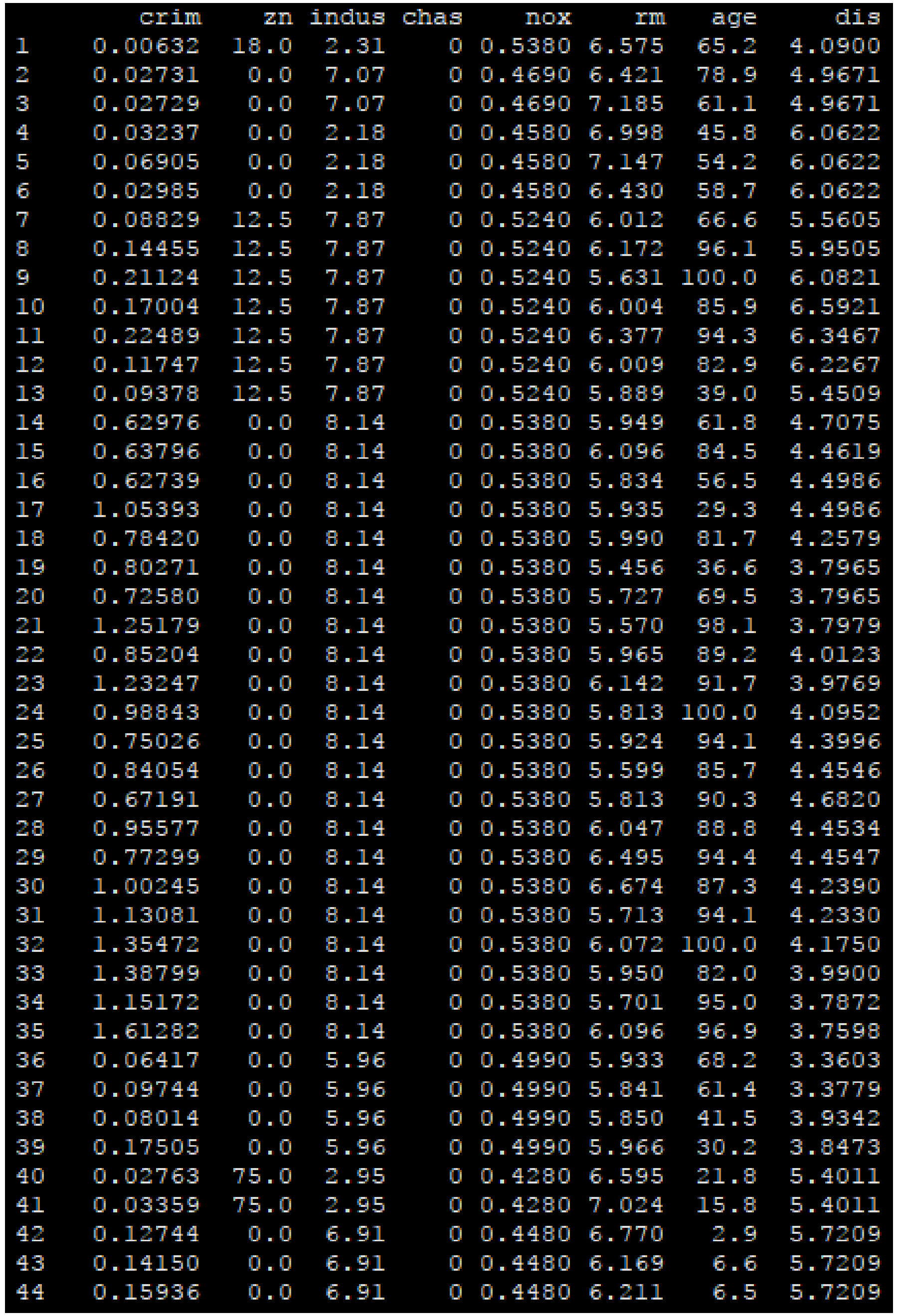
Subsetting Vectors

```
x = 54, 10, 79, 39, 7, 75, 93, 68, 77, 59, 31, 74, 3, 81
```

