Practical:

1. Solve the following equation and assign the results to an object called tot. Create a new vector of 5 elements called aa containing the following numbers, 1,2,3,4,5. Confirm that that object is a vector and find its length. Add tot with aa. Is the result what you expected?

$$\frac{3+1}{5^2} - 10\left(\frac{3}{20}+1\right)^3$$

```
tot<-(-15.04875)  #pay attention to the bracket

aa<-c(1,2,3,4,5)

is.vector(aa)

length(aa)

tot+aa
```

2. Create a vector called v1 containing numbers from 1:100.

```
v1<-1:100
```

3. Create a vector called v2 containing only even numbers from 1 to 200

```
v2<-seq(2,200,by=2)
```

4. check the length of each of the vectors and make sure it matches

```
length(v1);lenght(v2)
```

5. Confirm that both objects are, in fact, vectors.

```
is.vector(v1); is.vector(v2)
```

6. Create a dataframe called "dat" with vectors v1 and v2 as columns

```
dat<-as.data.frame(cbind(v1,v2))
```

7. Change the names of the columns "v1" and "v2" to "Sugar" and "Ray".

```
names(dat)<-c("Sugar","Ray")
```

8. Check the column names of your dataframe and extract element 88 from column Sugar, elements 34 to 45 from columns Ray and elements 8 to 10 from both columns. Repeat the exercise but access the elements using the "attach" function.

```
dat[88,1];dat[34:45,2];dat[8:10,]

Sugar[88];Ray[34:45];Sugar[8:10];Ray[8:10]
```

9. Create a new data frame called "newdat" containing only those rows where the value for Sugar is less than 20.

```
attach(dat)
```

```
newdat<-dat[Sugar<20,]
```

10: check that the new data frame is, in fact, a dataframe. If not, make it so. Explore the structure of the dataframe.

Assign letters of the alphabet in order as rownames to your dataframe.

```
is.data.frame(newdat)
```

```
str(newdat)
```

```
row.names(newdat)<-letters[1:length(newdat$Sugar)]
```

11. Extract the value corresponding to rowname "g" for Sugar.

newdat["g","Sugar"] or newdat[7,1]

12: Check how many objects you have created and delete all objects created so far

```
ls()
```

```
rm(list=ls())
```

13: call the "iris" dataset by using data(iris). Identify what class of object "iris" is, what the dimensions are, what the column and row names are and explore the structure of the dataset.

```
class(iris)
```

```
dim(iris)
```

```
names(iris)
```

```
row.names(iris)
```

14. Extract the header with the first 6 rows of the dataset "iris"

```
head(iris)
```

15. Create a new object called "sepal" that contains only the sepal measurements and the species from the larger dataset "iris" (check the column names).

```
sepal <- iris[,c(1,2,5)]
```

16. Create a subset of data for flowers with sepal width less than 3 using a vector of TRUE and FALSE

```
sort <- (iris$Sepal.Width <3)
```

```
smallv<- iris[small,]
```

OR

iris.sm.sw.2 = iris[iris$Sepal.Width <3,] #without using a vector of TRUE and FALSE

17. get the data for just the versicolor flowers and save it to an object called "versicol" using a TRUE and FALSE vector and obtain a summary of the data.

versi = (iris$Species=="versicolor")

versicol = iris[versi,]

summary(versicol)

18. The results from a predictive model are stored in an object called "alpha" (after you have run the following.)

data(iris)

attach(iris)

alpha<-summary(lm(Sepal.Length~Sepal.Width))

explore the object "alpha", what kind of object is it? Extract a vector of residuals and save it as a new object called "res" . Extract the Std.Error for the slope and intercept (found under coefficient).

is.vector(alpha) || is.list(alpha) # Is a vector, more specifically a list.

Part 2

res<-alpha$residuals

or

res<-alpha[[3]]

part 3

alpha$coefficients[3:4]