

```
1: > #This script was used to do the problems on
2: > #   a particular worksheet7
3: > #####
4: > ## C A S E:  1  ##
5: > #####
6: > n_1 <- 112 # the sample size
7: > x_1 <- 71  # the number of 'good' items
8: > c_level <- 0.8950 # set the confidence level
9: >
10: > p <- x_1/n_1 # get the proportion
11: > p
12: [1] 0.6339286
13: > #       compute the standard deviation estimate
14: > stddev <- sqrt( p*(1-p)/n_1 )
15: > stddev
16: [1] 0.04551914
17: > #   using this as an approximately normal distribution
18: > #   we can use qnorm to get a z value
19: > z <- qnorm( (1-c_level)/2, lower.tail=FALSE)
20: > z
21: [1] 1.621082
22: > moe <- z*stddev # compute the margin of error
23: > moe
24: [1] 0.07379027
25: > # make sure that the function is loaded
26: > source("../ci_prop.R")
27: > #   run the function
28: > ci_prop( n_1, x_1, c_level)
29:   lower      upper      p hat      z-score      p hat sd
30: 0.56013830 0.70771884 0.63392857 1.62108225 0.04551914
31: >
32: > #####
33: > ## C A S E:  2  ##
34: > #####
35: > n_1 <- 107 # the sample size
36: > x_1 <- 44 # the number of 'good' items
37: > c_level <- 0.9950
38: >
39: > p <- x_1/n_1 # get the proportion
40: > p
41: [1] 0.411215
42: > #       compute the standard deviation estimate
43: > stddev <- sqrt( p*(1-p)/n_1 )
44: > stddev
45: [1] 0.04756866
46: > #   using this as an approximately normal distribution
47: > #   we can use qnorm to get a z value
48: > z <- qnorm( (1-c_level)/2, lower.tail=FALSE)
49: > z
50: [1] 2.807034
51: > moe <- z*stddev # compute the margin of error
52: > moe
53: [1] 0.1335268
54: > #   run the function
55: > ci_prop( n_1, x_1, c_level)
56:   lower      upper      p hat      z-score      p hat sd
57: 0.27768811 0.54474180 0.41121495 2.80703377 0.04756866
58: >
59: >
60: > #####
```

```

61: > ## C A S E:  3  ##
62: > #####
63: > n_1 <- 114 # the sample size
64: > x_1 <- 90 # the number of 'good' items
65: > c_level <- 0.9800
66: >
67: > p <- x_1/n_1 # get the proportion
68: > p
69: [1] 0.7894737
70: > # compute the standard deviation estimate
71: > stddev <- sqrt( p*(1-p)/n_1 )
72: > stddev
73: [1] 0.03818296
74: > # using this as an approximately normal distribution
75: > # we can use qnorm to get a z value
76: > z <- qnorm( (1-c_level)/2, lower.tail=FALSE)
77: > z
78: [1] 2.326348
79: > moe <- z*stddev # compute the margin of error
80: > moe
81: [1] 0.08882685
82: > # run the function
83: > ci_prop( n_1, x_1, c_level)
84: lower upper p hat z-score p hat sd
85: 0.70064684 0.87830053 0.78947368 2.32634787 0.03818296
86: >
87: >
88: > #####
89: > ## C A S E:  4  ##
90: > #####
91: > #
92: > # generate the values
93: > source("../gnrnd4.R")
94: > gnrnd4(373788907,4768986)
95: style= 7 size= 90 seed= 37378 num digits= 0 alt_sign= 1
96: 8 9 8 6 7 4
97: 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 4 4 4 4 4 4 5 5 5 5 5 5 5 6
6 6 6
98: [1] "DONE "
99: > # verify the values
100: > head( L1 )
101: [1] 2 3 2 6 1 2
102: > tail( L1 )
103: [1] 1 2 3 3 4 1
104: > n_1 <- length( L1 ) # get the number of values
105: > n_1
106: [1] 90
107: > hold_table <- table( L1 ) # how many of each
108: > hold_table
109: L1
110: 1 2 3 4 5 6
111: 18 19 22 10 8 13
112: > pick <- 1 # set the item that is of interest
113: > # now find how many of those items we have
114: > x_1 <- as.numeric( hold_table[ pick ])
115: > x_1
116: [1] 18
117: >
118: > c_level <- 0.9850 # set the confidence level
119: >

```

```
120: > p <- x_1/n_1 # get the proportion
121: > p
122: [1] 0.2
123: > stddev <- sqrt( p*(1-p)/n_1 ) # get the std. dev.
124: > stddev
125: [1] 0.0421637
126: > z <- qnorm( (1-c_level)/2, lower.tail=FALSE)
127: > z
128: [1] 2.432379
129: > moe <- z*stddev
130: > moe
131: [1] 0.1025581
132: > # run the function
133: > ci_prop( n_1, x_1, c_level)
134:      lower      upper      p hat      z-score      p hat sd
135: 0.09744189 0.30255811 0.20000000 2.43237906 0.04216370
136: >
137: >
138: > #####
139: > ## C A S E: 5 ##
140: > #####
141: > #
142: > # generate the values
143: > source("../gnrnd4.R")
144: > gnrnd4(187659407,895695)
145: style= 7 size= 95 seed= 18765 num digits= 0 alt_sign= 1
146: 9 6 5 9 8
147: 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 3 3 3 3 3 4 4 4 4 4 4 4 4 4 5 5 5 5 5 5 5 5
148: [1] "DONE "
149: > # verify the values
150: > head( L1 )
151: [1] 1 2 5 5 3 5
152: > tail( L1 )
153: [1] 4 3 3 4 5 1
154: > n_1 <- length( L1 ) # get the number of values
155: > n_1
156: [1] 95
157: > hold_table <- table( L1 ) # how many of each
158: > hold_table
159: L1
160: 1 2 3 4 5
161: 18 11 15 20 31
162: > pick <- 1 # set the item that is of interest
163: > # now find how many of those items we have
164: > x_1 <- as.numeric( hold_table[ pick ])
165: > x_1
166: [1] 18
167: >
168: > c_level <- 0.9225 # set the confidence level
169: >
170: > p <- x_1/n_1 # get the proportion
171: > p
172: [1] 0.1894737
173: > stddev <- sqrt( p*(1-p)/n_1 ) # get the std. dev.
174: > stddev
175: [1] 0.04020649
176: > z <- qnorm( (1-c_level)/2, lower.tail=FALSE)
177: > z
178: [1] 1.76538
179: > moe <- z*stddev
```

```
180: > moe
181: [1] 0.07097972
182: > # run the function
183: > ci_prop( n_1, x_1, c_level)
184:      lower      upper      p hat      z-score      p hat sd
185: 0.11849396 0.26045341 0.18947368 1.76537954 0.04020649
186: >
```