Practical 1

Download the raw document here.

Today you will be creating and manipulating vectors, and data frames to uncover a top secret message.

Part One: Setup

Each of the following R chunks will cause an error and/or do the desired task incorrectly. Find the mistake, and correct it to complete the intended action.

1. Create vectors containing the upper case letters, lower case letters, and some punctuation marks.

```
lower_case <- c("a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o", "
upper_case <- c("A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "


punctuation <- c(".", ",", "!", "?", "'", "\"", "(", ")", " ", ":", ";", "-")
```

2. Make one long vector containing all the symbols.

```
my_symbols <- c(lower_case, upper_case, punctuation)
```

3. Turn the `my_symbols` vector into a data frame, with the variable name "Symbol"

```
my_symbols <- data.frame(Symbol = my_symbols)

my_symbols
```

```
   Symbol
1       a
2       b
3       c
4       d
5       e
6       f
7       g
8       h
9       i
```

```
10      j
11      k
12      l
13      m
14      n
15      o
16      p
17      q
18      r
19      s
20      t
21      u
22      v
23      w
24      x
25      y
26      z
27      A
28      B
29      C
30      D
31      E
32      F
33      G
34      H
35      I
36      J
37      K
38      L
39      M
40      N
41      O
42      P
43      Q
44      R
45      S
46      T
47      U
48      V
49      W
50      X
51      Y
52      Z
```

```
53      .
54      ,
55      !
56      ?
57      '
58      "
59      (
60      )
61
62      :
63      ;
64      -
```

4. Find the total number of symbols we have in our data frame.

```
len <- length(my_symbols$Symbol)
```

5. Create a new variable in your dataframe that assigns a number to each symbol.

```
my_symbols$Num <- 1:len
my_symbols
```

```
   Symbol Num
1       a   1
2       b   2
3       c   3
4       d   4
5       e   5
6       f   6
7       g   7
8       h   8
9       i   9
10      j  10
11      k  11
12      l  12
13      m  13
14      n  14
15      o  15
16      p  16
17      q  17
18      r  18
19      s  19
```

20	t	20
21	u	21
22	v	22
23	w	23
24	x	24
25	y	25
26	z	26
27	A	27
28	B	28
29	C	29
30	D	30
31	E	31
32	F	32
33	G	33
34	H	34
35	I	35
36	J	36
37	K	37
38	L	38
39	M	39
40	N	40
41	O	41
42	P	42
43	Q	43
44	R	44
45	S	45
46	T	46
47	U	47
48	V	48
49	W	49
50	X	50
51	Y	51
52	Z	52
53	.	53
54	,	54
55	!	55
56	?	56
57	'	57
58	"	58
59	(	59
60	)	60
61		61
62	:	62

```
63      ;  63
64      -  64
```

Part Two: Creating functions

Write your own function which includes the steps below.

0. Add 14 to every number (already done for you)
1. Multiply every number by 18, then subtract 257.
2. Exponentiate every number. (That is, do e^[number].)
3. Square every number.

```
arithmetics = function(x) {
  # step 0: add 14
  x = x + 14
  # step 1: multiply by 18, subtract 257
  x = (x * 18) - 257
  # step 2: exponentiate every number
  x = exp(x)
  # step 3: square every number
  x = x^2
  # print output
  return(x)
}
```

Test your function, do you get any errors when applying it?

```
test = seq(0, 1, length.out = 5)
arithmetics(test)
```

```
[1] 4.539993e-05 3.678794e-01 2.980958e+03 2.415495e+07 1.957296e+11
```

Part Three: Decoding the secret message

This chunk (which should NOT have errors) will load up the encoded secret message as a vector:

```
top_secret <- read.csv(
  "https://raw.githubusercontent.com/loreabad6/app-dev-gis/main/practicals/Practical1/Secret_
  header = TRUE)$x
```

By altering this top secret set of numbers, you will be able to create a message. Use your function above to alter the `top_secret` object.

```
# apply your function to the top_secret object and overwrite top_secret
top_secret = arithmetics(top_secret)
```

> ⚠ Checkpoint
>
> Headquarters has informed you that at this stage of decoding, there should be 552 numbers in the secret message that are below 17.

Hint: This is what is called a "relational" comparison, where you compare an object to a number and R will give you a vector of `TRUE`s and `FALSE`s based on whether the comparison is / is not met. You can then use these `TRUE`s and `FALSE`s as numbers, since `TRUE = 1` and `FALSE = 0` in R land. This is called a Boolean variable type.

```
sum(top_secret < 17)
```

```
[1] 552
```

