Bài tập: tóm tắt trang 1 -> 20 ( Intro\_to\_R\_Vietnamese)

1. **Giới thiệu R**

* Phân tích số liệu và biểu đồ thường được tiến hành bằng các phần mềm thông dụng như SAS, SPSS, Stata, Statistica, và S-Plus. Đây là những phần mềm được các công ti phần mềm phát triển và giới thiệu trên thị trường khoảng ba thập niên qua, và đã được các trường đại học, các trung tâm nghiên cứu và công ti kĩ nghệ trên toàn thế giới sử dụng cho giảng dạy và nghiên cứu.
* Nhưng vì chi phí sử dụng những phần mềm này rất đắt, không có khả năng sử dụng lâu dài nên các nhà nghiên cứu thống kê trên thế giới đã phát triển một phần mềm mới với chủ trương mã nguồn mở, sử dụng thống nhất và hoàn toàn miễn phí
* Năm 1996, trong một bài báo quan trọng về tính toán thống kê, hai nhà thống kê học Ross Ihaka và Robert Gentleman đã phát họa một ngôn ngữ mới cho phân tích thống kê và đặt tên là R.
* Cho đến nay, qua chưa đầy 10 năm phát triển, càng ngày càng có nhiều nhà thống kê học, toán học, nghiên cứu trong mọi lĩnh vực đã chuyển sang sử dụng R để phân tích dữ liệu khoa học.
* R là một phần mềm sử dụng cho phân tích
* thống kê và vẽ biểu đồ. Thật ra, về bản chất, R là ngôn ngữ máy tính đa năng, có thể sử dụng cho nhiều mục tiêu khác nhau, từ tính toán đơn giản, toán học giải trí (recreational mathematics), tính toán ma trận (matrix), đến các phân tích thống kê phức tạp. Vì là một ngôn ngữ, cho nên người ta có thể sử dụng R để phát triển thành các phần mềm chuyên môn cho một vấn đề tính toán cá biệt.

1. **Tải và cài đặt R (đã làm)**
2. **“Văn phạm” R**

* R là một ngôn ngữ tương tác (interactive language), có nghĩa là khi chúng ta ra lệnh, và nếu lệnh theo đúng “văn phạm”, R sẽ “đáp” lại bằng một kết quả. Và sự tương tác tiếp tục cho đến khi chúng ta đạt được yêu cầu.
* “Văn phạm” chung của R là một lệnh (command hoặc function) có cú pháp chung là:

Đối tượng <- hàm(thông số 1, thông số 2, … thông số n)

* Để biết một hàm có những thông số nào, ta dùng lệnh args(x) trong đó x là một hàm mà chúng ta cần biết:

> args(read.csv)

function (file, header = TRUE, sep = ",", quote = "\"", dec = ".", fill = TRUE, comment.char = "", ...)

NULL

* Ngoài lệnh args() R còn cung cấp lệnh help() để người sử dụng có thể hiểu “văn phạm” của từng hàm. Chẳng hạn như muốn biết hàm lm có những thông số (arguments) nào, chúng ta chỉ đơn giản lệnh:

> help(lm)

hay

> ?lm

* Một cửa sổ sẽ hiện ra bên phải của màn hình chỉ rõ cách sử dụng ra sao và thậm chí có cả ví dụ. Bạn đọc có thể đơn giản copy và dán ví dụ vào R để xem cách vận hành.
* R là một ngôn ngữ “đối tượng” (object oriented language). Điều này có nghĩa là các dữ liệu trong R được chứa trong object. Định hướng này cũng có vài ảnh hưởng đến cách viết của R. Chẳng hạn như thay vì viết x = 5 như thông thường chúng ta vẫn viết, thì R yêu cầu viết là x == 5.
* Đối với R, x = 5 tương đương với x <- 5. Cách viết sau (dùng kí hiệu <-) được khuyến khích hơn là cách viết trước (=).

Vd: x <- rnorm(10)

Một số kí hiệu hay dùng trong R là:

|  |  |
| --- | --- |
| **Kí hiệu** | **Ý nghĩa** |
| x == 5 | X bằng 5 |
| x != 5 | X không bằng 5 |
| x < y | X nhỏ hơn Y |
| x > y | X lớn hơn Y |
| z <= 7 | Z nhỏ hơn hoặc bằng 7 |
| p >= 1 | P lớn hơn hoặc bằng 1 |
| is.na(x) | Có phải x kaf biến số trống không (missing value) |
| A & B | A và B (AND) |
| A | B | A hoặc B (OR) |
| ! | Không là (NOT) |

* Với R, tất cả các câu chữ hay lệnh sau kí hiệu # đều không có hiệu ứng, vì # là kí hiệu dành cho người sử dụng thêm vào các ghi chú, ví dụ:

> # lệnh sau đây sẽ mô phỏng 10 giá trị normal

> x <- rnorm(10)

1. **Cách đặt tên trong R**

* Đặt tên một đối tượng (object) hay một biến số (variable) trong R khá linh hoạt, vì R không có nhiều giới hạn như các phần mềm khác. Tên một object phải được viết liền nhau (tức không được cách rời bằng một khoảng trống). Chẳng hạn như R chấp nhận “myobject” nhưng không chấp nhận “my object”.

Vd:

> myobject <- rnorm(10)

> my object <- rnorm(10)

Error: syntax error in "my object"

* Một điều quan trọng cần lưu ý là R phân biệt mẫu tự viết hoa và viết thường. Cho nên “My.object” khác với “my.object”.

