Practical 6:

1. Functions and their importance: A function is a set of statements organized together to perform a specific task. R has a large number of in-built functions and the user can create their own functions.

In R, a function is an object so the R interpreter is able to pass control to the function, along with arguments that may be necessary for the function to accomplish the actions.

The function in turn performs its task and returns control to the interpreter as well as any result which may be stored in other objects.

Importance: A programmer builds a function to avoid repeating the same task, or reduce complexity.

A function should be written to carry out a specified a tasks may or may not include arguments contain a body may or may not return one or more values.

2. Syntax of functions in R: An R function is created by using the keyword function. The basic syntax of an R function definition is as follows –

```
function_name <- function(arg_1, arg_2, ...) {
   Function body
}
```

3. Function with arguments:

```
> #function in Argument..
> func <- function(a) {
+     for(i in 1:a) {
+        b <- i^2
+        print(b)
+     }
+ }
>
> func(6)
[1] 1
[1] 4
[1] 9
[1] 16
[1] 25
[1] 36
>
```

```
> #function in Argument..
> func <- function(a,b) {
+      c<-a+b
+     return (c)
+ }
>
> print(func(6,7))
[1] 13
>
```

4. Function without arguments:

```
> #function without Arguments..
> fact <- function() {
+      a<-5
+      c<-1
+      while(a!=0){
+          c<-c*a
+          a<-a-1
+      }
+      return (c)
+ }
>
> print(fact())
[1] 120
>
```

```
> #function without Arguments..
> mean <- function() {
+     a<-5
+     c<-0
+     while(a!=0){
+         c<-c+a
+         a<-a-1
+     }
+     return (c/5)
+ }
>
> paste("Mean of 1 to 5:",mean())
[1] "Mean of 1 to 5: 3"
>
```

5. Calling a Function with Argument Values (by position and by name):

(i) By Position:

```
> fun <- function(a,b,c) {
+     d<-(a*b)+c
+     return (d)
+ }
>
> paste(fun(1,2,3))
[1] "5"
>
```

(ii)By Name:

```
> fun <- function(a,b,c) {
+     d<-(a*b)+c
+     return (d)
+ }
>
> paste(fun(b=2,c=5,a=1))
[1] "7"
>
```

6. Calling a Function with Default Argument:

```
> fun <- function(a=1,b=2,c=50) {
+      d<-(a*b)+c
+      return (d)
+ }
>
> paste(fun(a=2,b=1))
[1] "52"
>
```

7. Functions with return:

```
> fibb <- function(a) {
+      if(a==1){
+          return (0);
+      }
+      if(a==2){
+          return (1);
+      }
+      return (fibb(a-1)+fibb(a-2))
+ }
>
> paste("Fibonacci Value of 5 is",fibb(8))
[1] "Fibonacci Value of 5 is 13"
>
```