Answers for Assignment 14.1

1. Create the vectors

(a) (2, 3, ... , 29, 30)

(b) (30, 29, ... , 2)

(c) (1, 2, 3, .... , 29, 30, 29, 28, , 2, 1)

(d) (4, 6, 3) and assign it to the name dev.

Answer 1.

1. To create a vector with name v.

v<-c(2:30)

v

Output -

[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 26

[26] 27 28 29 30

Answer 2

1. To create a vector with name v1.

V1<-c(30:2)

V1

Output is –

[1] 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6

[26] 5 4 3 2

Answer 3

( c) To create a vector with name v2 (1, 2, 3, …. , 29, 30, 29, 28, , 2, 1)

Answer 3

V2<-c(1:30,29:1)

V2

Output is –

[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

[26] 26 27 28 29 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10

[51] 9 8 7 6 5 4 3 2 1

Answer 4

d) (4, 6, 3) and assign it to the name dev.

To create a vector with name dev

Dev<-c(4,6,3)

Dev

Output is -- [1] 4 6 3

e) (5, 6, 7, 5, 6, 7, , 5, 6, 7) where there are 10 occurrences of 5.

For parts (e), (f) and (g)

e) (5, 6, 7, 5, 6, 7, , 5, 6, 7) where there are 10 occurrences of 5.

Answer) repe <- rep(c(5, 6, 7),10)

Repe

Output is ) [1] 5 6 7 5 6 7 5 6 7 5 6 7 5 6 7 5 6 7 5 6 7 5 6 7 5 6 7 5 6 7

(f) (5, 6, 7, 5, 6, 7, , 5, 6, 7, 5) where there are 11 occurrences of 5, 10 occurrences of 6

and 10 occurrences of 7.

Answer ) v1f <- c(rep(c(5, 6, 7 ),10), 5)

v1f

Output ) [1] 5 6 7 5 6 7 5 6 7 5 6 7 5 6 7 5 6 7 5 6 7 5 6 7 5 6 7 5 6 7 5

(g) (4, 4, , 4, 6, 6, , 6, 3, 3, , 3) where there are 10 occurrences of 4, 20 occurrences

of 6 and 30 occurrences of 3.

Output is ) v1g <- c(rep(4,10),rep(6,20),rep(3,30))

V1g

Output is ) [1] 4 4 4 4 4 4 4 4 4 4 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 3 3 3 3 3 3 3 3

[39] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3

2. Create a vector of the values of e

X

sin(x) at x = 3, 3.1, 3.2, , 6.

Answer) x <- seq(3,6,by = 0.1)

v2 <- exp(x) \* cos(x)

x

v2

Output is) [1] 3.0 3.1 3.2 3.3 3.4 3.5 3.6 3.7 3.8 3.9 4.0 4.1 4.2 4.3 4.4 4.5 4.6 4.7 4.8

[20] 4.9 5.0 5.1 5.2 5.3 5.4 5.5 5.6 5.7 5.8 5.9 6.0

[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

3. Execute the following lines which create two vectors of random integers which are chosen with

replacement from the integers 0, 1, : : : , 999. Both vectors have length 250.

set.seed(100)

x <- Sample (0:999, 250, replace=T)

y <- Sample (0:999, 250, replace=T)

1. Identify out the values in y which are > 500.

Answer) x.rndm <- sample(0:999, 250, replace = TRUE)

y.rndm <- sample(0:999, 250, replace = TRUE)

y\_gt\_500\_indexes <- which(y.rndm >500, arr.ind=TRUE);

y\_gt\_500\_indexes

OUPUT) [1] 1 2 6 7 8 9 11 13 16 17 21 26 27 30 33 37 41 42

[19] 43 49 51 52 54 57 58 60 62 64 66 68 71 72 74 76 77 78

[37] 79 83 84 88 90 91 93 94 95 97 99 101 104 106 107 109 112 114

[55] 117 120 122 124 126 129 130 131 132 133 136 137 142 143 147 148 150 151

[73] 153 157 158 159 162 164 165 166 167 170 172 173 174 177 178 179 180 181

[91] 185 188 189 190 191 193 196 199 205 206 209 210 213 215 216 218 219 220

[109] 221 222 224 225 228 229 230 233 234 239 240 241 243 245 250

1. Identify the index positions in y of the values which are > 700?

Answer ) x.rndm <- sample(0:999, 250, replace = TRUE)

y.rndm <- sample(0:999, 250, replace = TRUE)

y\_gt\_700\_indexes <- which(y.rndm >700, arr.ind=TRUE);

y\_gt\_700\_indexes

Output) [1] 1 3 7 8 10 11 12 23 26 29 33 39 47 50 56 60 65 76 81

[20] 88 91 92 99 103 114 116 117 118 120 121 124 125 126 128 130 131 142 143

[39] 145 150 151 152 153 154 155 158 160 161 162 165 166 168 170 171 173 174 180

[58] 181 182 183 187 188 192 194 195 200 201 205 208 210 216 217 222 227 229 232

[77] 235 239 240 244 248 249

c) What are the values in x which are in Same index position to the values in y which

are > 400?