Ví dụ:

> My.object.u <- 15

> my.object.L <- 5

> My.object.u + my.object.L

[1] 20

* **Một vài điều cần lưu ý khi đặt tên trong R là:**
* Không nên đặt tên một biến số hay variable bằng kí hiệu “\_” (underscore) như “my\_object” hay “my-object”.
* Không nên đặt tên một object giống như một biến số trong một dữ liệu. Ví dụ, nếu chúng ta có một data.frame (dữ liệu hay dataset) với biến số age trong đó, thì không nên có một object trùng tên age, tức là không nên viết: age <- age. Tuy nhiên, nếu data.frame tên là data thì chúng ta có thể đề cập đến biến số age với một kí tự $ như sau: data$age. (Tức là biến số age trong data.frame data), và trong trường hợp đó, age <- data$age có thể chấp nhận được.

1. **Cách nhập dữ liệu vào R**

* Muốn làm phân tích dữ liệu bằng R, chúng ta phải có sẵn dữ liệu ở dạng mà R có thể hiểu được để xử lí. Dữ liệu mà R hiểu được phải là dữ liệu trong một data.frame. Có nhiều cách để nhập số liệu vào một data.frame trong R, từ nhập trực tiếp đến nhập từ các nguồn khác nhau. Sau đây là những cách thông dụng nhất:

1. **Nhập số liệu trực tiếp: c ()**

Vd: chúng ta có số liệu về độ tuổi và insulin cho 10 bệnh nhân như sau:

|  |  |
| --- | --- |
| 50 | 16.5 |
| 62 | 10.8 |
| 60 | 32.3 |
| 40 | 19.3 |
| 48 | 14.2 |
| 47 | 11.3 |
| 57 | 15.5 |
| 70 | 15.8 |
| 48 | 16.2 |
| 67 | 11.2 |

* Chúng ta có thể sử dụng function có tên c như sau:

> age <- c(50,62, 60,40,48,47,57,70,48,67)

> insulin <- c(16.5,10.8,32.3,19.3,14.2,11.3,15.5,15.8,16.2,11.2)

* Lệnh thứ nhất cho R biết rằng chúng ta muốn tạo ra một cột dữ liệu có tên là age, và lệnh thứ hai là tạo ra một cột khác có tên là insulin. Tất nhiên, chúng ta có thể lấy một tên khác mà mình thích.
* Chúng ta dùng function c để nhập dữ liệu. Chú ý rằng mỗi số liệu cho mỗi bệnh nhân được cách nhau bằng một dấu phẩy.
* Kí hiệu insulin <- (cũng có thể viết là insulin =) có nghĩa là các số liệu theo sau sẽ có nằm trong biến số insulin. Chúng ta sẽ gặp kí hiệu này rất nhiều lần trong khi sử dụng R.
* R là một ngôn ngữ cấu trúc theo dạng đối tượng (thuật ngữ chuyên môn là “object-oriented language”), vì mỗi cột số liệu hay mỗi một data.frame là một đối tượng (object) đối với R. Vì thế, age và insulin là hai đối tượng riêng lẻ. Bây giờ chúng ta cần phải nhập hai đối tượng này thành một data.frame để R có thể xử lí sau này. Để làm việc này chúng ta cần đến function data.frame:

> tuan <- data.frame(age, insulin)

* Trong lệnh này, chúng ta muốn cho R biết rằng nhập hai cột (hay hai đối tượng) “age” và “insulin” vào một đối tượng có tên là “tuan”.
* Đến đây thì chúng ta đã có một đối tượng hoàn chỉnh để tiến hành phân tích thống kê. Để kiểm tra xem trong tuan có gì, chúng ta chỉ cần đơn giản gõ:

> tuan

Và R sẽ báo cáo:

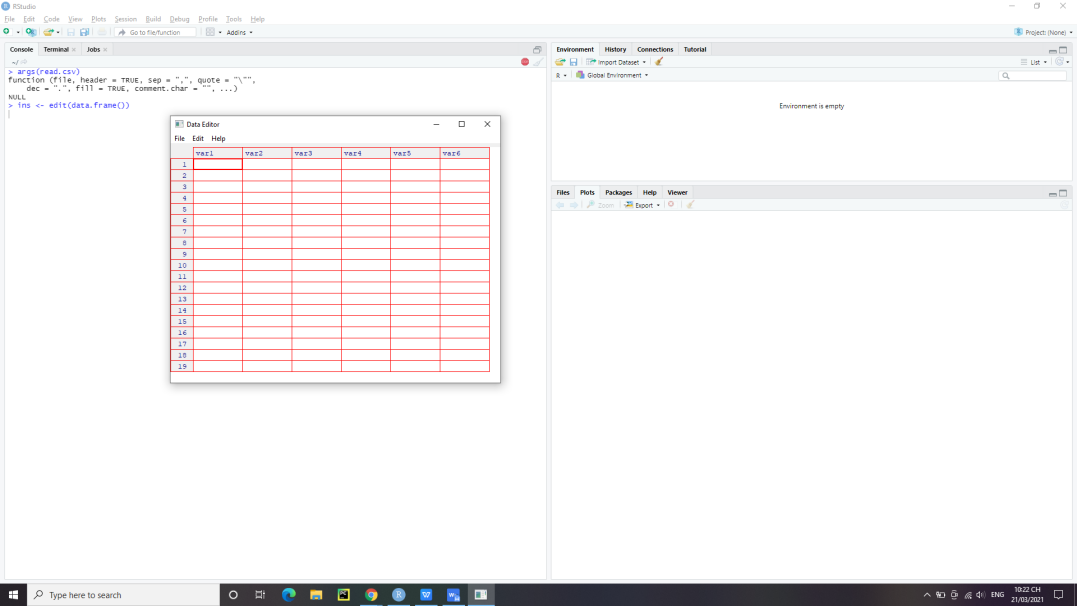
|  |  |  |
| --- | --- | --- |
|  | age | insulin |
| 1 | 50 | 16.5 |
| 2 | 62 | 10.8 |
| 3 | 60 | 32.3 |
| 4 | 40 | 19.3 |
| 5 | 48 | 14.2 |
| 6 | 47 | 11.3 |
| 7 | 57 | 15.5 |
| 8 | 70 | 15.8 |
| 9 | 48 | 16.2 |
| 10 | 67 | 11.2 |