4. Turn your vector of numbers into a matrix with 5 columns.

```
secret_matrix <- matrix(top_secret, ncol = 5, byrow = FALSE)

secret_matrix
```

```
      [,1] [,2] [,3] [,4] [,5]
 [1,]   32  361    5  2.5 30.5
 [2,]   19   25   18 30.5  7.5
 [3,]   29  484   61 19.5  9.0
 [4,]   22   25   23  4.0 30.5
 [5,]   64  324    8  0.5 13.5
 [6,]   73    1    1 19.5 10.5
 [7,]   18  144   39 30.5  9.5
 [8,]   31 3721   61 11.5 19.5
 [9,]   41 1521   38  0.5  9.0
[10,]   34   64    1  6.0  0.5
[11,]   76   81   39  5.5  6.0
[12,]   85  196    9 30.5  4.5
[13,]   30   49   39 11.5  0.5
[14,]   43  361   21  4.5 28.0
```

```
[15,]   53 3721    4 19.5 29.0
[16,]   46  225    5  4.0 30.5
[17,]   87   36   61 30.5 29.5
[18,]   97 3721  115 19.5  0.5
[19,]   87 1521  118  4.0  7.0
[20,]   55   64  161  2.5  2.0
[21,]   63   81  138  4.5 30.5
[22,]   56  361  115  9.0  9.5
[23,]   50 3721  114 30.5  4.0
[24,]  109  361  107  4.0  2.5
[25,]   89  225    9  2.5 30.5
[26,]   60  324   39  0.5 19.5
[27,]   59 1521   21  2.0  9.0
[28,]  117 3721    4  9.5  4.5
[29,]   64   81    5 30.5  2.5
[30,]   61  196   61  2.0  2.0
[31,]   74 3721   35  7.5 30.5
[32,]   76   64   57 11.5 19.5
[33,]  127   25   22  7.0  7.5
[34,]   82  324    5 11.5 30.5
[35,]   75 3721   61  0.5  1.5
[36,]   94  144    7  9.0 10.5
[37,]   79   25   15  2.0  9.0
[38,]  400  361   39 27.5 19.5
[39,] 3799  361   61 30.5  9.5
[40,]   89  225   39 23.0  2.5
[41,]  307  196   15  4.0 12.5
[42,]  253  361   56  2.5 30.5
[43,]  111 3721   58 30.5  0.5
[44,] 3809   81   61 13.5  9.5
[45,] 1611  196   59  7.0 30.5
[46,]  317 3721   27 19.5  9.5
[47,] 3815 1521   12  4.5  4.0
[48,]   97   64    9  8.0  2.5
[49,]  294   25    3  0.5 30.5
[50,] 3821 3721    5 19.5  9.5
[51,]  127  361   61  4.0  8.0
[52,]  300    9    8  4.5  7.5
[53,]  122   64    1  2.5  5.5
[54,] 3244  225    4  9.5  2.5
[55,] 3831  225   61 27.0 32.0
[56,] 3476  144   14 30.5  3.0
[57,] 1339  324   15 17.5  0.5
```

```
  [58,] 3837  225   61 30.5  7.0
  [59,]  647  225    9 19.5  1.5
  [60,]  345  169    4  4.0 12.5
  [61,]  318 2916    5  4.5 30.5
  [62,]  140 3721    1  7.0  1.5
  [63,]  151    1   61  5.5 10.5
  [64,]  452  196   23 32.0  9.0
  [65,] 3851   16    8 29.0 19.5
  [66,]  196 3721    1 30.5  9.5
  [67,]  359 1521   39 29.5  2.5
  [68,]  665   64   61  9.5 12.5
  [69,] 3859  225   38  4.0  4.5
  [70,]  309  441    1  2.5  7.0
  [71,]  143   49   39 30.5  3.5
  [72,]  340   64    9 11.5 30.5
  [73,]  771 3721   39  0.5  0.5
  [74,] 3869 1521   21  9.5  9.5
  [75,]  319   64    4 30.5 30.5
  [76,]  233   81    5  9.0 12.5
  [77,]  298  361   61  0.5  7.5
  [78,]  181 3721   23 19.5 10.5
  [79,]  519  529    1  4.0 28.5
  [80,] 3881    1   19  2.5  9.0
  [81,] 1387  361   54  9.0  2.5
  [82,] 3413 3721   61 30.5 30.5
  [83,]  650  196   15  3.5  3.0
  [84,]  193  225   18  6.0  0.5
  [85,] 3891 1521   61  0.5  6.0
  [86,]  208 3721   38  2.0  6.0
  [87,]  175    1   15 30.5  4.5
  [88,]  320 3721   14 19.5  7.0
  [89,]  322  484    7  4.0  3.5
  [90,]  205   25    9  2.5 30.5
  [91,]  378  324   39  9.0 19.5
  [92,] 3905  625   21  2.5  4.0
  [93,]  190 3721    4 30.5  9.0
  [94,]  813   49    5 11.5  7.5
  [95,] 3911  225   61  0.5 10.5
  [96,] 1713  225    5  9.5  3.5
  [97,]  258   16    9 30.5  4.0
  [98,]  277 3721   39  7.0 30.5
  [99,]  559  225    8  7.5 19.5
 [100,] 3921  256    5 30.5  4.0
```

```
[101,] 1723  256   18  7.5  2.5
[102,]  285  225   54  7.0 30.5
[103,]  375  324   61  2.5  0.5
[104,]  233 1521    2 30.5  4.5
[105,] 3346  441   21  6.0  9.0
[106,] 3576  196   39  4.5 27.5
[107,] 3935   81   61  9.5 30.5
[108,]  577 1521   39 19.5 15.0
[109,]  282  625    8  2.5  7.5
[110,]  245 3721   15  7.0 30.5
[111,] 3943   36   21  4.5 12.5
[112,]  585  225    7  7.0  7.5
[113,]  227  324    8  3.5 10.5
[114,]  309 3721   39 27.0 30.5
[115,]  246  361   61 30.5 19.5
[116,] 3953   64   39 19.5  4.0
[117,]  235  225    8  4.0  4.5
[118,]  380  529    5  4.5  7.0
[119,]  463   81   25  9.5  5.5
[120,]  681  196   61 30.5 30.5
[121,]  258   49   23 19.5 12.5
[122,] 3053 3721    5  4.5  7.5
[123,] 3967  225   18  6.5 10.5
[124,] 3612   36    5  2.5 30.5
[125,] 1475   36   61 27.0  1.5
[126,] 3973 3721   14 30.5  7.5
[127,]  423   64    9  0.5 10.5
[128,]  697   25    3  9.5  6.0
[129,]  619  324    5 30.5  2.0
[130,] 1781 3721   61  4.5 30.5
[131,] 3983  121    7 19.5  6.5
[132,]  268  196   18 30.5  0.5
[133,]  291  225    1  2.0  7.0
[134,] 3989  529   14  4.5  0.5
[135,]  319  144    4  2.0  3.5
[136,]  297   25   61  7.0  2.5
