Modu e 3: Data Structures: Vectors and Lists

1d Data Structures in R

- (atomic) vectors
- lists

The Different Vector Modes

```
vector() # an empty 'logical' (the default)
vector
```

```
logical(0)
```

```
vector(mode="character", length = 5) # a
vector of mode 'character' with 5 elements
```

```
[1] "" "" "" "" ""
```

```r
character(5) # the same thing, but using the constructor directly
```

```
[1] "" "" "" "" ""
```

```r
numeric(5)   # a numeric vector with 5 elements
```

```
[1] 0 0 0 0 0
```

```r
logical(5)   # a logical vector with 5 elements
```

```
[1] FALSE FALSE FALSE FALSE FALSE
```

```
x = c(1, 2, 3)
x
```

```
[1] 1 2 3
```

```
1:10
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
seq(10)
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
seq(from = 1, to = 10, by = 0.5)
```

```
 [1]  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5
5.0  5.5  6.0  6.5  7.0  7.5  8.0
[16]  8.5  9.0  9.5 10.0
```

```r
y = c(TRUE, TRUE, FALSE, FALSE)
y
```

```
[1]  TRUE  TRUE FALSE FALSE
```

```r
z = c("Andy", "Ben", "Charlie")
z
```

```
[1] "Andy"    "Ben"     "Charlie"
```

Examining Vectors

```
length(z)
```

```
[1] 3
```

```
class(z)
```

```
[1] "character"
```

```
str(z)
```

```
 chr [1:3] "Andy" "Ben" "Charlie"
```

Adding Elements

```
z = c(z, "Doug")
z
```

```
[1] "Andy"    "Ben"     "Charlie" "Doug"
```

```
z = c("Eric", z)
z
```

```
[1] "Eric"    "Andy"    "Ben"     "Charlie"
"Doug"
```

Accessing Elements by Index

```
z[3]
```

```
[1] "Ben"
```

```
z[2:4]
```

```
[1] "Andy"    "Ben"     "Charlie"
```

```
z[c(1,3)]
```

```
[1] "Eric" "Ben"
```

Accessing Element by Logical Vector

```
c(TRUE, TRUE, FALSE, FALSE, TRUE)
```

```
[1]  TRUE  TRUE FALSE FALSE  TRUE
```

```
z
```

```
[1] "Eric"    "Andy"    "Ben"     "Charlie"
"Doug"
```

```
z[c(TRUE, TRUE, FALSE, FALSE, TRUE)]
```

```
[1] "Eric" "Andy" "Doug"
```

```r
x = c(1, 2, 3, 11, 12, 13)
x < 10
```

```
[1]  TRUE  TRUE  TRUE FALSE FALSE FALSE
```

Exercise

1.

2.

```
x[x < 10]
```

```
[1] 1 2 3
```

```
x[x < 10] = 0
x
```

```
[1]  0  0  0 11 12 13
```

What Happens When You Mix Types Inside a Vector?

```
c(4, "ch")
c(TRUE, 5)
c(FALSE, 100)
c(TRUE, "ch")
```

Answer

```
c(4, "ch")
```

```
[1] "4"  "ch"
```

```
c(TRUE, 5)
```

```
[1] 1 5
```

```
c(FALSE, 100)
```

```
[1]   0 100
```

```
c(TRUE, "ch")
```

```
[1] "TRUE" "ch"
```

character > numeric > logical

```
as.numeric(c("1", "2", "3"))
```

```
[1] 1 2 3
```

```
as.character(1:2)
```

```
[1] "1" "2"
```

```
as.numeric(c("a"))
```

```
[1] NA
```

List

A list is a special type of vector. Each element can be a different type.

Create a list by hand

```
x = list(1, "a", TRUE)
x
```

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

[[2]]
[1] "a"

[[3]]
[1] TRUE
```

Access a list

```
x[[1]]
```

```
[1] 1
```

1. What is the class of x[1]?
2. What about x[[3]]?

Elements of a list can be named (i.e. lists can have the names attribute)

```
my_pie = list(type="key lime", diameter=7,
              is.vegetarian=TRUE)
```

```
my_pie
```

```
$type
[1] "key lime"