Answer ) x.rndm <- sample(0:999, 250, replace = TRUE)

y.rndm <- sample(0:999, 250, replace = TRUE)

x <- which(x.rndm > 400, arr.ind=TRUE)

y <- which(y.rndm > 400, arr.ind=TRUE)

p<-intersect(x, y)

p

OUTPUT) [1] 6 9 12 15 18 23 30 35 37 39 40 41 42 43 44 49 56 62 64

[20] 65 66 69 70 72 75 77 79 82 83 89 99 100 101 104 106 108 112 116

[39] 117 118 120 124 130 139 142 143 145 146 148 150 154 156 158 159 160 163 164

[58] 165 167 169 171 172 176 178 179 180 181 183 187 188 189 190 191 194 195 197

[77] 199 200 204 207 209 210 213 215 219 228 229 230 231 232 233 234 235 242 249

[96] 250

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

Answer) x.rndm <- sample(0:999, 250, replace = TRUE)

y.rndm <- sample(0:999, 250, replace = TRUE)

y\_gt\_200\_indexes <- which(y.rndm <=200, arr.ind=TRUE);

y\_gt\_200\_indexes

Output) [1] 7 13 21 24 35 36 41 43 46 50 56 58 60 69 76 83 88 93 102

[20] 112 119 131 136 138 139 140 143 144 145 154 169 174 175 177 178 183 209 212

[39] 220 232 233 239 248 250

1. How many numbers in x are divisible by 2?

Answer) x.rndm <- sample(0:999, 250, replace = TRUE)

y.rndm <- sample(0:999, 250, replace = TRUE)

x <- which((x.rndm%%2==0), arr.ind=TRUE);

x

Ouput) [1] 2 3 8 13 15 17 18 21 22 23 25 26 28 32 35 36 38 40

[19] 41 44 46 47 48 49 53 54 56 63 65 68 70 72 73 74 80 81

[37] 83 84 85 87 90 92 94 95 97 98 101 103 104 108 111 113 114 115

[55] 118 122 123 125 126 128 130 131 133 134 143 144 146 147 148 149 150 152

[73] 157 159 169 170 171 173 174 175 176 177 180 181 184 187 189 193 194 195

[91] 196 198 203 204 205 208 210 211 214 215 216 218 221 222 224 225 227 228

[109] 229 230 231 235 237 238 240 241 243 244 245 246 247 250

1. Sort the numbers in the vector x in the order of increasing values in y.

Answer) x.rndm <- sample(0:999, 250, replace = TRUE)

y.rndm <- sample(0:999, 250, replace = TRUE)

p <- sort(order(y.rndm)[x.rndm])

p

Output is) [1] 16 22 22 23 24 25 28 29 30 34 39 39 40 41 42 43 50 52 53

[20] 54 69 81 82 91 94 97 98 102 103 111 112 121 125 125 138 146 154 154

[39] 157 160 167 179 197 200 201 204 205 208 215 216 216 222 229 245

g) Create the vector (x1 + 2x2 - x3; x2 + 2x3 -x4 ,, xn−2 + 2xn−1 - xn).

Answer) x.rndm <- sample(0:999,250, replace = F, prob = NULL)

y.rndm <- sample(0:999,250, replace = F, prob = NULL)

v6d <- c( x.rndm[1:248] + 2 \* x.rndm[2:249] - x.rndm[3:250])

v6d

Output is )