Subsetting

Start with full data

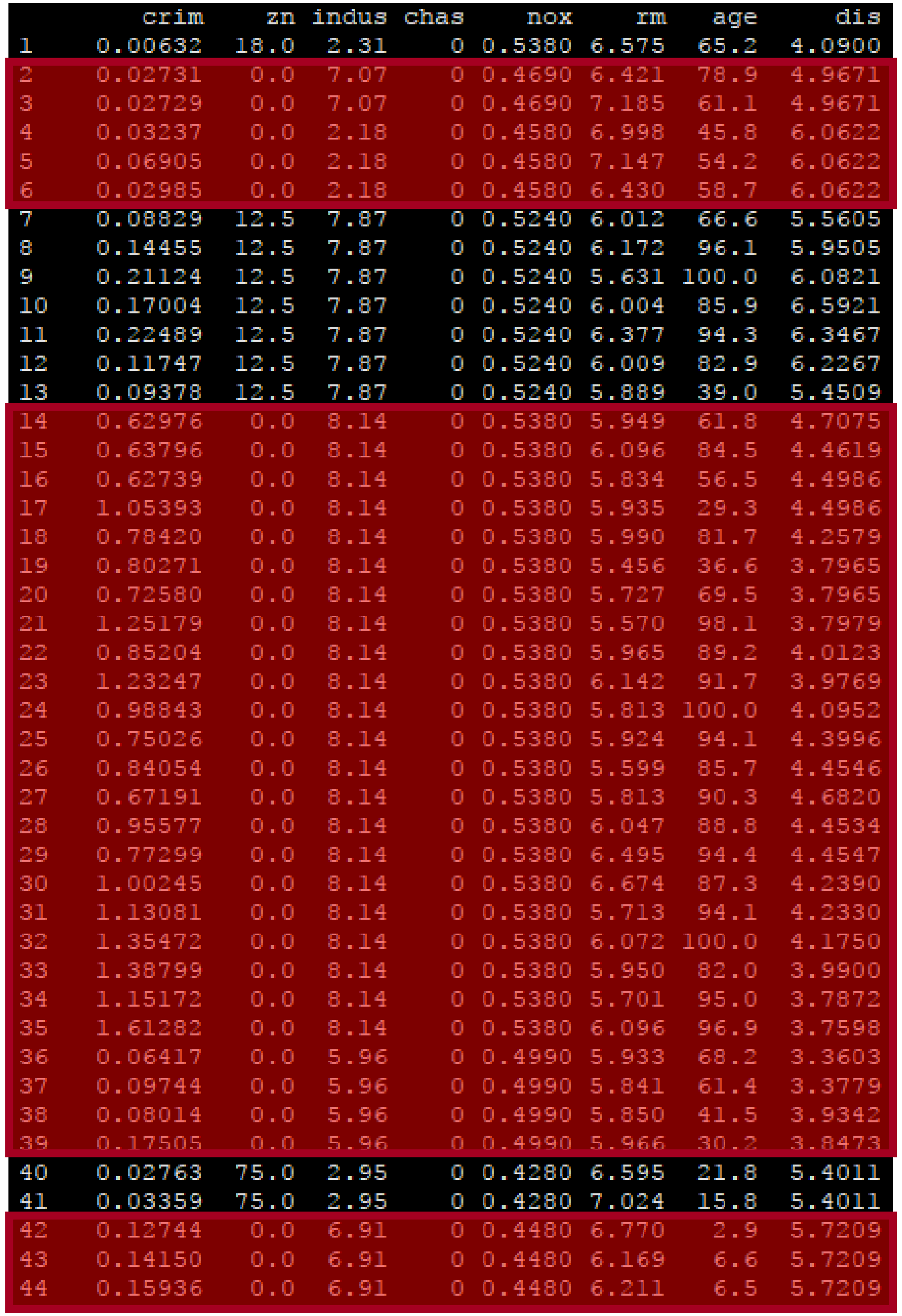


```
       crim   zn indus chas    nox    rm   age    dis
1   0.00632 18.0  2.31    0 0.5380 6.575  65.2 4.0900
2   0.02731  0.0  7.07    0 0.4690 6.421  78.9 4.9671
3   0.02729  0.0  7.07    0 0.4690 7.185  61.1 4.9671
4   0.03237  0.0  2.18    0 0.4580 6.998  45.8 6.0622
5   0.06905  0.0  2.18    0 0.4580 7.147  54.2 6.0622
6   0.02985  0.0  2.18    0 0.4580 6.430  58.7 6.0622
7   0.08829 12.5  7.87    0 0.5240 6.012  66.6 5.5605
8   0.14455 12.5  7.87    0 0.5240 6.172  96.1 5.9505
9   0.21124 12.5  7.87    0 0.5240 5.631 100.0 6.0821
10  0.17004 12.5  7.87    0 0.5240 6.004  85.9 6.5921
11  0.22489 12.5  7.87    0 0.5240 6.377  94.3 6.3467
12  0.11747 12.5  7.87    0 0.5240 6.009  82.9 6.2267
13  0.09378 12.5  7.87    0 0.5240 5.889  39.0 5.4509
14  0.62976  0.0  8.14    0 0.5380 5.949  61.8 4.7075
15  0.63796  0.0  8.14    0 0.5380 6.096  84.5 4.4619
16  0.62739  0.0  8.14    0 0.5380 5.834  56.5 4.4986
17  1.05393  0.0  8.14    0 0.5380 5.935  29.3 4.4986
18  0.78420  0.0  8.14    0 0.5380 5.990  81.7 4.2579
19  0.80271  0.0  8.14    0 0.5380 5.456  36.6 3.7965
20  0.72580  0.0  8.14    0 0.5380 5.727  69.5 3.7965
21  1.25179  0.0  8.14    0 0.5380 5.570  98.1 3.7979
22  0.85204  0.0  8.14    0 0.5380 5.965  89.2 4.0123
23  1.23247  0.0  8.14    0 0.5380 6.142  91.7 3.9769
24  0.98843  0.0  8.14    0 0.5380 5.813 100.0 4.0952
25  0.75026  0.0  8.14    0 0.5380 5.924  94.1 4.3996
26  0.84054  0.0  8.14    0 0.5380 5.599  85.7 4.4546
27  0.67191  0.0  8.14    0 0.5380 5.813  90.3 4.6820
28  0.95577  0.0  8.14    0 0.5380 6.047  88.8 4.4534
29  0.77299  0.0  8.14    0 0.5380 6.495  94.4 4.4547
30  1.00245  0.0  8.14    0 0.5380 6.674  87.3 4.2390
31  1.13081  0.0  8.14    0 0.5380 5.713  94.1 4.2330
32  1.35472  0.0  8.14    0 0.5380 6.072 100.0 4.1750
33  1.38799  0.0  8.14    0 0.5380 5.950  82.0 3.9900
34  1.15172  0.0  8.14    0 0.5380 5.701  95.0 3.7872
35  1.61282  0.0  8.14    0 0.5380 6.096  96.9 3.7598
36  0.06417  0.0  5.96    0 0.4990 5.933  68.2 3.3603
37  0.09744  0.0  5.96    0 0.4990 5.841  61.4 3.3779
38  0.08014  0.0  5.96    0 0.4990 5.850  41.5 3.9342
39  0.17505  0.0  5.96    0 0.4990 5.966  30.2 3.8473
40  0.02763 75.0  2.95    0 0.4280 6.595  21.8 5.4011
41  0.03359 75.0  2.95    0 0.4280 7.024  15.8 5.4011
42  0.12744  0.0  6.91    0 0.4480 6.770   2.9 5.7209
43  0.14150  0.0  6.91    0 0.4480 6.169   6.6 5.7209
44  0.15936  0.0  6.91    0 0.4480 6.211   6.5 5.7209
```

Subsetting

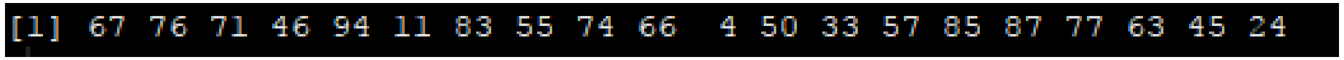
Start with full data

Select only the parts of our objects we want.