[137,] 1795   16   23 28.5 30.5
[138,] 1797   49   15 19.5  4.5
[139,]  359   25   18 30.5 19.5
[140,]  476 2916    4  9.5 28.0
[141,]  331 3721   19  7.5 30.0
[142,] 4005    1   61 10.5 30.5
[143,]  647  361   39  7.0 29.0
```

```
[144,]  513 3721   15  2.0 13.5
[145,]  459 1521   61 30.5  7.0
[146,]  317   64   19  0.5  2.0
[147,]  823   25    1 19.5 30.5
[148,]  360  324   25 30.5 11.5
[149,]  323   25   53  0.5  4.0
[150,]  624 3721   60  6.0  0.5
[151,]  327  529   42  6.0 19.5
[152,] 4025    1   18 30.5 30.5
[153,]  502  361    5 19.5  0.5
[154,]  333 3721   19  4.0  7.0
[155,]  311  196    5  2.5 30.5
[156,]  636  225   14 30.5  4.5
[157,] 4035 3721   39  9.0  3.5
[158,] 1837  225   12  4.5  7.0
[159,]  382  196   25  3.5  7.5
[160,]  345   25   61  4.0  9.0
[161,] 4043 3721   19 19.5  0.5
[162,]  333 1521    8 30.5  7.0
[163,]  351  225    5 11.5 19.5
[164,]  524 3721   61  7.5 30.5
[165,] 1851  144    2  9.0  6.0
[166,]  656   81    5  2.0  4.5
[167,]  359  361    7 30.0 19.5
[168,] 4057 1521    1 30.5 19.5
[169,]  563   25   14 29.0  6.0
[170,]  376  196   61 32.0  2.5
[171,] 4063 3721    1  1.0 30.5
[172,] 1865 1521    7 10.5  3.5
[173,]  410  225    1 19.5  4.5
[174,]  373 3721    9 30.5  9.0
[175,] 4071   64   14 17.5  6.0
[176,]  377   25   53 30.5 30.5
[177,]  355  324   61  9.5  9.5
[178,]  680 2916   58  4.0  4.0
[179,] 1879 3721   35  0.5  2.5
[180,]  424  361   61  6.0 28.5
[181,] 3171 1521   23  6.0  6.0
[182,] 4085   81   15 30.5  6.0
[183,] 1810  144   14  4.0 30.5
[184,]  393  144    4  0.5 19.5
[185,] 1891 3721    5 11.0  4.0
[186,] 4093   81   18  2.5  4.5
```

```
[187,]  543 1521   61 30.5  7.0
[188,]  401 3721    9 19.5  5.5
[189,] 4099  529    6  7.5 30.5
[190,]  741    1   61 30.5  6.5
[191,]  407   19   35  0.5  2.5
[192,]  409   61   61  9.5 30.5
[193,] 4230    7   19  5.5  3.0
[194,] 4109   15    8 30.5  7.5
[195,] 1911   15    1 19.5  9.0
[196,]  456    4   12  4.0 30.5
[197,]  395   61   12  2.5  0.5
[198,] 1917   16   61  6.5  9.5
[199,] 4119   18    6 30.5  5.5
[200,]  929    1    1 11.5  4.5
[201,]  627    3   12  4.0  7.0
[202,]  845   39   12  0.5  3.5
[203,]  550    9   61 19.5 27.5
[204,]  424    3   18 30.5 30.5
[205,] 4131    5    9 19.5 20.0
[206,]  416   61    7  4.0  7.5
[207,]  439   39    8  2.5 27.0
[208,] 4137   15   39 30.5 30.5
[209,]  454   61   61  7.0  4.5
[210,]  645   19   39  0.5 19.5
[211,]  863    1    8  6.5 28.5
[212,]  748   25   18  2.5  6.0
[213,] 4147   61   15 30.5  6.0
[214,] 1949    9   21  7.5 30.5
[215,]  494   39    7  3.0  7.0
[216,]  657   61    8 30.5  2.5
[217,]  875   15   61 19.5 11.0
[218,]  797   22   39  4.0  2.5
[219,]  439    5    8  2.5  9.0
[220,]  636   18    5 30.5 30.5
[221,]  458   60   61  1.5  2.0
[222,] 4165   61    5  7.5  7.5
[223,]  615   58    1 10.5 30.5
[224,]  529   64   18  7.0 19.5
[225,]  594   25   39 19.5  7.5
[226,]  477    5    8  9.0 30.5
[227,]  815   19   55 12.5  0.5
[228,] 4177   54   61 30.5  9.5
[229,]  474   61   34  4.5  5.5
```

```
[230,]  685   39   15  9.5 31.0
[231,]  991    8   23 27.0 30.5
[232,]  660    1   61 30.5  8.0
[233,] 3382   39    6 12.5  2.5
[234,] 4189   57   21  7.5  9.0
[235,] 1695   19   14 10.5  4.0
[236,] 4193   61   14 30.5  0.5
[237,] 1995    1   25  5.5  8.0
[238,]  540    2   61  7.0  9.5
[239,]  559   15    9  7.5 30.5
[240,]  676   21   39 11.5 17.5
[241,]  603   39   57 26.5 30.5
[242,] 4580   61   12 30.5  9.5
[243,] 3850   39   12 21.0  4.0
[244,] 4209    8   61  6.0  0.5
[245,] 3971    5   19  2.5  6.0
[246,]  528   61    5  0.5  6.0
[247,]  719   18    5  9.5 30.5
[248,]  820    9   13  2.5  9.5
[249,] 3414    7   61 27.0  2.5
[250,] 4221    8   39 30.5  2.5
[251,] 1127   39   15 19.5 30.5
[252,]  729   61   61  0.5  4.5
[253,]  947    4    3 28.5 19.5
[254,] 4229    9   15  0.5 30.5
[255,]  871   19   13  6.5 11.5
[256,]  537   39    5 27.0  9.0
[257,]  539    1   61 30.5  4.5
[258,] 3432   14   15  4.5 19.5
[259,] 4239    3   21  9.5 19.5
[260,] 1249    5   39 30.5  2.5
[261,]  666   64   61 19.5  7.0
[262,]  605    2    1  4.0 30.5
[263,]  535   21   13  4.5 10.5
[264,]  553   39   15  9.5  8.0
[265,] 4251   61   14 30.5 30.5
[266,]  596   39    7 20.0  9.5
[267,]  535    8   61  2.5  7.5
[268,]  552    5   39 11.5  6.5
[269,] 4259   14    8 30.5  2.5
[270,]  684   61    5 26.0 11.5
[271,]  567   35   61  2.5  4.0
[272,]  545   61   16  0.5  2.5
```

```
[273,]  870   23    5  6.0  9.0
[274,]  744   15   15  0.5  2.5
[275,] 2071   14   16  7.0 26.5
[276,] 4273    4   12  2.0 29.0
```