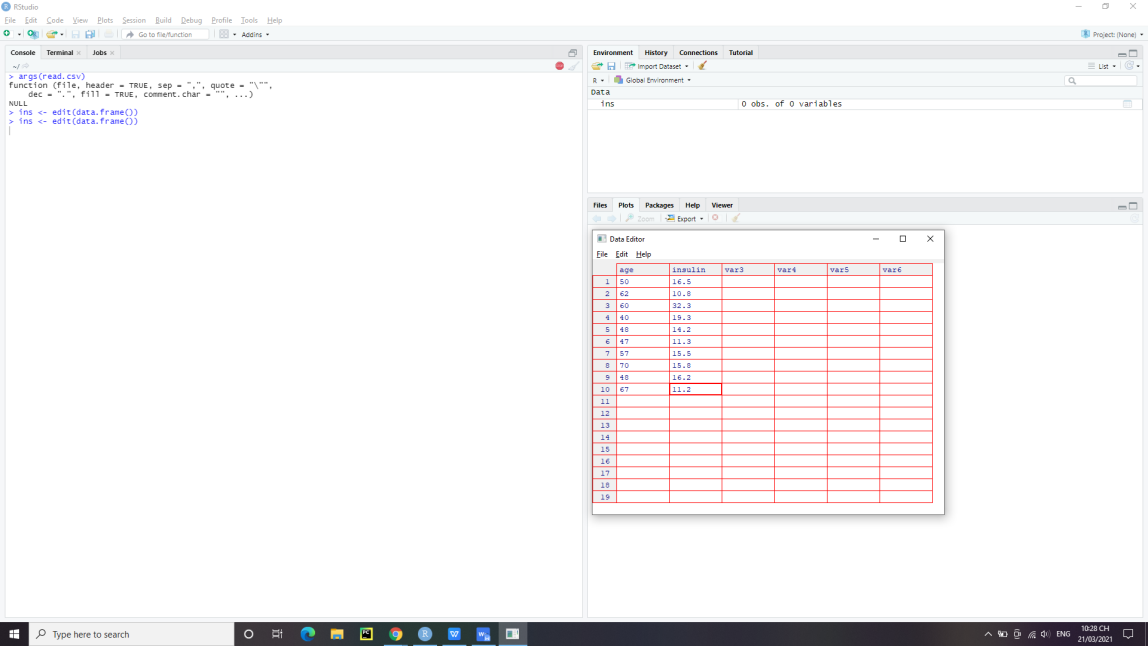
1. **Nhập số liệu trực tiếp: edit(data.frame())**

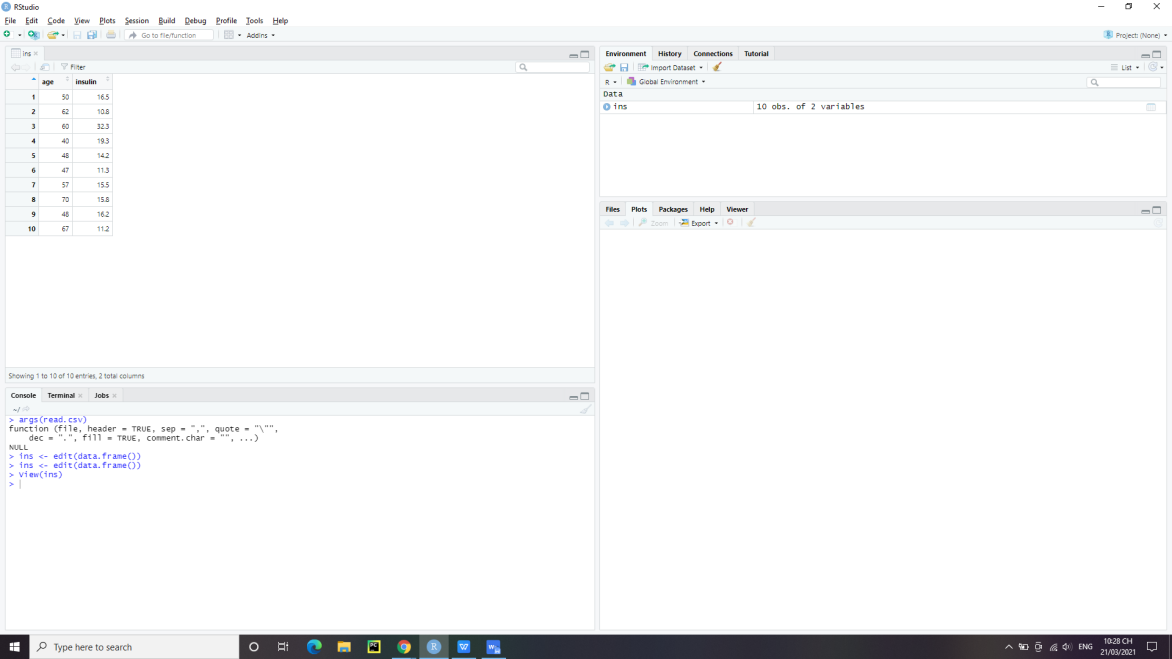
* Ví dụ 1 (tiếp tục): chúng ta có thể nhập số liệu về độ tuổi và insulin cho 10 bệnh nhân bằng một function rất có ích, đó là: edit(data.frame()). Với function này, R sẽ cung cấp cho chúng ta một window mới với một dãy cột và dòng giống như Excel, và chúng ta có thể nhập số liệu trong bảng đó. Ví dụ:

> ins <- edit(data.frame())

* R sẽ hiển thị một cửa sổ như hình dưới:



* Ở đây, R không biết chúng ta có biến số nào, cho nên R liệt kê các biến số var1, var2, v.v… Nhấp chuột vào cột var1 và thay đổi bằng cách gõ vào đó age. Nhấp chuột vào cột var2 và thay đổi bằng cách gõ vào đó insulin. Sau đó gõ số liệu cho từng cột.
* Sau khi xong, bấm nút chéo X ở góc phải của spreadsheet, chúng ta sẽ có một data.frame tên ins với hai biến số age và insulin.