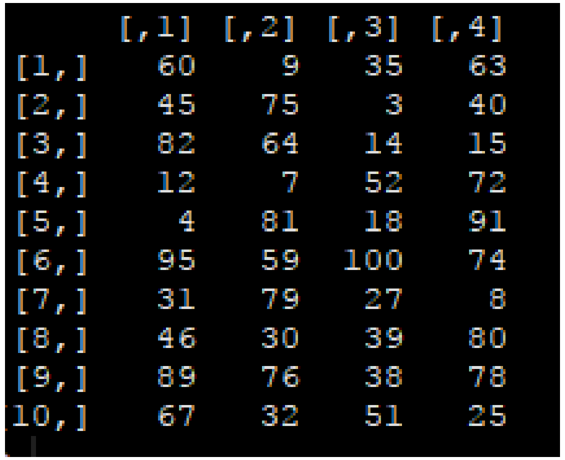
```
         crim   zn indus chas    nox    rm   age    dis
1     0.00632 18.0  2.31    0 0.5380 6.575  65.2 4.0900
2     0.02731  0.0  7.07    0 0.4690 6.421  78.9 4.9671
3     0.02729  0.0  7.07    0 0.4690 7.185  61.1 4.9671
4     0.03237  0.0  2.18    0 0.4580 6.998  45.8 6.0622
5     0.06905  0.0  2.18    0 0.4580 7.147  54.2 6.0622
6     0.02985  0.0  2.18    0 0.4580 6.430  58.7 6.0622
7     0.08829 12.5  7.87    0 0.5240 6.012  66.6 5.5605
8     0.14455 12.5  7.87    0 0.5240 6.172  96.1 5.9505
9     0.21124 12.5  7.87    0 0.5240 5.631 100.0 6.0821
10    0.17004 12.5  7.87    0 0.5240 6.004  85.9 6.5921
11    0.22489 12.5  7.87    0 0.5240 6.377  94.3 6.3467
12    0.11747 12.5  7.87    0 0.5240 6.009  82.9 6.2267
13    0.09378 12.5  7.87    0 0.5240 5.889  39.0 5.4509
14    0.62976  0.0  8.14    0 0.5380 5.949  61.8 4.7075
15    0.63796  0.0  8.14    0 0.5380 6.096  84.5 4.4619
16    0.62739  0.0  8.14    0 0.5380 5.834  56.5 4.4986
17    1.05393  0.0  8.14    0 0.5380 5.935  29.3 4.4986
18    0.78420  0.0  8.14    0 0.5380 5.990  81.7 4.2579
19    0.80271  0.0  8.14    0 0.5380 5.456  36.6 3.7965
20    0.72580  0.0  8.14    0 0.5380 5.727  69.5 3.7965
21    1.25179  0.0  8.14    0 0.5380 5.570  98.1 3.7979
22    0.85204  0.0  8.14    0 0.5380 5.965  89.2 4.0123
23    1.23247  0.0  8.14    0 0.5380 6.142  91.7 3.9769
24    0.98843  0.0  8.14    0 0.5380 5.813 100.0 4.0952
25    0.75026  0.0  8.14    0 0.5380 5.924  94.1 4.3996
26    0.84054  0.0  8.14    0 0.5380 5.599  85.7 4.4546
27    0.67191  0.0  8.14    0 0.5380 5.813  90.3 4.6820
28    0.95577  0.0  8.14    0 0.5380 6.047  88.8 4.4534
29    0.77299  0.0  8.14    0 0.5380 6.495  94.4 4.4547
30    1.00245  0.0  8.14    0 0.5380 6.674  87.3 4.2390
31    1.13081  0.0  8.14    0 0.5380 5.713  94.1 4.2330
32    1.35472  0.0  8.14    0 0.5380 6.072 100.0 4.1750
33    1.38799  0.0  8.14    0 0.5380 5.950  82.0 3.9900
34    1.15172  0.0  8.14    0 0.5380 5.701  95.0 3.7872
35    1.61282  0.0  8.14    0 0.5380 6.096  96.9 3.7598
36    0.06417  0.0  5.96    0 0.4990 5.933  68.2 3.3603
37    0.09744  0.0  5.96    0 0.4990 5.841  61.4 3.3779
38    0.08014  0.0  5.96    0 0.4990 5.850  41.5 3.9342
39    0.17505  0.0  5.96    0 0.4990 5.966  30.2 3.8473
40    0.02763 75.0  2.95    0 0.4280 6.595  21.8 5.4011
41    0.03359 75.0  2.95    0 0.4280 7.024  15.8 5.4011
42    0.12744  0.0  6.91    0 0.4480 6.770   2.9 5.7209
43    0.14150  0.0  6.91    0 0.4480 6.169   6.6 5.7209
44    0.15936  0.0  6.91    0 0.4480 6.211   6.5 5.7209
```

Vector: one-dimensional object of all the same data type



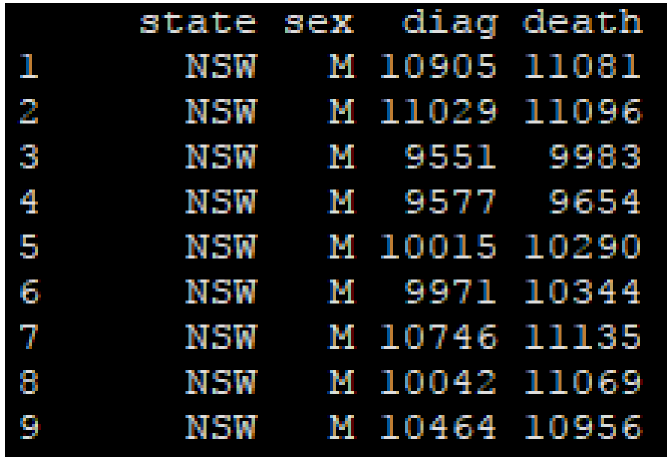
```
[1] 67 76 71 46 94 11 83 55 74 66  4 50 33 57 85 87 77 63 45 24
```

Matrix: two-dimensional object of all the same data type



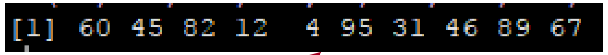
```
      [,1] [,2] [,3] [,4]
 [1,]   60    9   35   63
 [2,]   45   75    3   40
 [3,]   82   64   14   15
 [4,]   12    7   52   72
 [5,]    4   81   18   91
 [6,]   95   59  100   74
 [7,]   31   79   27    8
 [8,]   46   30   39   80
 [9,]   89   76   38   78
[10,]   67   32   51   25
```

Dataframe: two-dimensional object of any data types



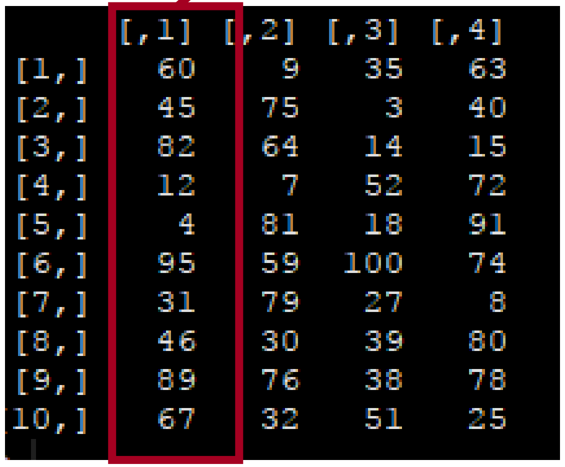
```
      state sex  diag death
1       NSW   M 10905 11081
2       NSW   M 11029 11096
3       NSW   M  9551  9983
4       NSW   M  9577  9654
5       NSW   M 10015 10290
6       NSW   M  9971 10344
7       NSW   M 10746 11135
8       NSW   M 10042 11069
9       NSW   M 10464 10956
```

Vector: one-dimensional object of all the same data type



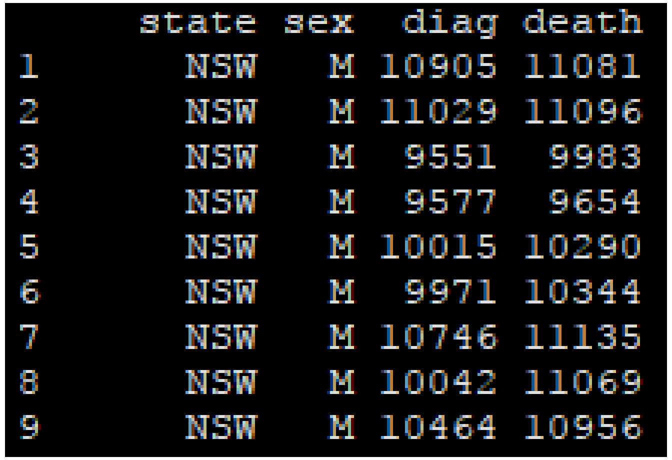
```
[1] 60 45 82 12  4 95 31 46 89 67
```

