R PROGRAMMING

ผู้ช่วยศาสตราจารย์ ดร. อัชฌาณัท รัตนเลิศนุสรณ์

สาขาสถิติประยุกต์ คณะวิทยาศาสตร์และเทคโนโลยี

มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี

สารบัญ

Contents

R data objects

R data objects พื้นฐานมีดังนี้

- **vectors**
- **lists**
- **matrices**
- **arrays**
- **factors**
- **data frames**

R data objects

vectors มีลักษณะการเก็บข้อมูลเป็นแถวหนึ่งแถว หรือเป็นคอลัมน์หนึ่งคอลัมน์ สามารถเก็บข้อมูลชนิดเดียวกันเท่านั้น

การสร้าง vectors สามารถใช้ฟังก์ชัน `c() --concatenate`

ตัวอย่างเช่น

```
> v1<-c(20,15,30.5,42.7)
> v1
[1] 20.0 15.0 30.5 42.7
> mode(v1)
[1] "numeric"
> v2<-c("20","15","30.5","42.7")
> v2
[1] "20"   "15"   "30.5" "42.7"
> mode(v2)
[1] "character"
```

R data objects

กรณีเก็บข้อมูลต่างชนิดกัน โปรแกรม R จะเปลี่ยนชนิดข้อมูลให้เป็นชนิดเดียวกัน

ตัวอย่างเช่น

```
> v3<-c(20,"Somchai",42L,4i)
> v3
[1] "20"      "Somchai" "42"      "0+4i"
```

ตัวแปร v3 (หรือเวกเตอร์ v3) เก็บข้อมูลชนิดตัวอักษรเท่านั้น

R data objects

การอ้างอิงสมาชิกของเวกเตอร์

```
###################
#     vector01.R  #
###################
x<-c()
x
y<-c(2,4,6,8,1,3,5,7)
y
y[2]
y[2:5]
y[-c(3,4,6)]
y[9:10]<-c(9,11)
y
y[y>5]
y[y>=4 & y<=9]
y[y<=5 | y>=8]
```

ผลลัพธ์ของคำสั่งใน vector01.R

```
> x<-c()
> x
NULL
> y<-c(2,4,6,8,1,3,5,7)
> y
[1] 2 4 6 8 1 3 5 7
> y[2]
[1] 4
> y[2:5]
[1] 4 6 8 1
> y[-c(3,4,6)]
[1] 2 4 1 5 7
> y[9:10]<-c(9,11)
> y
 [1]  2  4  6  8  1  3  5  7  9 11
> y[y>5]
[1]  6  8  7  9 11
> y[y>=4 & y<=9]
[1] 4 6 8 5 7 9
> y[y<=5 | y>=8]
[1]  2  4  8  1  3  5  9 11
```

R data objects

lists เป็นที่เก็บข้อมูลต่างชนิดกัน(เวกเตอร์ ฟังก์ชัน หรือ ลิสต์)ไว้ด้วยกัน

การสร้าง lists สามารถใช้ฟังก์ชัน `list()`

ตัวอย่างการสร้าง list

```
> list1 <- list(c(60,70,90),7.5,sqrt)
> list1
[[1]]
[1] 60 70 90

[[2]]
[1] 7.5

[[3]]
function (x)  .Primitive("sqrt")
```

สมาชิกตัวแรกของลิสต์เป็นเวกเตอร์ สมาชิกตัวที่สองเป็นตัวเลข และสมาชิกตัวที่สามเป็นฟังก์ชัน sqrt

R data objects

- การอ้างอิงสมาชิกใน list

```
######################
#    list01.R        #
######################
v<-c(4,7,10,9,15,12)
L1<-list(a=v,b=17.5,c=sqrt)
L1
L1$a
L1$b
L1$c
L1$a[1:4]
L1$a[L1$a>9]
```

- ผลลัพธ์ของ list01.R

```
> v<-c(4,7,10,9,15,12)
> L1<-list(a=v,b=17.5,c=sqrt)
> L1
$a
[1]  4  7 10  9 15 12

$b
[1] 17.5

$c
function (x)  .Primitive("sqrt")

> L1$a
[1]  4  7 10  9 15 12
> L1$b
[1] 17.5
> L1$c
function (x)  .Primitive("sqrt")
> L1$a[1:4]
[1]  4  7 10  9
> L1$a[L1$a>9]
[1] 10 15 12
```