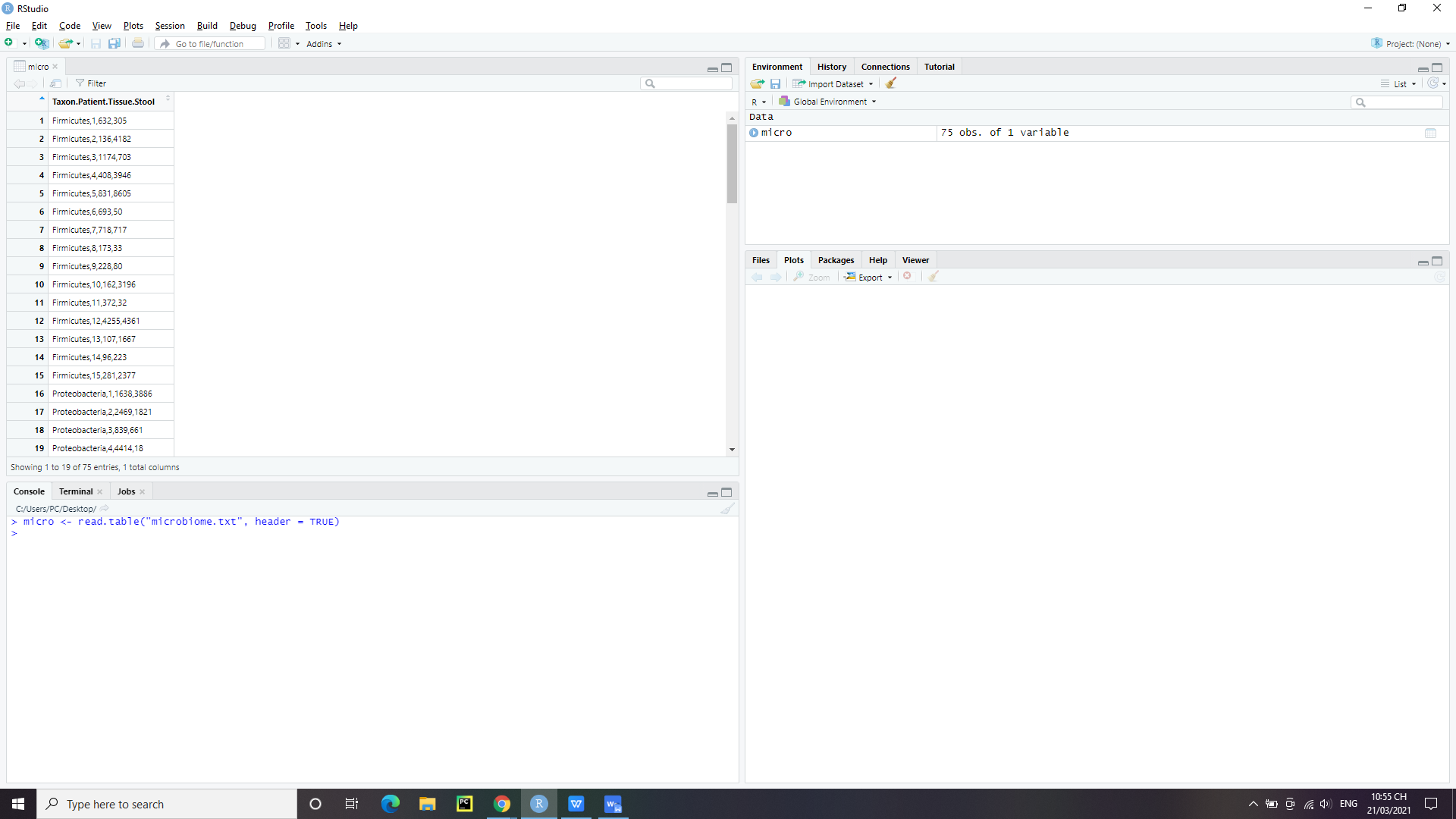
1. **Nhập số liệu từ một text file: read.table**

Vd: Cho file microbiome.txt. Nhập dữ liệu này vào R

* Chúng ta sẽ sử dụng lệnh read.table như sau:

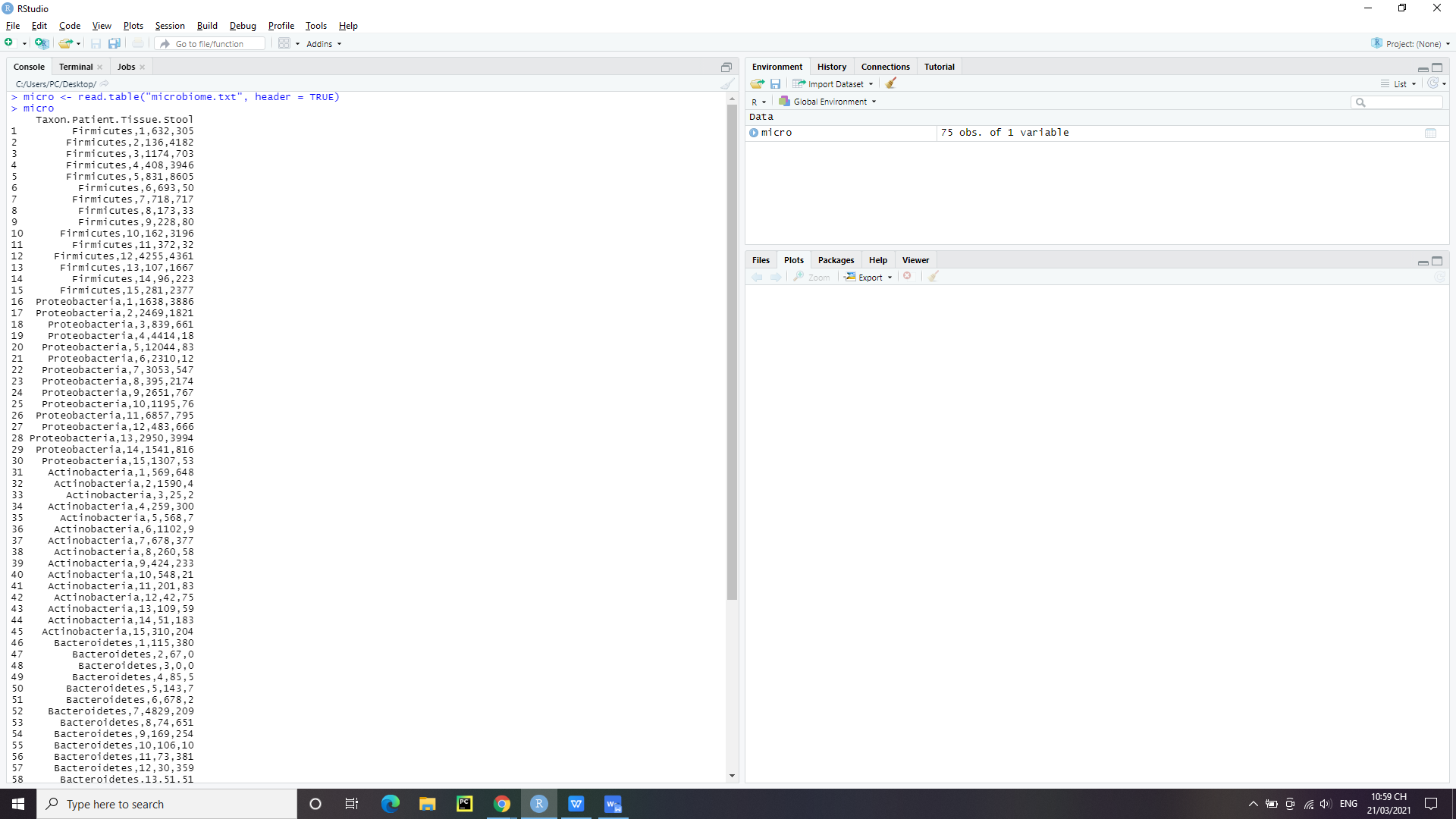
> setwd(“C:/Users/PC/Desktop”)

> micro <- read.table(“microbiome.txt”, header= TRUE)



* Lệnh thứ nhất chúng ta muốn đảm bảo R truy nhập đúng directory mà số liệu đang được lưu giữ. Lệnh thứ hai yêu cầu R nhập số liệu từ file có tên là “microbiome.txt” (trong directory C:/Users/PC/Desktop) và cho vào đối tượng micro. Trong lệnh này, header=TRUE có nghĩa là yêu cầu R đọc dòng đầu tiên trong file đó như là tên của từng cột dữ kiện.
* Chúng ta có thể kiểm tra xem R đã đọc hết các dữ liệu hay chưa bằng cách ra lệnh:

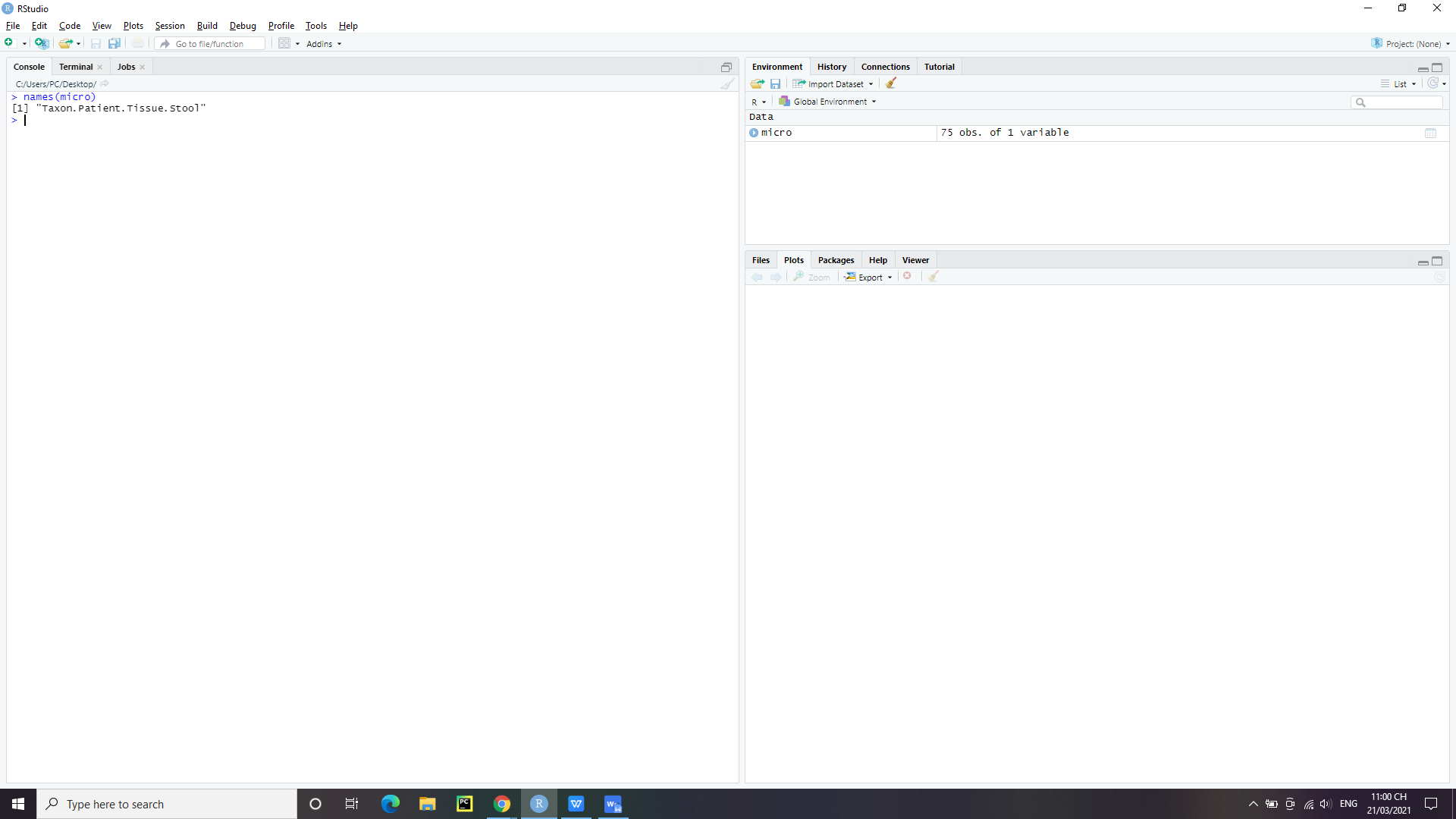
> micro



Hay

> names(micro)

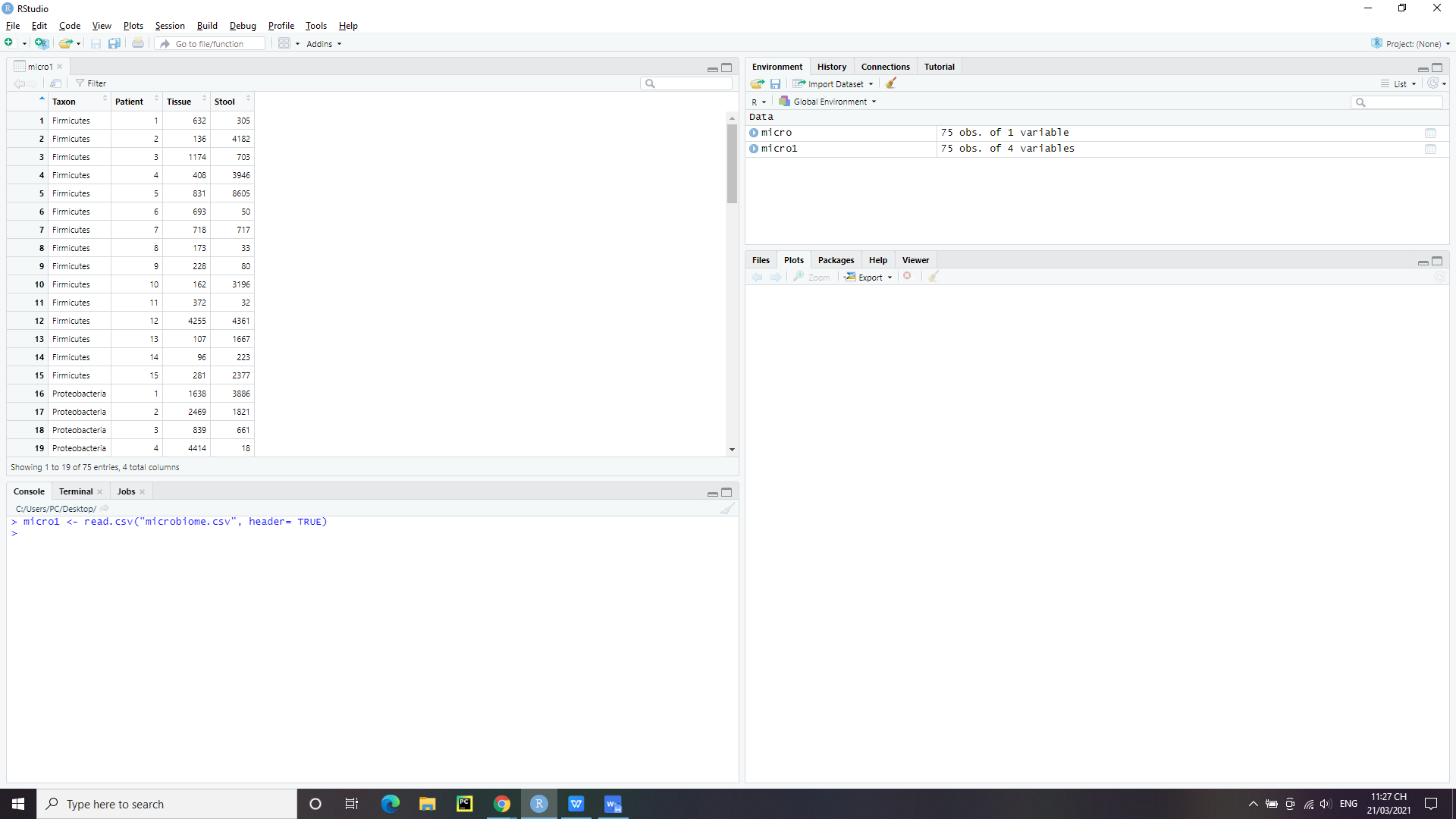
* R sẽ cho biết có các cột như sau trong dữ liệu (names là lệnh hỏi trong dữ liệu có những cột nào và tên gì):



