Problem\_Set\_1

Kristopher C. Toll

January 26, 2018

# R Programing exercises

### 1. Calculate the square root of 729

b = sqrt(729)  
print(b)

## [1] 27

### 2. Create a new variable a with value 1947.0

a = as.integer(1947.0)  
print(a)

## [1] 1947

### 3. Create a vector b contaning number form 1 to 6 and find out it's class.

b is a numeric variable.

b = seq(1, 6, by = 1)  
b.1 <- class(b)  
print(c(b, b.1))

## [1] "1" "2" "3" "4" "5" "6" "numeric"

### 4. Create a vector c containg following mixed elements

c = c(1, "a", 2, "b")  
print(c)

## [1] "1" "a" "2" "b"

1. Find out its class. It is a character variable

class(c)

## [1] "character"

1. Get the length of the vector. The length is four

length(c) # Figuring out the length of c

## [1] 4

1. Get the 2nd and 3rd elements, which is "a" and "2".

print(c[2]) # Printing the 2nd element

## [1] "a"

print(c[3]) # Printing the 3rd element

## [1] "2"

### 5. Create a vector d containing following elements c(1, 2, NA, 4, 5, 6, NA, NA, NA, 10)

Remove missing values from d

d = c(1, 2, NA, 4, 5, 6, NA, NA, NA, 10)  
d = as.numeric(na.omit(d)) # Removing NA valuse and converting the vector to a numeric one  
print(d)

## [1] 1 2 4 5 6 10

### 6. Create a vector of values of cos 3 at x = 3, 3.1, 3.2, ... 6

x = seq(3, 6, by = 0.1)  
x.1 = exp(x) \* cos(x) # cosine is reading in x as a radian unit  
print(x.1)

## [1] -19.884531 -22.178753 -24.490697 -26.773182 -28.969238 -31.011186  
## [7] -32.819775 -34.303360 -35.357194 -35.862834 -35.687732 -34.685042  
## [13] -32.693695 -29.538816 -25.032529 -18.975233 -11.157417 -1.362099  
## [19] 10.632038 25.046705 42.099201 61.996630 84.929067 111.061586  
## [25] 140.525075 173.405776 209.733494 249.468441 292.486707 338.564378  
## [31] 387.360340

#x.2 = exp(x) \* cos(x/180) # this funtions will read it in as degrees  
#print(x.2)

### 7. Calculate

s = seq(10, 100, by = 1) # Creating the sequence  
s.1 <- s^3 + 4 \* s^2 # transforming the sequence  
s.2 = sum(s.1) # Calculating the sum of the transformed sequence  
print(s.2)

## [1] 26852735

### 8. Execute the following line which create two vectors of random integers that are chosen with replacement from the integers 0, 1, ...999. Both vectors have length 250.

x <- sample(0:999, size = 250, replace = TRUE)  
y <- sample(0:999, size = 250, replace = TRUE)

1. Pick out the values in Y which are > 600

y.1 <- subset(y, y>600) # use the subset comand to pull out elements of a vector  
print(y.1)

## [1] 847 720 635 811 995 690 773 977 614 963 950 654 977 838 685 764 690  
## [18] 889 836 948 833 969 982 791 847 879 641 975 685 815 876 865 770 819  
## [35] 934 642 690 988 835 618 850 731 933 931 622 844 645 817 726 840 734  
## [52] 726 868 830 650 913 863 662 912 806 983 626 775 716 631 901 786 755  
## [69] 932 686 722 755 997 853 619 738 745 841 769 607 632 921 781 983 985  
## [86] 969 707 914 614 805 787 794 708 607 835 793 801 820

1. How many values in y are within 200 of the maximum value of the terms in y?

y.2 <- subset(y, y >= max(y)-200)  
print(length(y.2))

## [1] 51

1. Create the vector e

e <- abs(x-mean(x))^(1/2)  
print(e)