5. Separately from your `top_secret` numbers, create a new vector called "evens" of all the even numbers between 1 and 552. That is, "evens" should contain 2, 4, 6, 8 …, 552.

```
evens <- seq(2, 552, by = 2)

evens
```

```
  [1]   2   4   6   8  10  12  14  16  18  20  22  24  26  28  30  32  34  36
 [19]  38  40  42  44  46  48  50  52  54  56  58  60  62  64  66  68  70  72
 [37]  74  76  78  80  82  84  86  88  90  92  94  96  98 100 102 104 106 108
 [55] 110 112 114 116 118 120 122 124 126 128 130 132 134 136 138 140 142 144
 [73] 146 148 150 152 154 156 158 160 162 164 166 168 170 172 174 176 178 180
 [91] 182 184 186 188 190 192 194 196 198 200 202 204 206 208 210 212 214 216
[109] 218 220 222 224 226 228 230 232 234 236 238 240 242 244 246 248 250 252
[127] 254 256 258 260 262 264 266 268 270 272 274 276 278 280 282 284 286 288
[145] 290 292 294 296 298 300 302 304 306 308 310 312 314 316 318 320 322 324
[163] 326 328 330 332 334 336 338 340 342 344 346 348 350 352 354 356 358 360
[181] 362 364 366 368 370 372 374 376 378 380 382 384 386 388 390 392 394 396
[199] 398 400 402 404 406 408 410 412 414 416 418 420 422 424 426 428 430 432
[217] 434 436 438 440 442 444 446 448 450 452 454 456 458 460 462 464 466 468
[235] 470 472 474 476 478 480 482 484 486 488 490 492 494 496 498 500 502 504
[253] 506 508 510 512 514 516 518 520 522 524 526 528 530 532 534 536 538 540
[271] 542 544 546 548 550 552
```