1. **Nhập số liệu từ Excel: read.csv**

* Để nhập số liệu từ phần mềm Excel, chúng ta cần tiến hành 2 bước:
* Bước 1: Dùng lệnh “Save as” trong Excel và lưu số liệu dưới dạng “csv”
* Bước 2: Dùng R (lệnh read.csv) để nhập dữ liệu dạng csv.
* Việc đầu tiên là chúng ta cần làm, như nói trên, là vào Excel để lưu dưới dạng csv:
* B1: Vào Excel, chọn File Æ Save as
* B2: Chọn Save as type “CSV (Comma delimited)”
* Việc thứ hai là vào R và ra những lệnh sau đây:

> micro1 <- read.csv(“microbiome.csv”, header= TRUE)



1. **Nhập số liệu từ một SPSS: read.spss**

* Phần mềm thống kê SPSS lưu dữ liệu dưới dạng “sav”. Chẳng hạn như nếu chúng ta đã có một dữ liệu có tên là testo.sav trong directory c:\works\insulin, và muốn chuyển dữ liệu này sang dạng R có thể hiểu được, chúng ta cần sử dụng lệnh read.spss trong package có tên là foreign. Các lệnh sau đây sẽ hoàn tất dễ dàng việc này:
* Việc đầu tiên chúng ta cho truy nhập foreign bằng lệnh library:

> library(foreign)

* Việc thứ hai là lệnh read.spss:

> testo <- read.spss(“testo.sav”, to.data.frame=TRUE)

* Lệnh thứ hai read.spss yêu cầu R đọc số liệu từ “testo.sav”, và cho vào một data.frame có tên là testo.

1. **Thông tin về dữ liệu**

* Giả dụ như chúng ta đã nhập số liệu vào một data.frame có tên là chol như trong ví dụ 1. Để tìm hiểu xem trong dữ liệu này có gì, chúng ta có thể nhập vào R như sau:
* Dẫn cho R biết chúng ta muốn xử lí chol bằng cách dùng lệnh attach(arg) với arg là tên của dữ liệu.

> attach(micro)

* Chúng ta có thể kiểm tra xem chol có phải là một data.frame không bằng lệnh is.data.frame(arg) với arg là tên của dữ liệu. Ví dụ:

> is.data.frame(micro1)

1. TRUE

* R cho biết “micro1” quả là một data.frame.
* Có bao nhiêu cột (hay variable = biến số) và dòng số liệu (observations) trong dữ liệu này? Chúng ta dùng lệnh dim(arg) với arg là tên của dữ liệu. (dim viết tắt chữ dimension). Ví dụ (kết quả của R trình bày ngay sau khi chúng ta gõ lệnh):

> dim(micro1)

1. 75 4

* Như vậy, chúng ta có 50 dòng và 8 cột (hay biến số). Vậy những biến số này tên gì? Chúng ta dùng lệnh names(arg) với arg là tên của dữ liệu.

Ví dụ

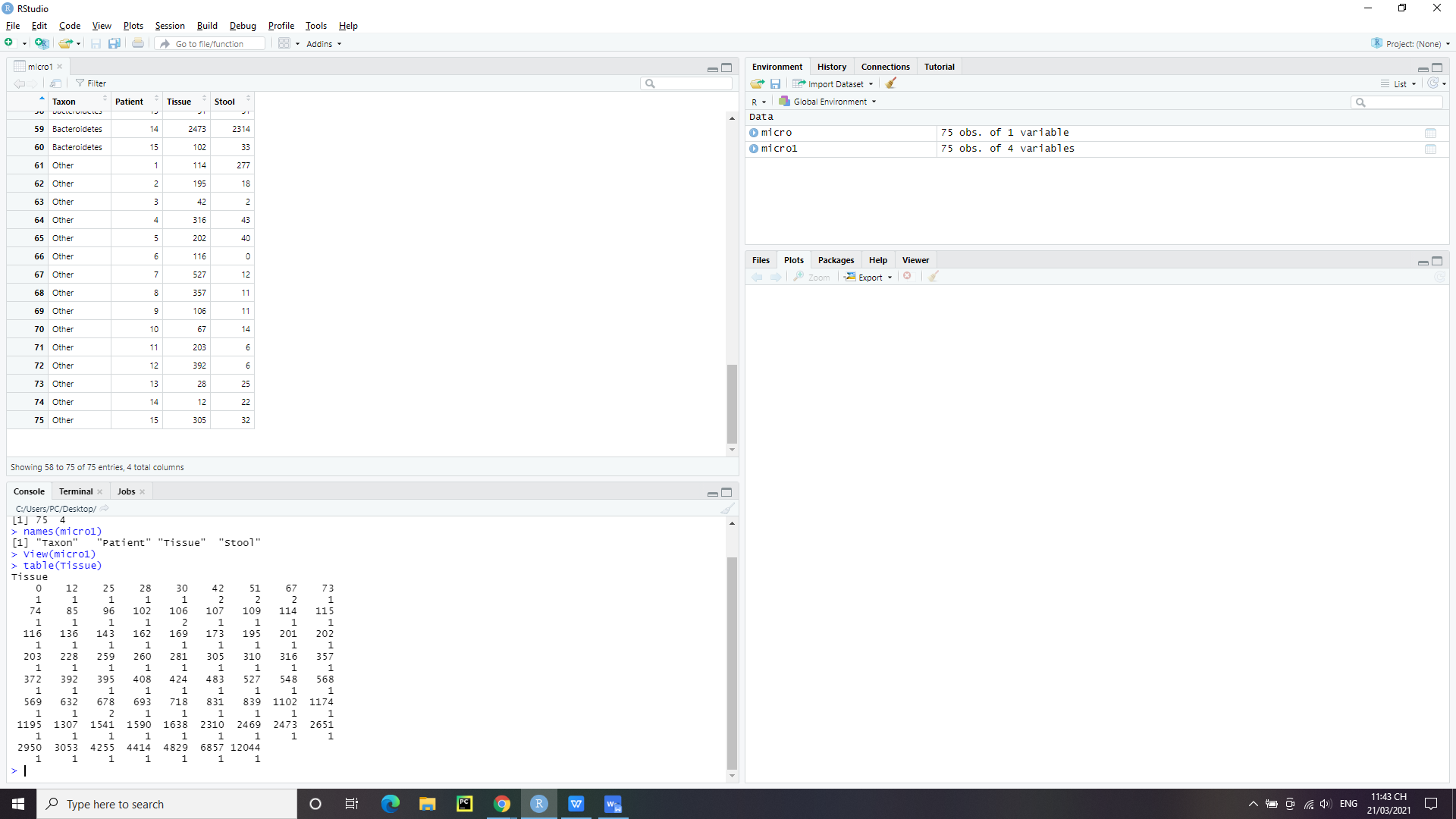
> names(micro1)