Matrix: two-dimensional object of all the same data type



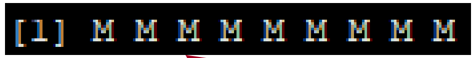
```
      [,1] [,2] [,3] [,4]
 [1,]   60    9   35   63
 [2,]   45   75    3   40
 [3,]   82   64   14   15
 [4,]   12    7   52   72
 [5,]    4   81   18   91
 [6,]   95   59  100   74
 [7,]   31   79   27    8
 [8,]   46   30   39   80
 [9,]   89   76   38   78
[10,]   67   32   51   25
```

Dataframe: two-dimensional object of any data types



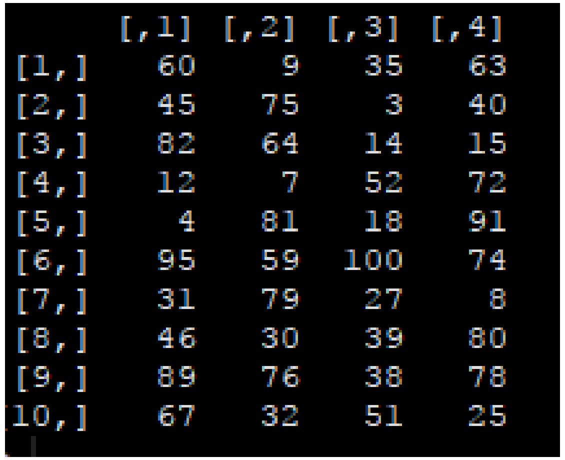
```
     state sex  diag death
1      NSW   M 10905 11081
2      NSW   M 11029 11096
3      NSW   M  9551  9983
4      NSW   M  9577  9654
5      NSW   M 10015 10290
6      NSW   M  9971 10344
7      NSW   M 10746 11135
8      NSW   M 10042 11069
9      NSW   M 10464 10956
```

Vector: one-dimensional object of all the same data type



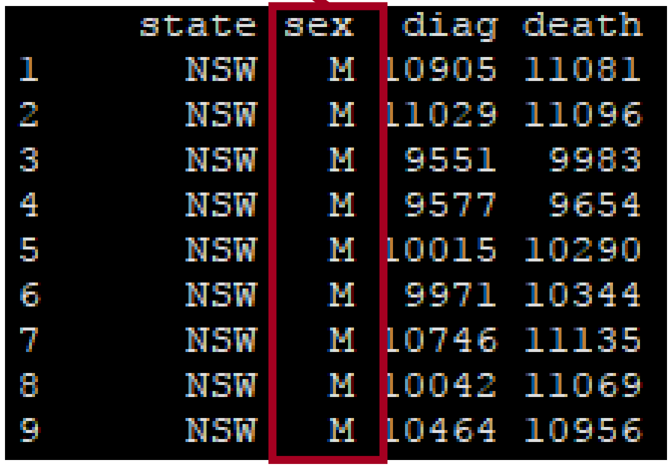
```
[1] M M M M M M M M M
```

Matrix: two-dimensional object of all the same data type



```
      [,1] [,2] [,3] [,4]
[1,]    60    9   35   63
[2,]    45   75    3   40
[3,]    82   64   14   15
[4,]    12    7   52   72
[5,]     4   81   18   91
[6,]    95   59  100   74
[7,]    31   79   27    8
[8,]    46   30   39   80
[9,]    89   76   38   78
[10,]   67   32   51   25
```

Dataframe: two-dimensional object of any data types



```
    state sex  diag death
1     NSW   M 10905 11081
2     NSW   M 11029 11096
3     NSW   M  9551  9983
4     NSW   M  9577  9654
5     NSW   M 10015 10290
6     NSW   M  9971 10344
7     NSW   M 10746 11135
8     NSW   M 10042 11069
9     NSW   M 10464 10956
```

Two primary ways to subset vectors:

Object Indices: selecting values by their location in the object

Logical Statements: selecting only values that are either TRUE or FALSE in a logical statement

Two primary ways to subset vectors:

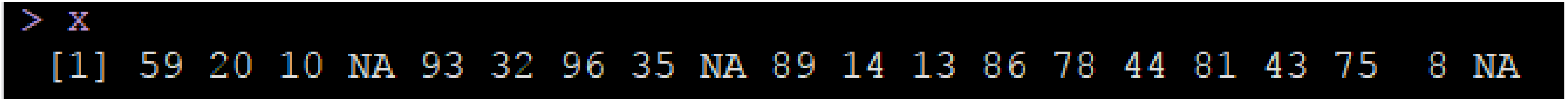
Object Indices: selecting values by their location in the object

Logical Statements: selecting only values that are either TRUE or FALSE in a logical statement

↳ Each uses the subset operator: [ ]

object[…]

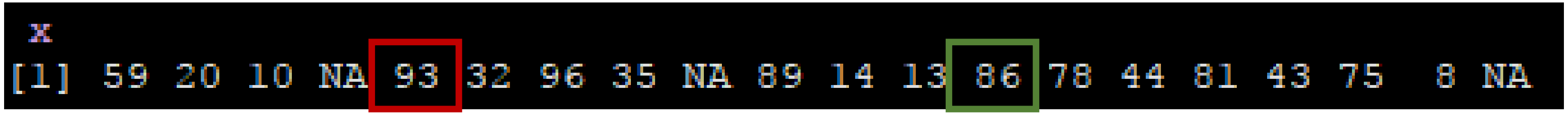
Object Indices



```
> x
 [1] 59 20 10 NA 93 32 96 35 NA 89 14 13 86 78 44 81 43 75  8 NA
```

1 2 3 . . . → . . . 19 20

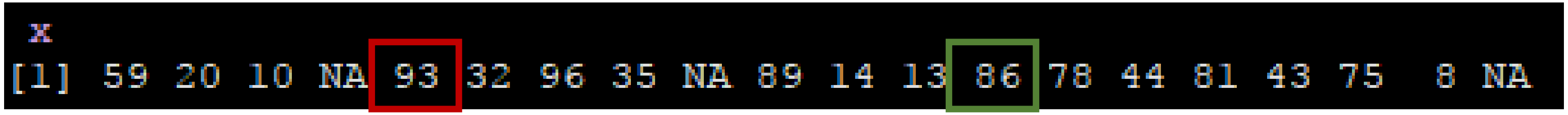
Object Indices



```
 x
[1]  59  20  10  NA  93  32  96  35  NA  89  14  13  86  78  44  81  43  75   8  NA
```