6. Subtract the "evens" vector from the first column of your secret message matrix.

```
secret_matrix[, 1] <- secret_matrix[, 1] - evens

secret_matrix
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]   30  361    5  2.5 30.5
[2,]   15   25   18 30.5  7.5
[3,]   23  484   61 19.5  9.0
[4,]   14   25   23  4.0 30.5
[5,]   54  324    8  0.5 13.5
[6,]   61    1    1 19.5 10.5
```

```
 [7,]    4  144   39 30.5  9.5
 [8,]   15 3721   61 11.5 19.5
 [9,]   23 1521   38  0.5  9.0
[10,]   14   64    1  6.0  0.5
[11,]   54   81   39  5.5  6.0
[12,]   61  196    9 30.5  4.5
[13,]    4   49   39 11.5  0.5
[14,]   15  361   21  4.5 28.0
[15,]   23 3721    4 19.5 29.0
[16,]   14  225    5  4.0 30.5
[17,]   53   36   61 30.5 29.5
[18,]   61 3721  115 19.5  0.5
[19,]   49 1521  118  4.0  7.0
[20,]   15   64  161  2.5  2.0
[21,]   21   81  138  4.5 30.5
[22,]   12  361  115  9.0  9.5
[23,]    4 3721  114 30.5  4.0
[24,]   61  361  107  4.0  2.5
[25,]   39  225    9  2.5 30.5
[26,]    8  324   39  0.5 19.5
[27,]    5 1521   21  2.0  9.0
[28,]   61 3721    4  9.5  4.5
[29,]    6   81    5 30.5  2.5
[30,]    1  196   61  2.0  2.0
[31,]   12 3721   35  7.5 30.5
[32,]   12   64   57 11.5 19.5
[33,]   61   25   22  7.0  7.5
[34,]   14  324    5 11.5 30.5
[35,]    5 3721   61  0.5  1.5
[36,]   22  144    7  9.0 10.5
[37,]    5   25   15  2.0  9.0
[38,]  324  361   39 27.5 19.5
[39,] 3721  361   61 30.5  9.5
[40,]    9  225   39 23.0  2.5
[41,]  225  196   15  4.0 12.5
[42,]  169  361   56  2.5 30.5
[43,]   25 3721   58 30.5  0.5
[44,] 3721   81   61 13.5  9.5
[45,] 1521  196   59  7.0 30.5
[46,]  225 3721   27 19.5  9.5
[47,] 3721 1521   12  4.5  4.0
[48,]    1   64    9  8.0  2.5
[49,]  196   25    3  0.5 30.5
```

```
[50,] 3721 3721    5 19.5  9.5
[51,]   25  361   61  4.0  8.0
[52,]  196    9    8  4.5  7.5
[53,]   16   64    1  2.5  5.5
[54,] 3136  225    4  9.5  2.5
[55,] 3721  225   61 27.0 32.0
[56,] 3364  144   14 30.5  3.0
[57,] 1225  324   15 17.5  0.5
[58,] 3721  225   61 30.5  7.0
[59,]  529  225    9 19.5  1.5
[60,]  225  169    4  4.0 12.5
[61,]  196 2916    5  4.5 30.5
[62,]   16 3721    1  7.0  1.5
[63,]   25    1   61  5.5 10.5
[64,]  324  196   23 32.0  9.0
[65,] 3721   16    8 29.0 19.5
[66,]   64 3721    1 30.5  9.5
[67,]  225 1521   39 29.5  2.5
[68,]  529   64   61  9.5 12.5
[69,] 3721  225   38  4.0  4.5
[70,]  169  441    1  2.5  7.0
[71,]    1   49   39 30.5  3.5
[72,]  196   64    9 11.5 30.5
[73,]  625 3721   39  0.5  0.5
[74,] 3721 1521   21  9.5  9.5
[75,]  169   64    4 30.5 30.5
[76,]   81   81    5  9.0 12.5
[77,]  144  361   61  0.5  7.5
[78,]   25 3721   23 19.5 10.5
[79,]  361  529    1  4.0 28.5
[80,] 3721    1   19  2.5  9.0
[81,] 1225  361   54  9.0  2.5
[82,] 3249 3721   61 30.5 30.5
[83,]  484  196   15  3.5  3.0
[84,]   25  225   18  6.0  0.5
[85,] 3721 1521   61  0.5  6.0
[86,]   36 3721   38  2.0  6.0
[87,]    1    1   15 30.5  4.5
[88,]  144 3721   14 19.5  7.0
[89,]  144  484    7  4.0  3.5
[90,]   25   25    9  2.5 30.5
[91,]  196  324   39  9.0 19.5
[92,] 3721  625   21  2.5  4.0
```

```
 [93,]    4 3721    4 30.5  9.0
 [94,]  625   49    5 11.5  7.5
 [95,] 3721  225   61  0.5 10.5
 [96,] 1521  225    5  9.5  3.5
 [97,]   64   16    9 30.5  4.0
 [98,]   81 3721   39  7.0 30.5
 [99,]  361  225    8  7.5 19.5
[100,] 3721  256    5 30.5  4.0
[101,] 1521  256   18  7.5  2.5
[102,]   81  225   54  7.0 30.5
[103,]  169  324   61  2.5  0.5
[104,]   25 1521    2 30.5  4.5
[105,] 3136  441   21  6.0  9.0
[106,] 3364  196   39  4.5 27.5
[107,] 3721   81   61  9.5 30.5
[108,]  361 1521   39 19.5 15.0
[109,]   64  625    8  2.5  7.5
[110,]   25 3721   15  7.0 30.5
[111,] 3721   36   21  4.5 12.5
[112,]  361  225    7  7.0  7.5
[113,]    1  324    8  3.5 10.5
[114,]   81 3721   39 27.0 30.5
[115,]   16  361   61 30.5 19.5
[116,] 3721   64   39 19.5  4.0
[117,]    1  225    8  4.0  4.5
[118,]  144  529    5  4.5  7.0
[119,]  225   81   25  9.5  5.5
[120,]  441  196   61 30.5 30.5
[121,]   16   49   23 19.5 12.5
[122,] 2809 3721    5  4.5  7.5
[123,] 3721  225   18  6.5 10.5
[124,] 3364   36    5  2.5 30.5
[125,] 1225   36   61 27.0  1.5
[126,] 3721 3721   14 30.5  7.5
[127,]  169   64    9  0.5 10.5
[128,]  441   25    3  9.5  6.0
[129,]  361  324    5 30.5  2.0
[130,] 1521 3721   61  4.5 30.5
[131,] 3721  121    7 19.5  6.5
[132,]    4  196   18 30.5  0.5
[133,]   25  225    1  2.0  7.0
[134,] 3721  529   14  4.5  0.5
[135,]   49  144    4  2.0  3.5
```

```
[136,]   25   25   61  7.0  2.5
