Exercise 2.2

User generated functions

useful commands: `function(){}`, `is.na()`, `which()`, `length()`, `floor()`, `median()`, `%%` (remainder), `month.name`

To make a function, we use:
my_function = function (arglist) {body}

01*. create a function that turns 0's in a numeric vector into NA's

```
zero_to_NA = function(v){
  v_new = v
  v_new[v==0] = NA
  return(v_new)
}
```

test your function by creating a vector that contains zeros and calling the function

```
test1 = c(1,0,0,3)
zero_to_NA(test1)
```

```
## [1]  1 NA NA  3
```

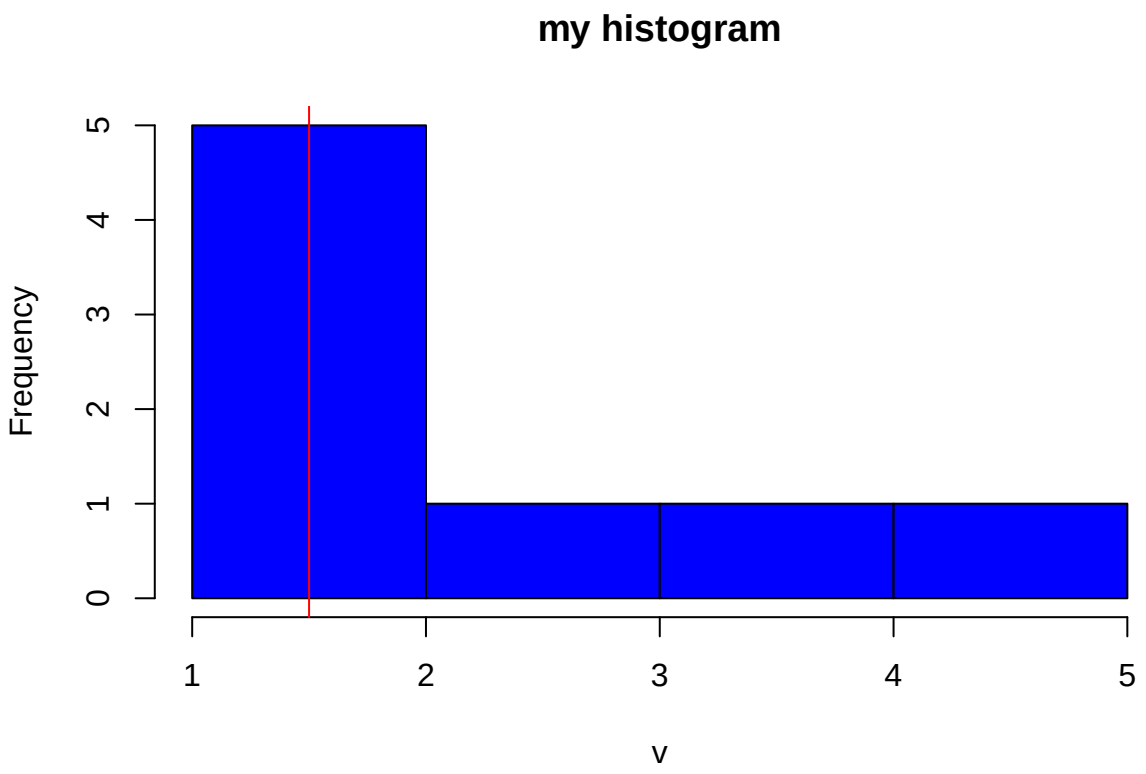
02*. create a function that plots a histogram of a vector and also adds a vertical line at the median

test your function by creating a numeric vector and calling the function

BONUS: use ... to pass optional arguments to your function (e.g. a title or color for your histogram)

```
medHist = function(v, ...){ #the name and arguments specified
  #additionally, we added ... to accept any other graphical arguments
  hist(v, ...) # we pass the optional graphical parameters to hist() using ...
  abline(v = median(v), col = "red")
}

medHist(c(1,1,1,1,2,3,4,5), col = "blue", main = "my histogram") # testing function
```



BONUS 03. create a function that when given a month name

```
- prints the subsequent month's names
  - i.e. `nextMonth("January")` returns `"February"`
- hint: the vector of month names is built into R .. try typing `month.name` in your console
- make sure your function works for December!
```

```
nextMonth = function(m){
  index = (which(month.name == m) + 1)%%12
  # %% gets the remainder from dividing by 12 .. so '13' becomes 1 (January)
  new_month = month.name[index]
  return(new_month)
}
```

Can also use if statements

```
nextMonth = function(m){
  index = (which(month.name == m) + 1)
  if(index == 13) {
    return(month.name[1])
  }
  if(index != 13) {
    new_month = month.name[index]
  }
}

nextMonth("February") # testing
nextMonth("December") # testing -> round corner back to January
```

```
## [1] "January"
```

note: can use `sapply()` to go over vector of more than length 1 (i.e. multiple months)

```
sapply(c("February", "December"), nextMonth) #returns "March", "January"
```

```
##  February  December
##   "March" "January"
```