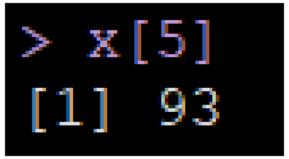
Single indices

Object Indices

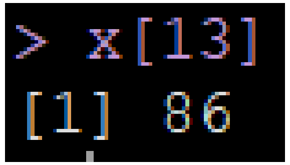


```
 x
[1] 59 20 10 NA 93 32 96 35 NA 89 14 13 86 78 44 81 43 75  8 NA
```

Single indices

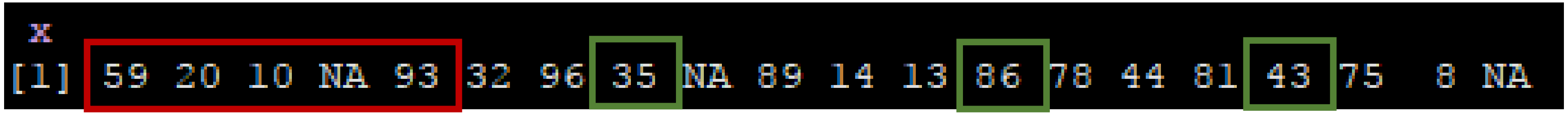


```
> x[5]
[1] 93
```



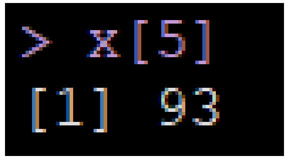
```
> x[13]
[1] 86
```

Object Indices

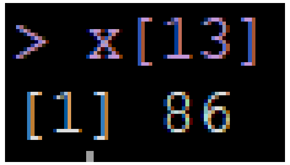


```
 x
[1]  59  20  10  NA  93  32  96  35  NA  89  14  13  86  78  44  81  43  75   8  NA
```

Single indices



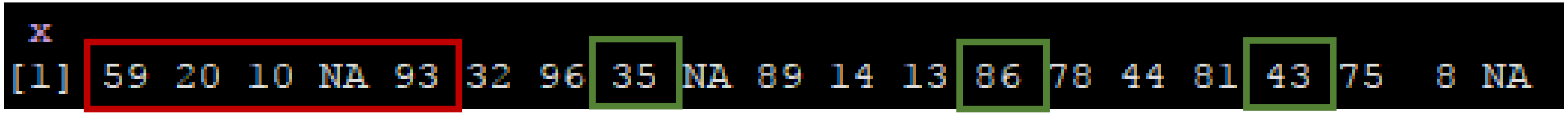
```
> x[5]
[1]  93
```



```
> x[13]
[1]  86
```

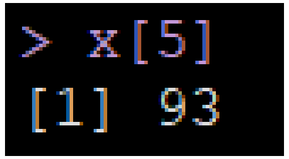
Vectors of indices

Object Indices

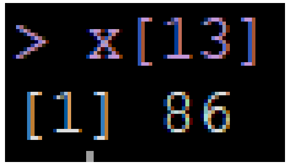


```
x
[1] 59 20 10 NA 93 32 96 35 NA 89 14 13 86 78 44 81 43 75  8 NA
```

Single indices

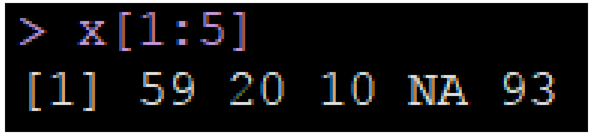


```
> x[5]
[1] 93
```

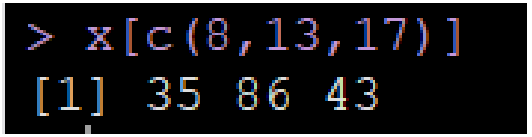


```
> x[13]
[1] 86
```

Vectors of indices

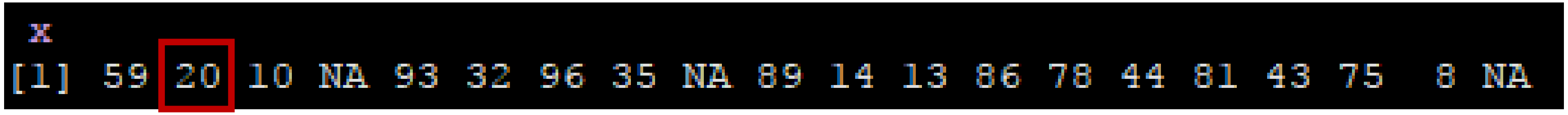


```
> x[1:5]
[1] 59 20 10 NA 93
```



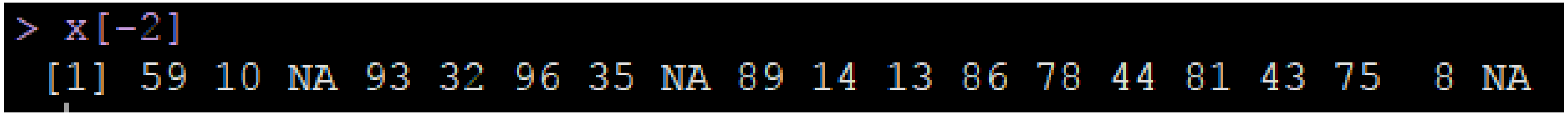
```
> x[c(8,13,17)]
[1] 35 86 43
```

Object Indices



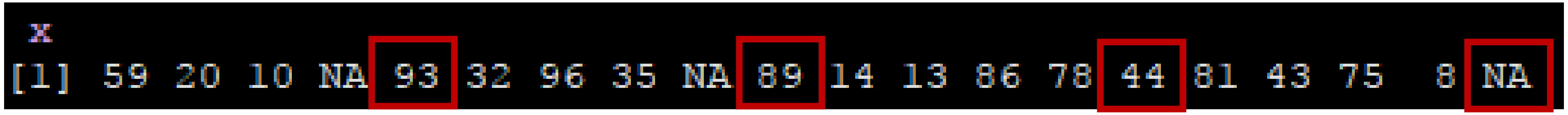
```
 x
[1]  59  20  10  NA  93  32  96  35  NA  89  14  13  86  78  44  81  43  75   8  NA
```

Remove values



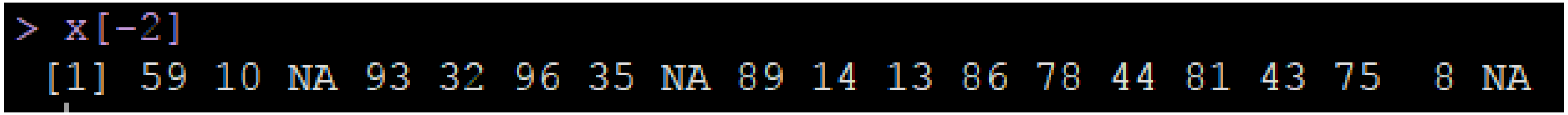
```
> x[-2]
 [1]  59  10  NA  93  32  96  35  NA  89  14  13  86  78  44  81  43  75   8  NA
```