[137,] 1521   16   23 28.5 30.5
[138,] 1521   49   15 19.5  4.5
[139,]   81   25   18 30.5 19.5
[140,]  196 2916    4  9.5 28.0
[141,]   49 3721   19  7.5 30.0
[142,] 3721    1   61 10.5 30.5
[143,]  361  361   39  7.0 29.0
[144,]  225 3721   15  2.0 13.5
[145,]  169 1521   61 30.5  7.0
[146,]   25   64   19  0.5  2.0
[147,]  529   25    1 19.5 30.5
[148,]   64  324   25 30.5 11.5
[149,]   25   25   53  0.5  4.0
[150,]  324 3721   60  6.0  0.5
[151,]   25  529   42  6.0 19.5
[152,] 3721    1   18 30.5 30.5
[153,]  196  361    5 19.5  0.5
[154,]   25 3721   19  4.0  7.0
[155,]    1  196    5  2.5 30.5
[156,]  324  225   14 30.5  4.5
[157,] 3721 3721   39  9.0  3.5
[158,] 1521  225   12  4.5  7.0
[159,]   64  196   25  3.5  7.5
[160,]   25   25   61  4.0  9.0
[161,] 3721 3721   19 19.5  0.5
[162,]    9 1521    8 30.5  7.0
[163,]   25  225    5 11.5 19.5
[164,]  196 3721   61  7.5 30.5
[165,] 1521  144    2  9.0  6.0
[166,]  324   81    5  2.0  4.5
[167,]   25  361    7 30.0 19.5
[168,] 3721 1521    1 30.5 19.5
[169,]  225   25   14 29.0  6.0
[170,]   36  196   61 32.0  2.5
[171,] 3721 3721    1  1.0 30.5
[172,] 1521 1521    7 10.5  3.5
[173,]   64  225    1 19.5  4.5
[174,]   25 3721    9 30.5  9.0
[175,] 3721   64   14 17.5  6.0
[176,]   25   25   53 30.5 30.5
[177,]    1  324   61  9.5  9.5
[178,]  324 2916   58  4.0  4.0
```

```
[179,] 1521 3721   35  0.5  2.5
[180,]   64  361   61  6.0 28.5
[181,] 2809 1521   23  6.0  6.0
[182,] 3721   81   15 30.5  6.0
[183,] 1444  144   14  4.0 30.5
[184,]   25  144    4  0.5 19.5
[185,] 1521 3721    5 11.0  4.0
[186,] 3721   81   18  2.5  4.5
[187,]  169 1521   61 30.5  7.0
[188,]   25 3721    9 19.5  5.5
[189,] 3721  529    6  7.5 30.5
[190,]  361    1   61 30.5  6.5
[191,]   25   19   35  0.5  2.5
[192,]   25   61   61  9.5 30.5
[193,] 3844    7   19  5.5  3.0
[194,] 3721   15    8 30.5  7.5
[195,] 1521   15    1 19.5  9.0
[196,]   64    4   12  4.0 30.5
[197,]    1   61   12  2.5  0.5
[198,] 1521   16   61  6.5  9.5
[199,] 3721   18    6 30.5  5.5
[200,]  529    1    1 11.5  4.5
[201,]  225    3   12  4.0  7.0
[202,]  441   39   12  0.5  3.5
[203,]  144    9   61 19.5 27.5
[204,]   16    3   18 30.5 30.5
[205,] 3721    5    9 19.5 20.0
[206,]    4   61    7  4.0  7.5
[207,]   25   39    8  2.5 27.0
[208,] 3721   15   39 30.5 30.5
[209,]   36   61   61  7.0  4.5
[210,]  225   19   39  0.5 19.5
[211,]  441    1    8  6.5 28.5
[212,]  324   25   18  2.5  6.0
[213,] 3721   61   15 30.5  6.0
[214,] 1521    9   21  7.5 30.5
[215,]   64   39    7  3.0  7.0
[216,]  225   61    8 30.5  2.5
[217,]  441   15   61 19.5 11.0
[218,]  361   22   39  4.0  2.5
[219,]    1    5    8  2.5  9.0
[220,]  196   18    5 30.5 30.5
[221,]   16   60   61  1.5  2.0
```

```
[222,] 3721   61    5  7.5  7.5
[223,]  169   58    1 10.5 30.5
[224,]   81   64   18  7.0 19.5
[225,]  144   25   39 19.5  7.5
[226,]   25    5    8  9.0 30.5
[227,]  361   19   55 12.5  0.5
[228,] 3721   54   61 30.5  9.5
[229,]   16   61   34  4.5  5.5
[230,]  225   39   15  9.5 31.0
[231,]  529    8   23 27.0 30.5
[232,]  196    1   61 30.5  8.0
[233,] 2916   39    6 12.5  2.5
[234,] 3721   57   21  7.5  9.0
[235,] 1225   19   14 10.5  4.0
[236,] 3721   61   14 30.5  0.5
[237,] 1521    1   25  5.5  8.0
[238,]   64    2   61  7.0  9.5
[239,]   81   15    9  7.5 30.5
[240,]  196   21   39 11.5 17.5
[241,]  121   39   57 26.5 30.5
[242,] 4096   61   12 30.5  9.5
[243,] 3364   39   12 21.0  4.0
[244,] 3721    8   61  6.0  0.5
[245,] 3481    5   19  2.5  6.0
[246,]   36   61    5  0.5  6.0
[247,]  225   18    5  9.5 30.5
[248,]  324    9   13  2.5  9.5
[249,] 2916    7   61 27.0  2.5
[250,] 3721    8   39 30.5  2.5
[251,]  625   39   15 19.5 30.5
[252,]  225   61   61  0.5  4.5
[253,]  441    4    3 28.5 19.5
[254,] 3721    9   15  0.5 30.5
[255,]  361   19   13  6.5 11.5
[256,]   25   39    5 27.0  9.0
[257,]   25    1   61 30.5  4.5
[258,] 2916   14   15  4.5 19.5
[259,] 3721    3   21  9.5 19.5
[260,]  729    5   39 30.5  2.5
[261,]  144   64   61 19.5  7.0
[262,]   81    2    1  4.0 30.5
[263,]    9   21   13  4.5 10.5
[264,]   25   39   15  9.5  8.0
```

```
[265,] 3721   61   14 30.5 30.5
[266,]   64   39    7 20.0  9.5
[267,]    1    8   61  2.5  7.5
[268,]   16    5   39 11.5  6.5
[269,] 3721   14    8 30.5  2.5
[270,]  144   61    5 26.0 11.5
[271,]   25   35   61  2.5  4.0
[272,]    1   61   16  0.5  2.5
[273,]  324   23    5  6.0  9.0
[274,]  196   15   15  0.5  2.5
[275,] 1521   14   16  7.0 26.5
[276,] 3721    4   12  2.0 29.0
```