## [1] 7.926159 14.656876 12.213763 18.133505 17.715078 15.039149 17.199535  
## [8] 22.004000 21.373441 16.426320 22.117324 20.659719 21.334854 7.669681  
## [15] 14.006284 14.323966 22.409284 9.788973 21.683542 21.767499 15.900440  
## [22] 12.199344 16.253492 16.577817 10.140217 11.697179 9.755819 12.335964  
## [29] 3.977939 4.563332 21.019610 13.993713 19.446748 16.678609 17.228581  
## [36] 17.170440 21.051746 10.008796 11.908988 16.088008 19.574882 18.225696  
## [43] 21.358277 21.606110 17.939454 10.778497 15.356562 21.591109 6.842806  
## [50] 16.426320 12.916037 16.284225 17.949262 3.343052 19.230809 19.282738  
## [57] 22.409284 19.778170 13.957937 4.709140 13.644633 19.004631 6.230891  
## [64] 21.867236 16.191850 16.698024 18.470950 15.529842 7.336484 13.813906  
## [71] 12.090327 19.421020 21.775583 20.562685 14.860148 14.939076 7.268012  
## [78] 5.493269 16.056899 21.544744 17.743280 16.757804 16.376080 11.626521  
## [85] 21.521524 9.686279 19.489074 17.640181 11.481463 21.683542 19.004631  
## [92] 11.409470 21.287931 10.206664 21.090851 12.497040 16.067856 18.088007  
## [99] 17.257578 9.530163 17.753197 21.381674 19.945526 17.601818 7.669681  
## [106] 15.204473 19.360372 20.900335 6.721309 17.034553 10.808515 4.563332  
## [113] 15.453932 9.686279 22.386960 17.411950 7.012560 9.958715 14.939076  
## [120] 17.459210 14.416102 7.153740 11.217130 14.450744 18.133505 9.174748  
## [127] 8.990217 18.105911 16.160941 15.368019 9.496104 21.981265 13.447081  
## [134] 8.707698 20.029378 22.207566 18.686252 12.616497 18.846114 19.794545  
## [141] 12.576804 16.191850 20.659719 18.889786 17.024218 18.632659 19.021672  
## [148] 21.067131 12.158289 19.954348 18.461419 15.721832 13.970540 19.100366  
## [155] 8.650087 20.079243 14.416102 20.812881 20.683907 19.308651 3.489413  
## [162] 19.702183 4.452415 7.268012 16.376080 14.554175 18.270851 11.409470  
## [169] 11.437832 19.213953 20.416268 20.302118 5.729223 19.727544 19.565889  
## [176] 21.790457 16.847077 8.650087 18.578913 21.698479 21.973256 17.024218  
## [183] 12.034284 10.352584 14.393610 19.161837 4.144394 16.757804 14.183653  
## [190] 15.335449 13.970540 20.948126 19.995600 9.941026 16.916737 21.138212  
## [197] 12.172756 16.426320 7.129095 8.010992 5.179189 17.995110 4.379041  
## [204] 13.970540 18.596344 12.954690 11.053325 19.844999 21.767499 20.900335  
## [211] 15.900440 18.596344 8.954105 12.536985 19.021672 15.039149 13.348258  
## [218] 17.315427 12.890927 10.591317 17.209765 8.256876 16.426320 13.460163  
## [225] 17.286527 1.350555 9.371446 16.222700 18.942650 21.466812 6.097212  
## [232] 21.279662 21.373441 13.826641 12.482948 8.954105 19.437695 16.528037  
## [239] 21.890089 3.438604 21.051746 6.259073 16.222700 9.857789 17.640181  
## [246] 18.605806 7.988992 6.570845 15.039149 15.900440

1. Create another vector

n=2:length(y)  
d <- (y[n]-x[n-1]) # It worked!  
print(d)

## [1] 417 -229 -409 -78 541 -84 614 -943 294 772 -292 451 -557 -119  
## [15] 84 -600 -555 -231 14 538 374 619 193 327 170 -180 66 332  
## [29] 43 -226 384 -105 -601 67 489 566 -846 -440 339 -82 -804 64  
## [43] -582 810 -74 -225 -155 -11 -89 610 227 211 167 287 409 449  
## [57] -148 -5 343 -457 296 -169 361 -508 -742 314 -757 -368 437 -117  
## [71] -633 -582 -535 140 593 -240 -321 -155 93 -920 -40 558 58 306  
## [85] -314 -118 316 -395 112 -683 -687 -603 -256 -414 238 -504 237 -353  
## [99] 640 59 -321 -836 490 5 132 -106 389 389 -515 67 -238 -137  
## [113] 477 534 -705 -552 -534 -539 -676 743 315 -270 255 -139 41 -11  
## [127] 432 -55 -272 -84 -536 -624 9 -213 -868 -605 -701 165 -122 -95  
## [141] 189 -21 68 590 192 -389 458 54 523 -818 409 90 -428 522  
## [155] -151 17 64 -918 227 743 157 31 333 -124 -588 306 824 -423  
## [169] 264 -325 237 -130 40 -418 606 23 6 -120 -207 879 776 19  
## [183] -90 155 232 -358 70 474 28 27 309 -50 213 459 -363 527  
## [197] -22 515 -307 -423 47 576 329 81 -73 127 -312 -80 588 576  
## [211] 681 634 -4 333 -78 266 -664 776 -365 102 -350 353 -104 -60  
## [225] 303 314 -370 -213 653 333 264 212 394 -268 203 135 492 -305  
## [239] -137 48 -680 -245 -262 203 -655 -38 391 -437 -255