Object Indices

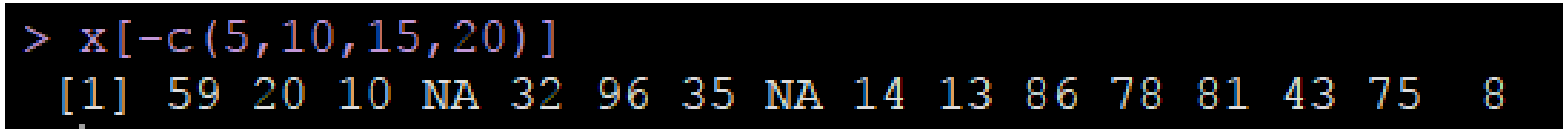


```
x
[1] 59 20 10 NA 93 32 96 35 NA 89 14 13 86 78 44 81 43 75  8 NA
```

Remove values

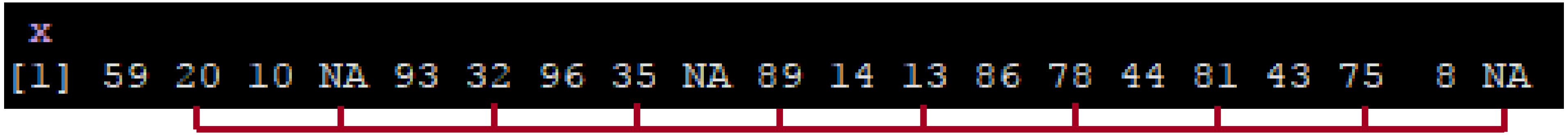


```
> x[-2]
 [1] 59 10 NA 93 32 96 35 NA 89 14 13 86 78 44 81 43 75  8 NA
```



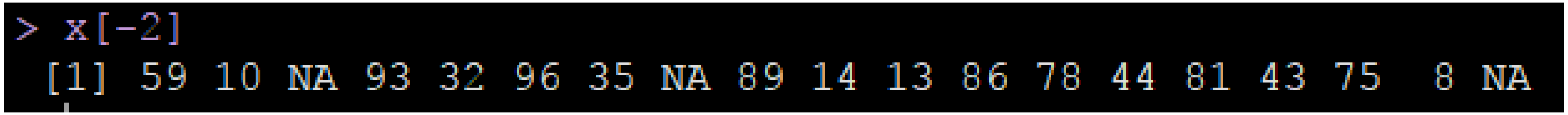
```
> x[-c(5,10,15,20)]
 [1] 59 20 10 NA 32 96 35 NA 14 13 86 78 81 43 75  8
```

Object Indices

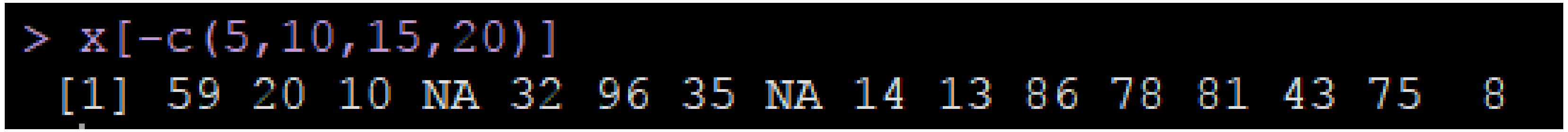


```
 x
[1] 59 20 10 NA 93 32 96 35 NA 89 14 13 86 78 44 81 43 75  8 NA
```

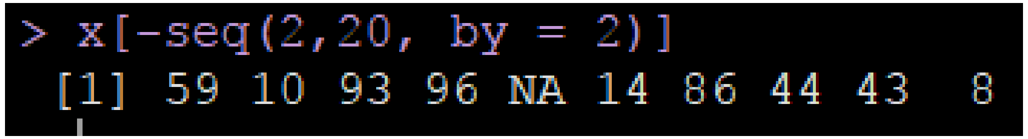
Remove values



```
> x[-2]
 [1] 59 10 NA 93 32 96 35 NA 89 14 13 86 78 44 81 43 75  8 NA
```

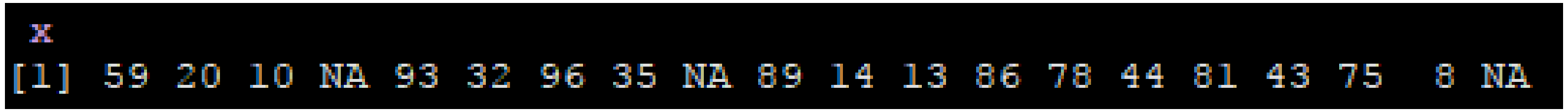


```
> x[-c(5,10,15,20)]
 [1] 59 20 10 NA 32 96 35 NA 14 13 86 78 81 43 75  8
```



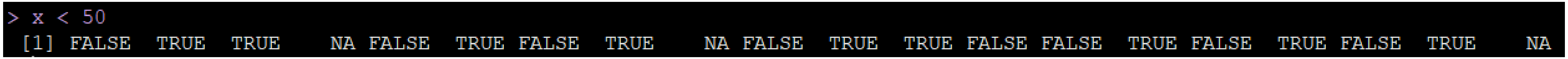
```
> x[-seq(2,20, by = 2)]
 [1] 59 10 93 96 NA 14 86 44 43  8
```

Logical Statements

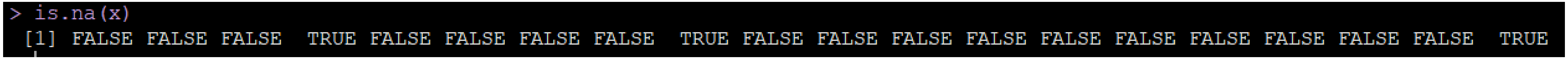


```
 x
[1] 59 20 10 NA 93 32 96 35 NA 89 14 13 86 78 44 81 43 75  8 NA
```

We can use logical statements to create vectors of TRUE and FALSE, each corresponding to an element in our original vector.

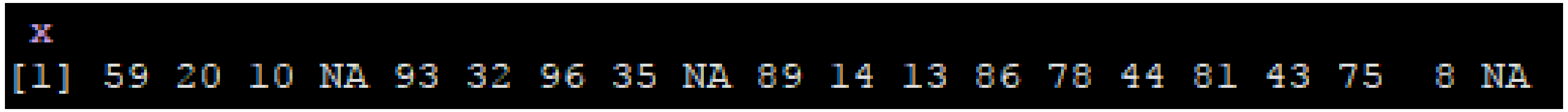


```
> x < 50
 [1] FALSE  TRUE  TRUE    NA FALSE  TRUE FALSE  TRUE    NA FALSE  TRUE  TRUE FALSE FALSE  TRUE FALSE  TRUE FALSE  TRUE    NA
```



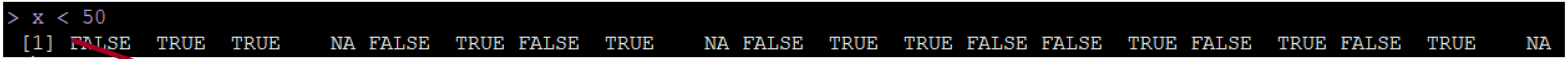
```
> is.na(x)
 [1] FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
```

Logical Statements

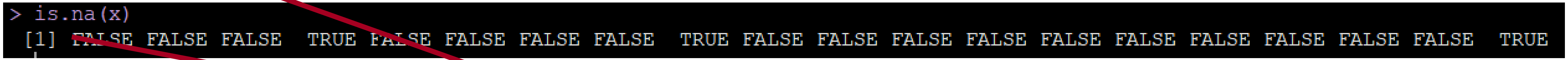


```
 x
[1] 59 20 10 NA 93 32 96 35 NA 89 14 13 86 78 44 81 43 75  8 NA
```

We can use logical statements to create vectors of TRUE and FALSE, each corresponding to an element in our original vector.