[1] 441 1855 886 915 1031 927 2290 1295 740 0 -344 1293 2381 1006 1393

[16] 622 544 1740 1402 1761 403 764 -333 1234 1803 2016 -44 1716 636 1424

[31] 1085 1404 63 743 1797 2166 544 -287 756 1993 1965 1342 341 -565 1713

[46] 1667 335 -133 1296 1917 312 -27 2087 1045 1498 97 1534 651 1358 542

[61] 579 1535 465 1908 1854 325 302 33 960 -156 1076 939 919 709 620

[76] 1283 1398 632 1621 1347 -602 989 2011 1174 2111 359 1172 501 837 1389

[91] 1238 595 164 101 2009 433 580 677 938 2066 175 1540 172 1397 1504

[106] 1374 -26 1666 -68 1168 1715 1864 1324 117 789 2044 920 -371 1616 2027

[121] -212 1336 758 757 317 1598 1182 60 -437 1331 1635 263 1137 1279 691

[136] -284 1285 -193 1810 610 690 903 -494 1515 859 2102 805 1437 1564 677

[151] 1111 578 -595 732 1763 1724 2523 243 1576 1167 464 -233 814 1149 196

[166] 541 632 -334 1352 1371 1964 1732 1601 1300 1282 652 -746 1145 1552 1797

[181] 2295 1011 774 1410 277 687 1855 1770 139 1510 2205 99 752 2462 678

[196] 1742 1463 280 505 -267 1505 705 1181 2026 587 -19 586 519 1075 958

[211] 1340 898 684 119 1157 1259 1229 1783 335 753 978 1543 934 1136 2060

[226] 322 1011 399 156 2150 633 1009 881 434 197 1449 1672 1199 129 828

[241] 935 1917 1022 -138 1115 1413 1475 2231

h) Calculate: n-1 Σ (e−xi+10/ xi+ 10) i=1

Answer) x.rndm <- sample(0:999,250, replace = T)

n<-c(1:10)

i<-c(1:10)

v6c <- n-1\*((exp(-x.rndm[1:250]\*i+10) / (x.rndm[1:250]\*i+10)))

v6c

Output is) [1] 1.000000 2.000000 3.000000 4.000000 5.000000

[6] 6.000000 7.000000 8.000000 9.000000 10.000000

[11] 1.000000 2.000000 3.000000 4.000000 5.000000

[16] 6.000000 7.000000 8.000000 9.000000 10.000000

[21] 1.000000 2.000000 3.000000 4.000000 5.000000

[26] 6.000000 7.000000 8.000000 9.000000 10.000000

[31] 1.000000 2.000000 3.000000 4.000000 5.000000

[36] 6.000000 7.000000 8.000000 9.000000 10.000000

[41] 1.000000 2.000000 3.000000 4.000000 5.000000

[46] 6.000000 7.000000 8.000000 9.000000 10.000000

[51] 1.000000 2.000000 3.000000 4.000000 5.000000

[56] 6.000000 7.000000 8.000000 9.000000 10.000000

[61] 1.000000 2.000000 3.000000 4.000000 5.000000

[66] 6.000000 7.000000 8.000000 9.000000 10.000000

[71] 1.000000 2.000000 3.000000 4.000000 5.000000

[76] 6.000000 7.000000 8.000000 9.000000 10.000000

[81] 1.000000 2.000000 3.000000 4.000000 5.000000

[86] 6.000000 7.000000 8.000000 9.000000 10.000000

[91] 1.000000 2.000000 3.000000 4.000000 5.000000

[96] 6.000000 7.000000 8.000000 9.000000 10.000000

[101] 1.000000 2.000000 3.000000 4.000000 5.000000

[106] 6.000000 7.000000 8.000000 9.000000 10.000000

[111] 1.000000 2.000000 3.000000 4.000000 5.000000

[116] 6.000000 7.000000 8.000000 9.000000 10.000000

[121] 1.000000 2.000000 3.000000 4.000000 5.000000

[126] 6.000000 7.000000 8.000000 9.000000 10.000000

[131] 1.000000 2.000000 3.000000 4.000000 5.000000

[136] 6.000000 7.000000 8.000000 9.000000 10.000000

[141] 1.000000 2.000000 3.000000 4.000000 5.000000

[146] 6.000000 7.000000 8.000000 9.000000 10.000000

[151] 1.000000 2.000000 3.000000 4.000000 5.000000

[156] 6.000000 7.000000 8.000000 9.000000 10.000000

[161] 1.000000 2.000000 3.000000 4.000000 5.000000

[166] 6.000000 7.000000 8.000000 9.000000 10.000000

[171] 1.000000 2.000000 3.000000 4.000000 5.000000

[176] 6.000000 7.000000 8.000000 9.000000 10.000000

[181] 1.000000 2.000000 3.000000 4.000000 5.000000

[186] 6.000000 7.000000 -2194.646579 9.000000 10.000000

[191] 1.000000 2.000000 3.000000 4.000000 5.000000

[196] 6.000000 7.000000 8.000000 9.000000 10.000000

[201] 1.000000 2.000000 3.000000 4.000000 5.000000

[206] 6.000000 7.000000 8.000000 9.000000 10.000000

[211] 1.000000 2.000000 3.000000 4.000000 5.000000

[216] 6.000000 7.000000 8.000000 9.000000 10.000000

[221] 1.000000 2.000000 3.000000 3.589497 5.000000

[226] 6.000000 7.000000 8.000000 9.000000 10.000000

[231] 1.000000 2.000000 3.000000 4.000000 5.000000

[236] 6.000000 7.000000 8.000000 9.000000 10.000000

[241] 1.000000 2.000000 3.000000 4.000000 5.000000

[246] 6.000000 7.000000 8.000000 9.000000 10.000000