7. Subtract 100 from all numbers 18-24th rows of the 3rd column.

```
secret_matrix[18:24, 3] <- secret_matrix[18:24, 3] - 100
```

8. Multiply all numbers in the 4th and 5th column by 2.

```
secret_matrix[, 4] <- secret_matrix[, 4] * 2
secret_matrix[, 5] <- secret_matrix[, 5] * 2
```

9. Turn your matrix back into a vector.

```
top_secret_vector <- as.vector(secret_matrix)
top_secret_vector
```

```
  [1]   30   15   23   14   54   61    4   15   23   14   54   61    4   15
 [15]   23   14   53   61   49   15   21   12    4   61   39    8    5   61
 [29]    6    1   12   12   61   14    5   22    5  324 3721    9  225  169
 [43]   25 3721 1521  225 3721    1  196 3721   25  196   16 3136 3721 3364
 [57] 1225 3721  529  225  196   16   25  324 3721   64  225  529 3721  169
 [71]    1  196  625 3721  169   81  144   25  361 3721 1225 3249  484   25
 [85] 3721   36    1  144  144   25  196 3721    4  625 3721 1521   64   81
 [99]  361 3721 1521   81  169   25 3136 3364 3721  361   64   25 3721  361
[113]    1   81   16 3721    1  144  225  441   16 2809 3721 3364 1225 3721
[127]  169  441  361 1521 3721    4   25 3721   49   25 1521 1521   81  196
[141]   49 3721  361  225  169   25  529   64   25  324   25 3721  196   25
[155]    1  324 3721 1521   64   25 3721    9   25  196 1521  324   25 3721
[169]  225   36 3721 1521   64   25 3721   25    1  324 1521   64 2809 3721
[183] 1444   25 1521 3721  169   25 3721  361   25   25 3844 3721 1521   64
[197]    1 1521 3721  529  225  441  144   16 3721    4   25 3721   36  225
```

```
[211]  441  324 3721 1521   64  225  441  361    1  196   16 3721  169   81
[225]  144   25  361 3721   16  225  529  196 2916 3721 1225 3721 1521   64
[239]   81  196  121 4096 3364 3721 3481   36  225  324 2916 3721  625  225
[253]  441 3721  361   25   25 2916 3721  729  144   81    9   25 3721   64
[267]    1   16 3721  144   25    1  324  196 1521 3721  361   25  484   25
[281]  324    1  144 3721 1521   64   81  196   49  361 3721  225   36 3721
[295] 1521   64   81  361 3721  361  225  324 1521 3721   81  196 3721   64
[309]   25  324 3721  144   25  361  361  225  196  361 3721   81  196 3721
[323] 1521   64   25 3721  361    9   64  225  225  144  324  225  225  169
[337] 2916 3721    1  196   16 3721 1521   64  225  441   49   64 3721 1521
[351]   64   81  361 3721  529    1  361 3721  196  225 1521 3721    1 3721
[365]  484   25  324  625 3721   49  225  225   16 3721  225  256  256  225
[379]  324 1521  441  196   81 1521  625 3721   36  225  324 3721  361   64
[393]  225  529   81  196   49 3721  225   36   36 3721   64   25  324 3721
[407]  121  196  225  529  144   25   16   49   25 2916 3721    1  361 3721
[421] 1521   64   25  324   25 3721  529    1  361 3721  196  225 3721  225
[435]  196   25 3721 1521  225 3721  144   81  361 1521   25  196 3721 1521
[449]  225 3721   64   25  324 2916 3721  361 1521   81  144  144 3721   81
[463] 1521 3721  529    1   19   61    7   15   15    4   61   16   18    1
[477]    3   39    9    3    5   61   39   15   61   19    1   25   61    9
[491]   39   61   15   22    5   18   60   61   58   64   25    5   19   54
[505]   61   39    8    1   39   57   19   61    1    2   15   21   39   61
[519]   39    8    5   61   18    9    7    8   39   61    4    9   19   39
[533]    1   14    3    5   64    2   21   39   61   39    8    5   14   61
[547]   35   61   23   15   14    4    5   18   61   23    8    1   39   61
[561]   38    1   39    9   39   21    4    5   61   15   18   61   38   15
[575]   14    7    9   39   21    4    5   61   35   57   22    5   61    7
[589]   15   39   61   39   15   56   58   61   59   27   12    9    3    5
[603]   61    8    1    4   61   14   15   61    9    4    5    1   61   23
[617]    8    1   39   61   38    1   39    9   39   21    4    5   61   23
[631]    1   19   54   61   15   18   61   38   15   14    7    9   39   21
[645]    4    5   61    5    9   39    8    5   18   54   61    2   21   39
[659]   61   39    8   15   21    7    8   39   61   39    8    5   25   61
[673]   23    5   18    5   61   14    9    3    5   61    7   18    1   14
[687]    4   61   23   15   18    4   19   61   39   15   61   19    1   25
[701]   53   60   42   18    5   19    5   14   39   12   25   61   19    8
[715]    5   61    2    5    7    1   14   61    1    7    1    9   14   53
[729]   61   58   35   61   23   15   14    4    5   18   61    9    6   61
[743]   35   61   19    8    1   12   12   61    6    1   12   12   61   18
[757]    9    7    8   39   61   39    8   18   15   21    7    8   61   39
[771]    8    5   61    5    1   18   39    8   55   61   34   15   23   61
[785]    6   21   14   14   25   61    9   39   57   12   12   61   19    5
[799]    5   13   61   39   15   61    3   15   13    5   61   15   21   39
```

```
 [813]  61   1  13  15  14   7  61  39   8   5  61  16   5  15
 [827]  16  12   5  61  39   8   1  39  61  23   1  12  11  61
 [841]  23   9  39   8  61  39   8   5   9  18  61   8   5   1
 [855]   4  19  61   4  15  23  14  23   1  18   4  55  61  46
 [869]   8   5  61  27  14  39   9  16   1  39   8   9   5  19
 [883]  54  61  35  61  39   8   9  14  11  64  58  61  59  19
 [897]   8   5  61  23   1  19  61  18   1  39   8   5  18  61
 [911]   7  12   1   4  61  39   8   5  18   5  61  23   1  19
 [925]  61  14  15  61  15  14   5  61  12   9  19  39   5  14
 [939]   9  14   7  54  61  39   8   9  19  61  39   9  13   5
 [953]  54  61   1  19  61   9  39  61   4   9   4  14  57  39
 [967]  61  19  15  21  14   4  61   1  39  61   1  12  12  61
 [981]  39   8   5  61  18   9   7   8  39  61  23  15  18   4
 [995]  60  61  58  64   2  21  39  61  35  61  19   8   1  12
[1009]  12  61   8   1  22   5  61  39  15  61   1  19  11  61
[1023]  39   8   5  13  61  23   8   1  39  61  39   8   5  61
[1037]  14   1  13   5  61  15   6  61  39   8   5  61   3  15
[1051]  21  14  39  18  25  61   9  19  54  61  25  15  21  61
[1065]  11  14  15  23  53  61  42  12   5   1  19   5  54  61
[1079]  39   1  57   1  13  54  61   9  19  61  39   8   9  19
[1093]  61  40   5  23  61  52   5   1  12   1  14   4  61  15
[1107]  18  61  27  21  19  39  18   1  12   9   1  56  58  61
[1121]  59   1  14   4  61  19   8   5  61  39  18   9   5   4
[1135]  61  39  15  61   3  21  18  39  19   5  25  61   1  19
[1149]  61  19   8   5  61  19  16  15  11   5  64   6   1  14
[1163]   3  25  61   3  21  18  39  19   5  25   9  14   7  61
[1177]   1  19  61  25  15  21  57  18   5  61   6   1  12  12
[1191]   9  14   7  61  39   8  18  15  21   7   8  61  39   8
[1205]   5  61   1   9  18  55  61  30  15  61  25  15  21  61
[1219]  39   8   9  14  11  61  25  15  21  61   3  15  21  12
[1233]   4  61  13   1  14   1   7   5  61   9  39  56  60  61
[1247]  58  27  14   4  61  23   8   1  39  61   1  14  61   9
[1261]   7  14  15  18   1  14  39  61  12   9  39  39  12   5
[1275]  61   7   9  18  12  61  19   8   5  57  12  12  61  39
[1289]   8   9  14  11  61  13   5  61   6  15  18  61   1  19
[1303]  11   9  14   7  55  61  40  15  54  61   9  39  57  12
[1317]  12  61  14   5  22   5  18  61   4  15  61  39  15  61
[1331]   1  19  11  62  61  16   5  18   8   1  16  19  61  35
[1345]  61  19   8   1  12  12  61  19   5   5  61   9  39  61
[1359]  23  18   9  39  39   5  14  61  21  16  61  19  15  13
[1373]   5  23   8   5  18   5  53  58
```