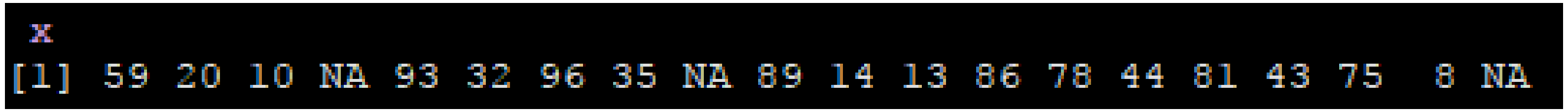
```
> x < 50
 [1] FALSE  TRUE  TRUE    NA FALSE  TRUE FALSE  TRUE    NA FALSE  TRUE  TRUE FALSE FALSE  TRUE FALSE  TRUE FALSE  TRUE    NA
```



```
> is.na(x)
 [1] FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
```

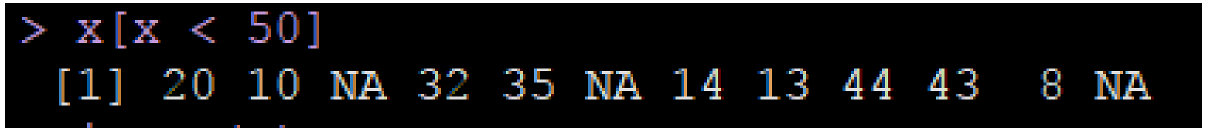
```
x[...]
```

Logical Statements

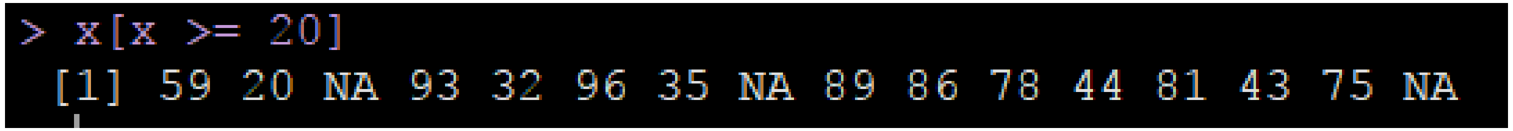


```
 x
[1] 59 20 10 NA 93 32 96 35 NA 89 14 13 86 78 44 81 43 75  8 NA
```

Selecting with inequalities

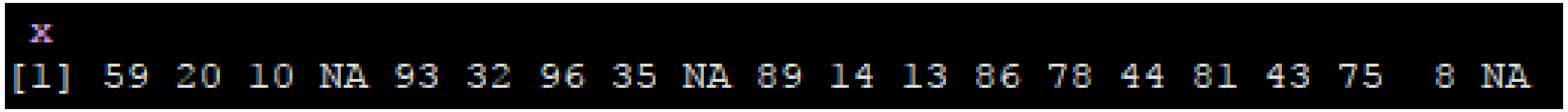


```
> x[x < 50]
 [1] 20 10 NA 32 35 NA 14 13 44 43  8 NA
```



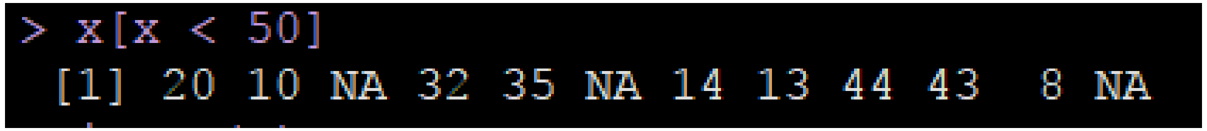
```
> x[x >= 20]
 [1] 59 20 NA 93 32 96 35 NA 89 86 78 44 81 43 75 NA
```

Logical Statements

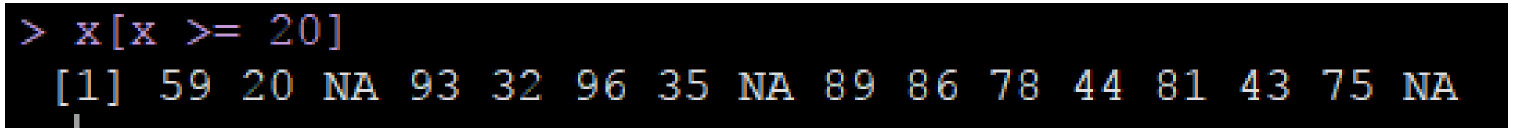


```
 x
[1] 59 20 10 NA 93 32 96 35 NA 89 14 13 86 78 44 81 43 75  8 NA
```

Selecting with inequalities

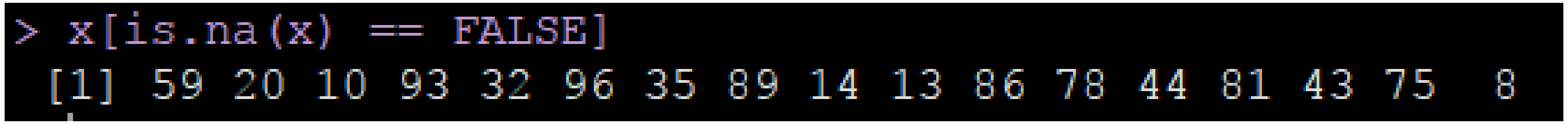


```
> x[x < 50]
 [1] 20 10 NA 32 35 NA 14 13 44 43  8 NA
```



```
> x[x >= 20]
 [1] 59 20 NA 93 32 96 35 NA 89 86 78 44 81 43 75 NA
```

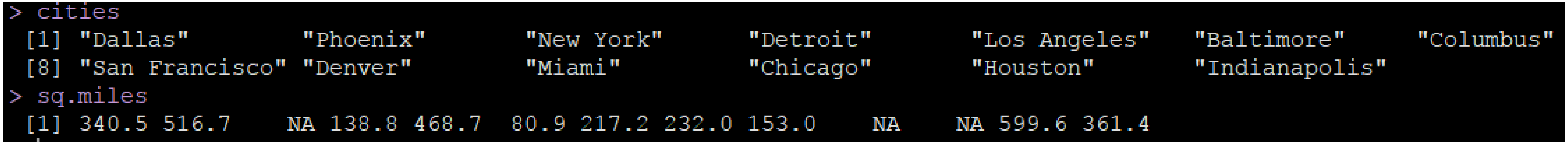
Removing NAs



```
> x[is.na(x) == FALSE]
 [1] 59 20 10 93 32 96 35 89 14 13 86 78 44 81 43 75  8
```

