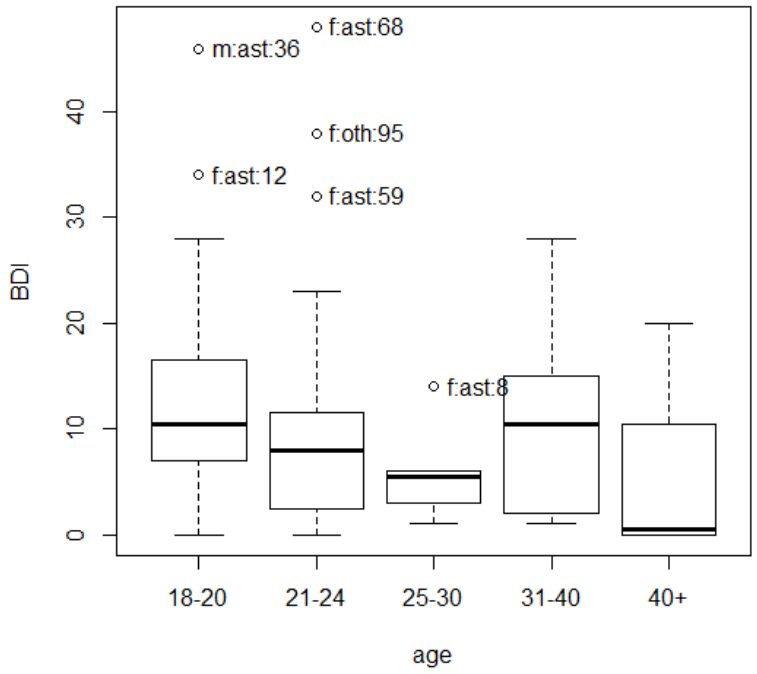
Stat 462: Lab 1 Solutions

```
library( DAAG )
library( MASS )

# 1.2
# Create a new data frame by extracting the rows 1,2,4,11,13,18
  orings.new <- orings[ c(1,2,4,11,13,18), ]
# Set the graphics device to accept two plots in one row
  par( mfrow = c(1,2) )
# Plot the data used for deciding whether to launch
  plot( total ~ temperature, data = orings.new, main = "Data used" )
# Plot all of the data
  plot( total ~ temperature, data = orings, main = "All data" )


# 1.3
# Use str() to get information about possum
  str( possum )
# Determine which rows have one or more values missing
# and in which columns the missing values appear
  possum[ !complete.cases(possum), ]


# 1.10
# Evaluate the expression
  1000*((1+0.075)^5 - 1)
# Modify expression for 3.5% p.a.
  1000*((1+0.035)^5 - 1)
# Change exponent to seq(1,10)
  1000*((1+0.075)^seq(1,10) - 1)
  # This is the cumulative interest earned each year
  # for the next ten years.


# 1.11
  gender <- factor( c( rep("female", 91), rep("male", 92) ) )
  table( gender )
  # As expected, the table gives the frequency distribution of
  # females and males, with females first since they appear
  # first in the vector.
  gender <- factor( gender, levels = c( "male", "female" ) )
  table( gender )
  # This time the table gives the frequency distribution of
  # females and males, but with males first because they are
  # associated with the first factor level.
  gender <- factor( gender, levels = c("Male", "female" ) )
    # Note the mistake: "Male" should be "male"
  table( gender )
  # In this case, there are no "Male" counts. Remember that R
```

```
  # is case-sensitive, so "Male" and "male" are different
  table( gender, exclude = NULL )
  # This table shows the number of elements of gender that are
  # not associated with either Male or female (so the male count)
  # see ?table
  rm( gender )


# 1.13
# As the power of the transformation decreases towards
# zero (the log transform), the distribution of both
# brain and body becomes more uniform.
# For larger powers, we see that there are a few very
# large values of both brain and body compared to the rest
# of the data. For the log-log plot, we see that there is
# a generally positive relationship between log(body) and
# log(brain).
  par( mfrow=c(2,2) ) # 2 by 2 layout on the page
  library( MASS ) # Animals is in the MASS package
  plot( brain ~ body, data = Animals )
  plot( sqrt( brain ) ~ sqrt( body ), data = Animals )
  plot( I( brain^0.1 ) ~ I( body^0.1 ), data = Animals )
  # I() forces its argument to be treated "as is"
  plot( log( brain ) ~ log( body ), data = Animals )
  par( mfrow=c(1,1) ) # Restore to 1 figure per page


# 1.16

# From the previous question 1.15...
  plot( BDI ~ age, data = socsupport )

  gender1 <- with( socsupport, abbreviate( gender, 1 ) )
  table( gender1 ) # Examine the result
  country3 <- with( socsupport, abbreviate( country, 3 ) )
  table( country3 ) # Examine the result

  num <- with( socsupport, seq( along=gender ) )
  # Generate row numbers
  lab <- paste( gender1, country3, num, sep = ":" )
  # Now use identify to place labels on the outlying points
  with( socsupport, identify( age, BDI, labels = lab ) )
  # Press escape when finished...
```




```
# 1.19
  vltcv <- stack( vlt[, 1:3] )
  head( vltcv )
  table( vltcv )
  # All windows look different, in particular window 2
  # which has far fewer 1's and correspondingly more
  # 2's, 3's, and 4's

# 1.21
  # Set up graphics device for 2x4 plots, saving the old
  # graphics parameters
  oldpar <- par( mfrow = c(2,4) )
  # instead of using an explicit loop, use sapply() to
  # apply the plotting function over each column of austpop
  # Creating a new plotting function is easiest
  plotpop <- function( i, ap ){
                plot( ap[,1], log(ap[,i]), xlab = "Year"
                    , ylab = names(ap)[i], pch = 16, ylim = c(0,10) )
              }
  sapply( 2:9, FUN = plotpop, ap = austpop )

  # Compare to for loop
  for( i in 2:9 ){
    plot( austpop[,1], log( austpop[,i] ), xlab = "Year"
        , ylab = names(austpop)[i], pch=16, ylim=c(0,10) )
  }

  # NOTE: The hint in the book does not allow you to change
  # the y-axis labels, because sapply strips the names from
  # the vectors when it passes them to FUN.
```