> ⚠ Checkpoint
>
> Headquarters has informed you that at this stage of decoding, all numbers in indices 500 and beyond are below 70.

Hint: Use a relational comparison similar to what you used in the last checkpoint, but here you will need to subset values from your vector! It may be helpful to think of below as *not equal* to or *smaller than* 70.

```
sum(top_secret_vector[500:length(top_secret_vector)] > 70)
```

```
[1] 0
```

10. Take the square root of all numbers in indices 38 to 465.

```
top_secret_vector[38:465] <- sqrt(top_secret_vector[38:465])
```

11. Round all numbers to the nearest whole number.

```
top_secret_vector <- round(top_secret_vector)
```

12. Replace all instances of the number 39 with 20.

```
top_secret_vector[top_secret_vector == 39] <- 20
```

> ⚠ Checkpoint
>
> Headquarters has informed you that your final message should have 507 even numbers.

Hint: Checking for divisibility is an interesting operation that isn't done much in R. Modulus is the operation you are interested in, where you are checking for whether the numbers are divisible by 2, with no remainder. See what you can find about modulus in R!

```
sum(top_secret_vector %% 2 == 0) # Should return 507 even numbers
```

```
[1] 507
```

```
# Should be 507!
```

Part Four: The secret message!

Use your final vector of numbers as indices for `my_symbols` to discover the final message! The code to do so is already there for you:

```
cat(my_symbols$Symbol[top_secret_vector], sep = "")
```

Down, down, down. Would the fall never come to an end? "I wonder how many miles I've fallen by this time?" she said aloud. "I must be getting somewhere near the centre of the earth. Let me see: that would be four thousand miles down, I think—" (for, you see, Alice had learnt several things of this sort in her lessons in the schoolroom, and though this was not a very good opportunity for showing off her knowledge, as there was no one to listen to her, still it was good practice to say it over) "—yes, that's about the right distance—but then I wonder what Latitude or Longitude I've got to?" (Alice had no idea what Latitude was, or Longitude either, but thought they were nice grand words to say.)Presently she began again. "I wonder if I shall fall right through the earth! How funny it'll seem to come out among the people that walk with their heads downward! The Antipathies, I think—" (she was rather glad there was no one listening, this time, as it didn't sound at all the right word) "—but I shall have to ask them what the name of the country is, you know. Please, ta'am, is this New Zealand or Australia?" (and she tried to curtsey as she spoke—fancy curtseying as you're falling through the air! Do you think you could manage it?) "And what an ignorant little girl she'll think me for asking! No, it'll never do to ask: perhaps I shall see it written up somewhere."

Google the message, if you do not recognize it, and find its title and author.

> **Solution**
>
> Title: Alice's Adventures in Wonderland and Author: Lewis Carroll
> Upload the .qmd doc and the rendered html to Blackboard (don't forget to add all your teammates names!). The first team receives an extra point each in class participation