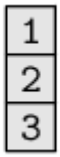
Data Structures in R

There are five basic data structures that are available in *R*.

1. Vectors

A collection of values that all have the same data type. The **elements** of a vector are all numbers, giving a **numeric vector**, or all character values, giving a **character vector**.

A vector can be used to represent a single variable in a data set.



Example:

```
> seq(1, 10)
 [1]  1  2  3  4  5  6  7  8  9 10
```

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

```
> seq(1, 10, by=3)
[1]  1  4  7 10
```

```
> shapes <- rep(c("round", "oval", "long"), 4)
> shapes
 [1] "round" "oval"  "long"  "round" "oval"  "long"  "round" "oval"  "long"
[10] "round" "oval"  "long"
```

```
> patterns <- rep(c("pattern", "plain"), each=3, length=12)
> patterns
 [1] "pattern" "pattern" "pattern" "plain"   "plain"   "plain"   "pattern"
 [8] "pattern" "pattern" "plain"   "plain"   "plain"
```

```
> shades <- rep(c("light", "dark"), each=6)
> shades
 [1] "light" "light" "light" "light" "light" "light" "dark"  "dark"  "dark"
[10] "dark"  "dark"  "dark"
```

2. Factors

A collection of values that all come from a fixed set of possible values. A factor is similar to a vector, except that the values within a factor are limited to a fixed set of possible values.

A factor can be used to represent a categorical variable in a data set.

Examples:

```
> shapesFactor <- factor(shapes, levels=c("round", "oval", "long"))
> shapesFactor
 [1] round oval  long  round oval  long  round oval  long  round oval  long
Levels: round oval long
```

3. Matrices

A two-dimensional collection of values that all have the same type. The values are arranged in rows and columns.

There is also an **array** data structure that extends this idea to more than two dimensions.

Example:

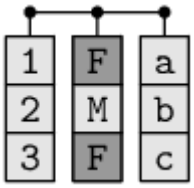
```
> matrix(1:6, ncol=3)
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
> array(1:8, dim=c(2, 2, 2))
, , 1

     [,1] [,2]
[1,]    1    3
[2,]    2    4

, , 2

     [,1] [,2]
[1,]    5    7
[2,]    6    8
```

4. Data frames

A collection of vectors that all have the same length. This is like a matrix, except that each column can contain a different data type.

A data frame can be used to represent an entire data set.



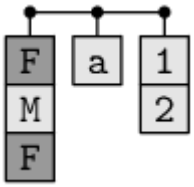
Example:

```
> candy <- data.frame(shapes, patterns, shades, counts)
> candy
   shapes patterns shades counts
1   round  pattern  light      2
2    oval  pattern  light      0
3    long  pattern  light      3
4   round    plain  light      1
5    oval    plain  light      3
6    long    plain  light      2
7   round  pattern   dark      9
8    oval  pattern   dark      0
9    long  pattern   dark      2
10  round    plain   dark      1
11   oval    plain   dark     11
12   long    plain   dark      2
```

5. Lists

A collection of data structures. The **components** of a list can be simply vectors--similar to a data frame, but with each column allowed to have a different length. However, a list can also be a much more complicated structure.

This is a very flexible data structure. Lists can be used to store any combination of data values together.



Example:

```
> dimnames(candy)
[[1]]
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12"

[[2]]
[1] "shapes"   "patterns" "shades"   "counts"
```

```
> list(rownames=rownames(candy), colnames=colnames(candy))
$rownames
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12"

$colnames
[1] "shapes"   "patterns" "shades"   "counts"
```