### 9. In this exercise, we will consider a quadratic equations of the form . Create a vector of coefficients for a quadratic equations.

coeffs <- sample(-20:20, size = 3, replace = TRUE)

1. Determine the length of the object coeffs.

print(length(coeffs))

## [1] 3

1. Create 200 values of x from a regularly spaced vector between -3 and 3

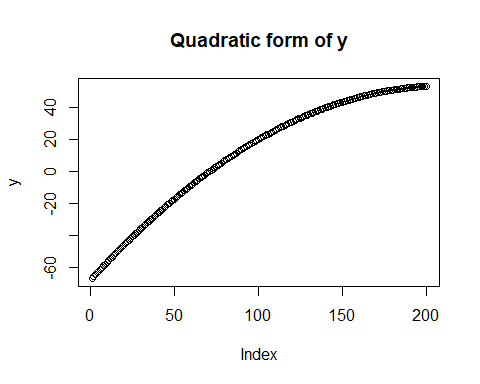
x <- seq(from = -3, to = 3, length.out = 200)

1. Now obtain the value of the quadratic function (y) at each value of x

y = coeffs[1] + coeffs[2] \* x + coeffs[3] \* x^2

1. Construct the plot

plot(y, main = "Quadratic form of y")



### 10. Without using R, determine the result of the following computation

### 11. Create the following matrix with 15 rows

A = matrix(c(rep(c(10, -5, 10), times = 15)), nrow = 15, byrow = TRUE)  
print(A)

## [,1] [,2] [,3]  
## [1,] 10 -5 10  
## [2,] 10 -5 10  
## [3,] 10 -5 10  
## [4,] 10 -5 10  
## [5,] 10 -5 10  
## [6,] 10 -5 10  
## [7,] 10 -5 10  
## [8,] 10 -5 10  
## [9,] 10 -5 10  
## [10,] 10 -5 10  
## [11,] 10 -5 10  
## [12,] 10 -5 10  
## [13,] 10 -5 10  
## [14,] 10 -5 10  
## [15,] 10 -5 10

A.1 = A # Copy the matrix  
A.1[,3] = A.1[,1] + A.1[,2] # rewrite the 3 column as a sum of the first two  
print(A.1)

## [,1] [,2] [,3]  
## [1,] 10 -5 5  
## [2,] 10 -5 5  
## [3,] 10 -5 5  
## [4,] 10 -5 5  
## [5,] 10 -5 5  
## [6,] 10 -5 5  
## [7,] 10 -5 5  
## [8,] 10 -5 5  
## [9,] 10 -5 5  
## [10,] 10 -5 5  
## [11,] 10 -5 5  
## [12,] 10 -5 5  
## [13,] 10 -5 5  
## [14,] 10 -5 5  
## [15,] 10 -5 5

### 12. Create a function that given two number will return he sum of those two number

add <- function(a,b){  
 c = a + b  
 return(c)  
}  
  
add(5,10)

## [1] 15

### 13. Create a function that given a vector and an integer will return how many times the integer appears inside the vector

count <- function(x, int){  
 y <- vector()  
 for(i in 1:length(x)){  
 ifelse(x[i] == int, y[i] <- 1, y[i] <- 0)  
 }  
 z = sum(y)  
 return(z)  
}  
  
# testing the function  
  
x <- c(4,5, 6, 6, 7, 8)   
count(x = x, int = 6) # The argument should return a 2 given vector x