$diameter
[1] 7

$is.vegetarian
[1] TRUE
```

```
names(my_pie)
```

```
[1] "type"           "diameter"
"is.vegetarian"
```

```
my_pie$type
```

```
[1] "key lime"
```

```r
dat = data.frame(id = letters[1:5], x = 1:5, y = 16:20)
dat
```

```
  id x  y
1  a 1 16
2  b 2 17
3  c 3 18
4  d 4 19
5  e 5 20
```

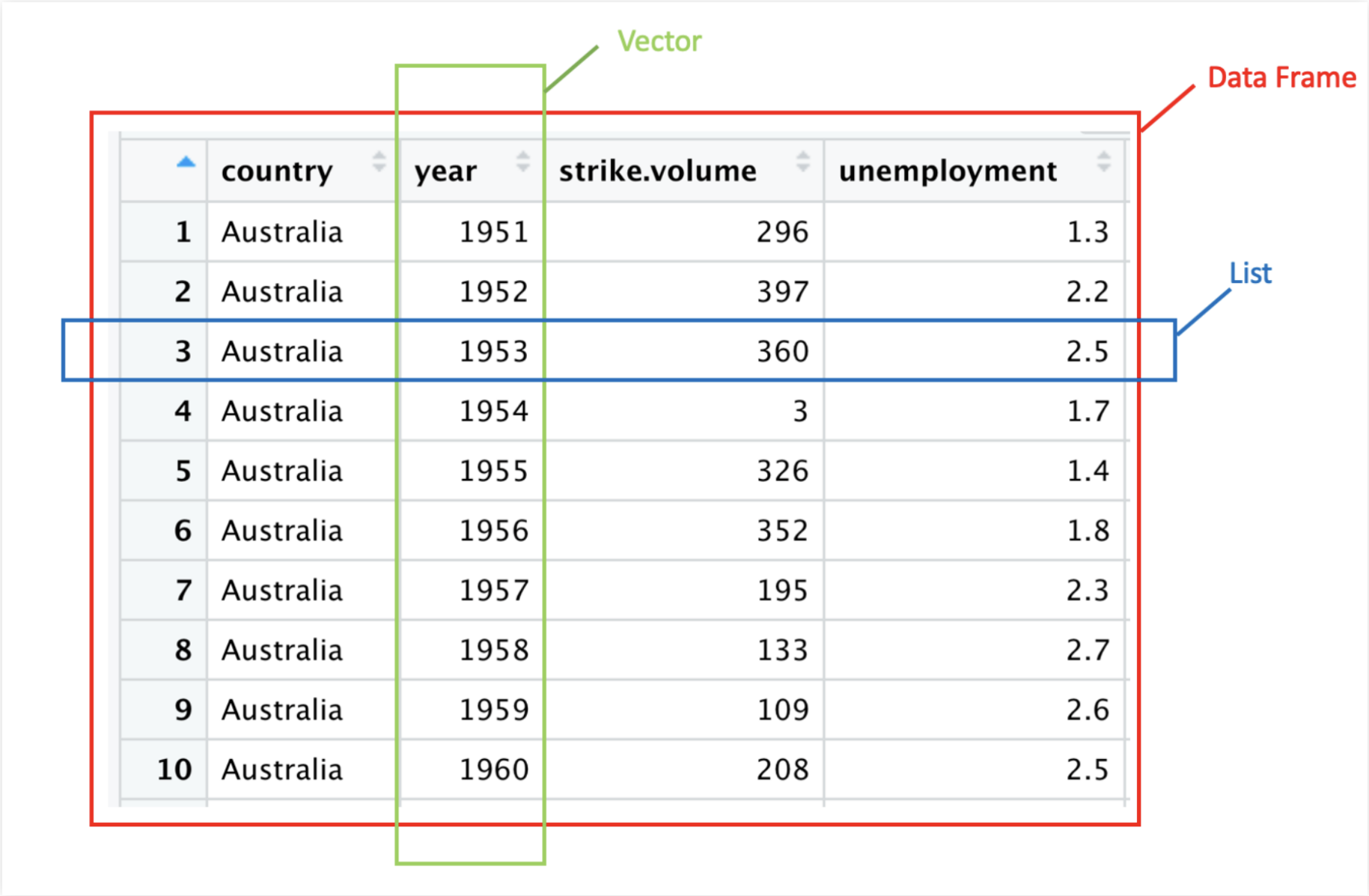
```r
is.list(dat)
```

```
[1] TRUE
```

```r
class(dat)
```

```
[1] "data.frame"
```

Data Frame vs Vector vs List



	country	year	strike.volume	unemployment
1	Australia	1951	296	1.3
2	Australia	1952	397	2.2
3	Australia	1953	360	2.5
4	Australia	1954	3	1.7
5	Australia	1955	326	1.4
6	Australia	1956	352	1.8
7	Australia	1957	195	2.3
8	Australia	1958	133	2.7
9	Australia	1959	109	2.6
10	Australia	1960	208	2.5

Recap

- Can you list different ways to access data in data frame?
- What is the difference between a vector and a list?