SESSION 5: Data Management Using R Assignment 3

Problem Statement

1. Test whether two vectors are exactly equal (element by element)

vec1 = c(rownames(mtcars[1:15,]))

vec2 = c(rownames(mtcars[11:25,]))

`identical(vec1,vec2)`

2. Sort the character vector in ascending order and descending order

vec1 = c(rownames(mtcars[1:15,]))

vec2 = c(rownames(mtcars[11:25,]))

sort(vec1, decreasing = TRUE)

sort(vec2, decreasing = TRUE)

sort(vec1, decreasing = FALSE)

sort(vec2, decreasing = FALSE)

3.What is the major difference between str c() and paste() show an example.

Compactly display the internal **str**ucture of an R object, a diagnostic function and an alternative to `summary` (and to some extent, `dput`). Ideally, only one line for each 'basic' structure is displayed. It is especially well suited to compactly display the (abbreviated) contents of (possibly nested) lists. The idea is to give reasonable output for **any** R object. It calls `args` for (non-primitive) function objects.
`strOptions()` is a convenience function for setting `options(str = .)`, see the examples.

# NOT RUN {

require(stats); require(grDevices); require(graphics)

## The following examples show some of 'str' capabilities

str(1:12)

str(ls)

str(args) #- more useful than args(args) !

str(freeny)

```
str(str)
str(.Machine, digits.d = 20) # extra digits for identification of binary numbers
str( lsfit(1:9, 1:9))
str( lsfit(1:9, 1:9), max.level = 1)
str( lsfit(1:9, 1:9), width = 60, strict.width = "cut")
str( lsfit(1:9, 1:9), width = 60, strict.width = "wrap")
op <- options(); str(op)   # save first;
                           # otherwise internal options() is used.
```

`paste` converts its arguments (*via* `as.character`) to character strings, and concatenates them (separating them by the string given by `sep`). If the arguments are vectors, they are concatenated term-by-term to give a character vector result. Vector arguments are recycled as needed, with zero-length arguments being recycled to `""`.
Note that `paste()` coerces `NA_character_`, the character missing value, to `"NA"` which may seem undesirable, e.g., when pasting two character vectors, or very desirable, e.g.in `paste("the value of p is ", p)`.
`paste0(…, collapse)` is equivalent to `paste(…, sep = "", collapse)`, slightly more efficiently.
If a value is specified for `collapse`, the values in the result are then concatenated into a single string, with the elements being separated by the value of `collapse`.

```
# NOT RUN {
## When passing a single vector, paste0 and paste work like as.character.
paste0(1:12)
paste(1:12)        # same
as.character(1:12) # same
```

4. Introduce a separator when concatenating the strings

```
str1 = 'Hello'
str2 = 'World!'
str1
result = paste(str1,str2)
result
print (result)
```