## [1] 2

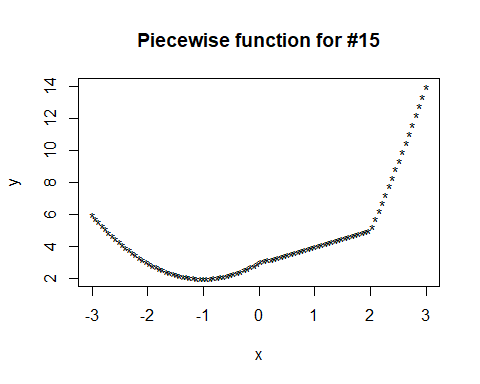
### 14. Create a function that given an integer vector will return

zsquared <- function(x){  
 z <- numeric(length(x))  
 for(i in 1:length(x)){  
 z[i] <- x[i]^i  
 }  
 return(z)  
}  
  
# Testing the function  
  
x <- c(2, 2, 3)  
zsquared(x) # should return two 4s given vector x

## [1] 2 4 27

### 15. Create a piecewise function

piecewise <- function(x){  
 y <- numeric(length(x))  
 for(i in 1:length(x)){  
 if(x[i] < 0 ){  
 y[i] = (x[i]^2 + 2 \* x[i] + 3)  
 }  
 else if(x[i] >= 0 & x[i] < 2 ){  
 y[i] = (x[i] + 3)  
 }  
 else if( 2 <= x[i]){  
 y[i] = (x[i]^2 + 4 \* x[i] - 7)  
 }  
 }  
 return(y)  
}  
  
# Testing the piecewise function  
  
x <-seq(-3, 3, length = 100)  
y <- piecewise(x)  
  
plot(x, y, main = "Piecewise function for #15", pch = "\*", col = 617, bg = 456)



# Theory

### Problem 1:

Show that for , with and

### Problem 2:

Show that the autocovariance function can be written as where and

### Problem 3:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| time, t | Yt | Yt-1 | Yt-3 | µ̂Y(t) | Yt - µ̂Y(t) | (Yt - µ̂Y(t))2 | Yt-1 - µ̂Y(t) | ( Yt-1 - µ̂Y(t))2 | ( Yt - µ̂Y(t))( Yt-1 - µ̂Y(t)) |
| Jan-49 | 112.00 |  |  | 126.67 | -14.67 | 215.11 |  |  |  |
| Feb-49 | 118.00 | 112.00 |  | 126.67 | -8.67 | 75.11 | -14.67 | 215.11 | 127.11 |
| Mar-49 | 132.00 | 118.00 | 112.00 | 126.67 | 5.33 | 28.44 | -8.67 | 75.11 | -46.22 |
| Apr-49 | 129.00 | 132.00 | 118.00 | 126.67 | 2.33 | 5.44 | 5.33 | 28.44 | 12.44 |
| May-49 | 121.00 | 129.00 | 132.00 | 126.67 | -5.67 | 32.11 | 2.33 | 5.44 | -13.22 |
| Jun-49 | 135.00 | 121.00 | 129.00 | 126.67 | 8.33 | 69.44 | -5.67 | 32.11 | -47.22 |
| Jul-49 | 148.00 | 135.00 | 121.00 | 126.67 | 21.33 | 455.11 | 8.33 | 69.44 | 177.78 |
| Aug-49 | 148.00 | 148.00 | 135.00 | 126.67 | 21.33 | 455.11 | 21.33 | 455.11 | 455.11 |
| Sep-49 | 136.00 | 148.00 | 148.00 | 126.67 | 9.33 | 87.11 | 21.33 | 455.11 | 199.11 |
| Oct-49 | 119.00 | 136.00 | 148.00 | 126.67 | -7.67 | 58.78 | 9.33 | 87.11 | -71.56 |
| Nov-49 | 104.00 | 119.00 | 136.00 | 126.67 | -22.67 | 513.78 | -7.67 | 58.78 | 173.78 |
| Dec-49 | 118.00 | 104.00 | 119.00 | 126.67 | -8.67 | 75.11 | -22.67 | 513.78 | 196.44 |
| sum | 1520.00 | 1402.00 | 1298.00 | 1520.00 | 0.00 | 2070.67 | 8.67 | 1995.56 | 1163.56 |

Sample Variance: = 2070.67/(12-1) = 188.24

Sample autocovariance: (1/11)\*1163.56 = 105.78

Sample Autocorrelation: 1163.56/ sqrt(2070.67\*1995.56) = 0.57