1. “Taxon” “Patient” “Tissue” “Stool”

* Trong biến số “Tissue”, chúng ta có bao nhiêu nam và nữ? Để trả lời câu hỏi này, chúng ta có thể dùng lệnh table(arg) với arg là tên của biến số.

Ví dụ:

> table(Tissue)



1. **Tạo dãy số bằng hàm seq, rep và gl**

R còn có công dụng tạo ra những dãy số rất tiện cho việc mô phỏng và thiết kế thí nghiệm. Những hàm thông thường cho dãy số là seq (sequence), rep (repetition) và gl (generating levels):

1. **Áp dụng rep**

* Công thức của hàm rep là rep(x, times, ...), trong đó, x là một biến số và times là số lần lặp lại.

Ví dụ:

• Tạo ra số 10, 3 lần:

> rep(10, 3)

1. 10 10 10

• Tạo ra số 1 đến 4, 3 lần:

> rep(c(1:4), 3)

1. 1 2 3 4 1 2 3 4 1 2 3 4

• Tạo ra số 1.2, 2.7, 4.8, 5 lần:

> rep(c(1.2, 2.7, 4.8), 5)

1. 1.2 2.7 4.8 1.2 2.7 4.8 1.2 2.7 4.8 1.2 2.7 4.8 1.2 2.7 4.8

• Tạo ra số 1.2, 2.7, 4.8, 5 lần:

> rep(c(1.2, 2.7, 4.8), 5)

1. 1.2 2.7 4.8 1.2 2.7 4.8 1.2 2.7 4.8 1.2 2.7 4.8 1.2 2.7 4.8
2. **Áp dụng seq**

• Tạo ra một vector số từ 1 đến 12:

> x <- (1:12)

> x

1. 1 2 3 4 5 6 7 8 9 10 11 12

> seq(12)

1. 1 2 3 4 5 6 7 8 9 10 11 12

• Tạo ra một vector số từ 12 đến 5:

> x <- (12:5)

> x

1. 12 11 10 9 8 7 6 5

> seq(12,7)

1. 12 11 10 9 8 7

Công thức chung của hàm seq là seq(from, to, by= ) hay seq(from, to,

length.out= ). Cách sử dụng sẽ được minh hoạ bằng vài ví dụ sau đây:

• Tạo ra một vector số từ 4 đến 6 với khoảng cách bằng 0.25:

> seq(4, 6, 0.25)

1. 4.00 4.25 4.50 4.75 5.00 5.25 5.50 5.75 6.00

• Tạo ra một vector 10 số, với số nhỏ nhất là 2 và số lớn nhất là 15

> seq(length=10, from=2, to=15)

[1] 2.000000 3.444444 4.888889 6.333333 7.777778 9.222222

10.666667 12.111111 13.555556 15.000000

1. **Áp dụng gl**

gl được áp dụng để tạo ra một biến thứ bậc (categorical variable), tức biến không để tính toán, mà là đếm. Công thức chung của hàm gl là gl(n, k, length = n\*k, labels = 1:n, ordered = FALSE) và cách sử dụng sẽ được minh hoạ bằng vài ví dụ sau đây:

• Tạo ra biến gồm bậc 1 và 2; mỗi bậc được lặp lại 8 lần:

> gl(2, 8)

1. 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2

Levels: 1 2

Hay một biến gồm bậc 1, 2 và 3; mỗi bậc được lặp lại 5 lần:

> gl(3, 5)

[1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3

Levels: 1 2 3

• Tạo ra biến gồm bậc 1 và 2; mỗi bậc được lặp lại 10 lần (do đó length=20):

> gl(2, 10, length=20)

1. 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2

Levels: 1 2

Hay:

> gl(2, 2, length=20)

1. 1 1 2 2 1 1 2 2 1 1 2 2 1 1 2 2 1 1 2 2

Levels: 1 2

• Cho thêm kí hiệu:

> gl(2, 5, label=c("C", "T"))

1. C C C C C T T T T T

Levels: C T

• Tạo một biến gồm 4 bậc 1, 2, 3, 4. Mỗi bậc lặp lại 2 lần.

> rep(1:4, c(2,2,2,2))

1. 1 1 2 2 3 3 4 4

Cũng tương đương với:

> rep(1:4, each = 2)

1. 1 1 2 2 3 3 4 4