Vectors

- *R* operates on named data structures. The simplest such structure is the vector, which is a single entity consisting of an ordered collection of numbers or characters.
- The most common types of vectors are:
  - Numeric vectors
  - Character vectors
  - Logical vectors
- There are, of course, other types of vectors.
  - Colour vectors - potentially useful later on.
  - Order vectors - The rankings of items in a vector.
  - Complex number vectors - not part of this course.

Vectors: Creating and editing a vector

- From last class.
- To create a vector, use the assignment operator "=" or ($<-$)and the concatenate function "c()".
- For numeric vectors, the values entered are simply numbers.

```
>x =c(10.4,5.6,3.1,6.4,8.9)
>
```

Vectors: Character & logical vector

- For character vectors, the values are simply characters, specified with quotation marks.
- Single quotation marks

```
Charvec <- c('Dog', 'Cat', 'Shed', 'Spoon')
```

- A logical vectors is a vector whose elements are TRUE, FALSE or NA (i.e. null)

```
Logvec <- c(TRUE, FALSE,TRUE,TRUE )
```

Graphical Data Entry Interface

- The `data.entry()` command calls a spreadsheet graphical user interface, which can be used to edit data. All changes are saved automatically.
- Alternatively, the `edit()` command calls the 'R editor', which can be used to edit specified data or the code used to define that data.

```
x<-edit(x)
```

Vectors: Empty vectors

- Another method of creating vectors is to use the follow
  - `numeric(length = n)`
  - `character (length = n)`
  - `logical (length = n)`
- These commands create empty vectors, of the appropriate kind, of length n.

```
> x<-numeric(4)
> x
[1] 0 0 0 0
```

Vectors: Characteristics

- We can use several *R* commands to gather information about a vector.
  - `length(x)` - how many elements in a vector.
  - `unique(x)` - display each unique item in a vector.
  - `sum(x)` - the sum of the elements in a vector.
  - `prod(x)` - the product of the elements in a vector.
- We can also find statistical information about a vector
  - `summary(x)` - summary statistics of a vector.
  - `mean(x)` - the mean value of a vector.
  - `sd(x)` - the standard deviation of a vector.

Vectors: Characteristics (contd)

```
> mean(x)
[1] 6.375
> sd(y)
[1] 2.858846
>
> median(z)
[1] 16
>
> summary(x)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  3.100   4.975   6.000   6.375   7.400  10.400
```

Calculations using vectors

- Calculations are performed on a vector on a case-wise basis. That is to say, the calculations are carried out on each element individually.

```
> y^2
[1]  2.56 12.25 60.84 44.89 65.61
```

```
x <- c(11.1, 11.54,15.6,17.8,16.9,14.6, 12.7)
y <- c(0.2,    0.6, 0.7, 0.3,0.3,0.5,0.6)
z <- 1:3
```

- Try the following calculations.

```
> y*z
>
> sum(z)
>
> sum(y^2)
>
> sum(y*z)
```

Accessing vector's elements

- The *n*th element of vector 'x' can be accessed by specifying its index when calling 'x'.

  ```
  >x[3]
  [1] 15.6
  ```

- A sequence of elements of vector 'x' can be accessed by specifying the lower and upper bound of the the range, in form x[l:u].

  ```
  > x[2:4]
  ```

Modifying a vector

- A vector can be updated by assigning an extra value to it.

```
> logvec<-c(logvec,TRUE)
> logvec
[1]  TRUE FALSE  TRUE  TRUE  TRUE
```

- A vector can be repeated *n* times using the `rep()` command.

```
> rep(charvec,2)
[1] "blue" "pink" "red" "blue" "pink" "red"
```

- Omitting and deleting the *n*th element of vector 'x'.

```
>charvec[-5]
>charvec <- charvec[-5]
```

Relational operators

A relational operator tests some kind of relation between two entities. For R the relational operators are as follows:

Equals	==	Less or equal to	¡=
Not Equal	!=	Greater than	¿
Less than	¡	Greater than	¿=

Logical operators

- The logical operators are AND, OR and NOT
- if c1 and c2 are logical expressions, then $c1 \& c2$ is their intersection ('AND'), $c1|c2$ is their union ('OR'), and $!c1$ is the negation of c1.

AND	&	also	&&
OR	|	also	||
NOT	!		

Useful Commands For Vectors

```
x  = c(13,16,36,55,23,11)

sort(x )
rev(x)
rep(x ,2)
rep(x ,3)
rep(x ,each=3)
diff(x )
order(x )
rank(x )
```

```
> x  = c(13,16,36,55,23,11)
>
> sort(x )
[1] 11 13 16 23 36 55
>
> rev(x )
[1] 11 23 55 36 16 13
```

```
> rep(x ,2)
 [1] 13 16 36 55 23 11 13 16 36 55 23 11
> rep(x ,3)
 [1] 13 16 36 55 23 11 13 16 36 55 23 11 13 16 36 55 23 1
>
> rep(x ,each=3)
 [1] 13 13 13 16 16 16 36 36 36 55 55 55 23 23 23 11 11 1
```

```
> diff(x )
[1]   3  20  19 -32 -12
>
> order(x )
[1] 6 1 2 5 3 4
>
> rank(x )
[1] 2 3 5 6 4 1
```

Using the colon operator

A ‘count-up’ or a ‘count-down’ will be determined automatically.

```
1:20
20:1
10:20
```

Using the seq() operator

Firstly we will mimic the sequences that we have created using the colon operator.

```
seq(1,20)
seq(20,1)
```

Examples using operators

We can use relational and logical operators to selecting elements of a vector with specified criteria.

```
x <- 1:12

#selecting all elements of x greater than 5
x[x>5]

#selecting all elements of x greater or equal to than 5
x[x>=5]

#selecting all elements of x greater than 5 #or less than
x[(x>5)|(x<3)]

#selecting all elements of x between 3 and 5
x[(x>3)&(x<5)]
```

Data Selection and manipulation

- `sort(x)` : sorts the object x in ascending order.
- `rev(x)` : reverses the order of x without sorting it