Logical Statements

Can use *any* logical vector

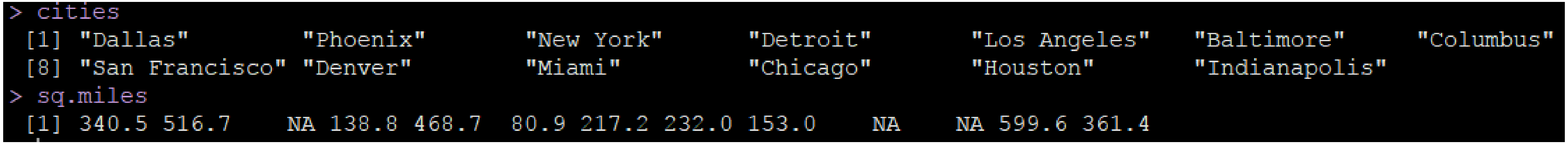


```
> cities
 [1] "Dallas"        "Phoenix"       "New York"      "Detroit"       "Los Angeles"   "Baltimore"     "Columbus"
 [8] "San Francisco" "Denver"        "Miami"         "Chicago"       "Houston"       "Indianapolis"
> sq.miles
 [1] 340.5 516.7    NA 138.8 468.7  80.9 217.2 232.0 153.0    NA    NA 599.6 361.4
```

For which cities are we missing data?

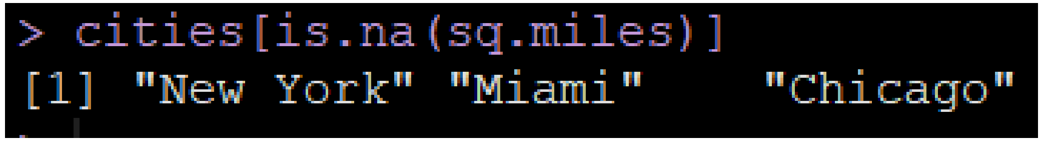
Logical Statements

Can use *any* logical vector



```
> cities
 [1] "Dallas"        "Phoenix"       "New York"      "Detroit"       "Los Angeles"   "Baltimore"     "Columbus"
 [8] "San Francisco" "Denver"        "Miami"         "Chicago"       "Houston"       "Indianapolis"
> sq.miles
 [1] 340.5 516.7    NA 138.8 468.7  80.9 217.2 232.0 153.0    NA    NA 599.6 361.4
```

For which cities are we missing data?

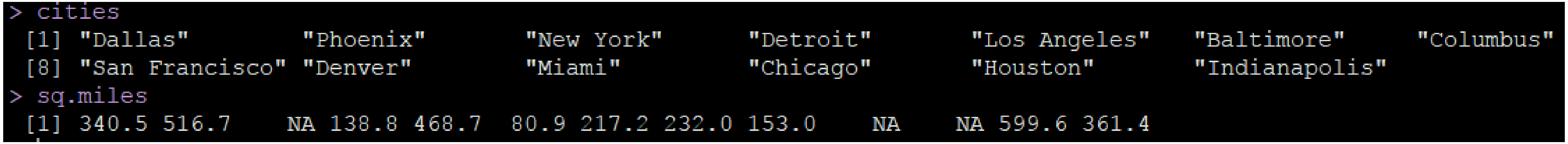


```
> cities[is.na(sq.miles)]
[1] "New York" "Miami"    "Chicago"
```

Which cities are more than 300 square miles in area?

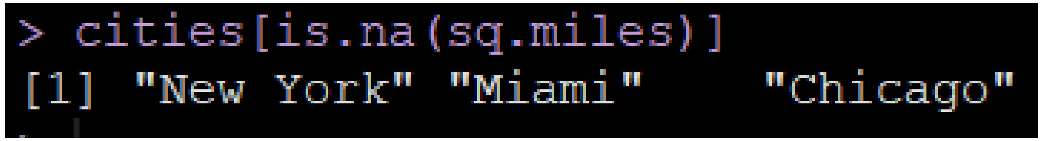
Logical Statements

Can use *any* logical vector



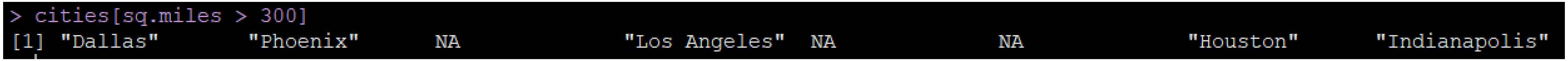
```
> cities
 [1] "Dallas"        "Phoenix"       "New York"      "Detroit"       "Los Angeles"   "Baltimore"     "Columbus"
 [8] "San Francisco" "Denver"        "Miami"         "Chicago"       "Houston"       "Indianapolis"
> sq.miles
 [1] 340.5 516.7    NA 138.8 468.7  80.9 217.2 232.0 153.0    NA    NA 599.6 361.4
```

For which cities are we missing data?



```
> cities[is.na(sq.miles)]
[1] "New York" "Miami"    "Chicago"
```

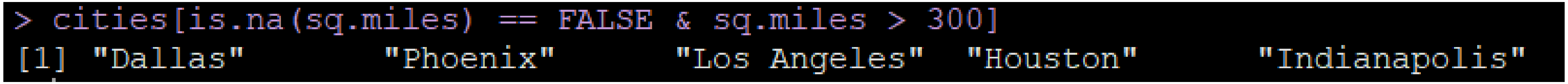
Which cities are more than 300 square miles in area?



```
> cities[sq.miles > 300]
[1] "Dallas"        "Phoenix"       NA              "Los Angeles"   NA              NA              "Houston"       "Indianapolis"
```

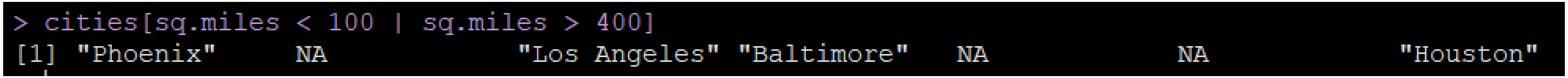
Multiple logical statements

AND: &



```
> cities[is.na(sq.miles) == FALSE & sq.miles > 300]
[1] "Dallas"       "Phoenix"      "Los Angeles"  "Houston"      "Indianapolis"
```

OR: |



```
> cities[sq.miles < 100 | sq.miles > 400]
[1] "Phoenix"     NA            "Los Angeles" "Baltimore"   NA            NA            "Houston"
```