1 2 3 4 5 0 0 0
```

Hide

```
insert_at(1:5, c(0, 0, 0), at = 9)
```

```
## [1] 1 2 3 4 5 0 0 0
```