R data objects

matrices เป็นที่เก็บข้อมูลแบบอะเรย์ 2 มิติ ขนาด nxm

ตัวอย่างการสร้างเมทริกซ์ด้วยฟังก์ชัน `matrix()`

```
> Matx<- matrix( c(1,3,5,0,4,6), nrow = 2, ncol = 3)
> Matx
     [,1] [,2] [,3]
[1,]    1    5    4
[2,]    3    0    6
```

ตัวแปร Matx เป็นเมทริกซ์ขนาด 2x3

R data objects

- การอ้างอิงสมาชิกใน matrix

```
######################
#    matrix01.R      #
######################
v<-c(1,7,9,2,8,10)
X<-matrix(v,nrow=2,ncol=3)
X
X[1,]
X[1:2,]
X[,1]
X[,1:3]
X[,-2]
X[X>5]
```

- ผลลัพธ์ของ matrix01.R

```
> v<-c(1,7,9,2,8,10)
> X<-matrix(v,nrow=2,ncol=3)
> X
     [,1] [,2] [,3]
[1,]    1    9    8
[2,]    7    2   10
> X[1,]
[1] 1 9 8
> X[1:2,]
     [,1] [,2] [,3]
[1,]    1    9    8
[2,]    7    2   10
> X[,1]
[1] 1 7
> X[,1:3]
     [,1] [,2] [,3]
[1,]    1    9    8
[2,]    7    2   10
> X[,-2]
     [,1] [,2]
[1,]    1    8
[2,]    7   10
> X[X>5]
[1]  7  9  8 10
```

R data objects

arrays เป็นที่เก็บข้อมูลแบบอะเรย์ 1 มิติ , 2 มิติ และ 3 มิติ

ตัวอย่างการสร้างอะเรย์ด้วยฟังก์ชัน `array()`

```
> Array.x <- array(c(1,3,5,7,2,4,6,8),dim = c(4,2))
> Array.x
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
```

R data objects

- การอ้างอิงสมาชิกใน array

```
######################
#    array01.R       #
######################
v<-c(1,4,7,10,2,5,8,11)
A<-array(v,dim=c(4,2))
A
A[1,]
A[2:3,]
A[,1]
A[,1:2]
A[-3,]
A[A<=2 | A>=7]
```

- ผลลัพธ์ของ array01.R

```
> v<-c(1,4,7,10,2,5,8,11)
> A<-array(v,dim=c(4,2))
> A
     [,1] [,2]
[1,]    1    2
[2,]    4    5
[3,]    7    8
[4,]   10   11
> A[1,]
[1] 1 2
> A[2:3,]
     [,1] [,2]
[1,]    4    5
[2,]    7    8
> A[,1]
[1]  1  4  7 10
> A[,1:2]
     [,1] [,2]
[1,]    1    2
[2,]    4    5
[3,]    7    8
[4,]   10   11
> A[-3,]
     [,1] [,2]
[1,]    1    2
[2,]    4    5
[3,]   10   11
> A[A<=2 | A>=7]
[1]  1  7 10  2  8 11
```

R data objects

factors เป็นที่เก็บข้อมูลจำแนกประเภท อาทิ เพศ อาชีพ ระดับการศึกษา เป็นต้น

ตัวอย่างการสร้าง Factors ด้วยฟังก์ชัน `factor()`

```
> gender <- factor(c("Male","Female","Female","Male","Female"))
> gender
[1] Male   Female Female Male   Female
Levels: Female Male
```

ตัวแปร gender เป็นตัวแปรชนิด factors เก็บข้อมูลจำแนกกลุ่ม มี 2 กลุ่ม

คือ เพศหญิง และเพศชาย

R data objects

- การอ้างอิงสมาชิกใน factor

```
######################
#    factor01.R      #
######################
gender<-c("male","female","male")
F<-factor(gender)
F
F[1]
F[1:2]
F[-2]
```

- ผลลัพธ์ของ factor01.R

```
> gender<-
c("male","female","male")
> F<-factor(gender)
> F
[1] male   female male
Levels: female male
> F[1]
[1] male
Levels: female male
> F[1:2]
[1] male   female
Levels: female male
> F[-2]
[1] male male
Levels: female male
```

R data objects

data frames เป็นที่เก็บข้อมูลแบบตาราง(Table) หรือแฟ้มข้อมูล (data file) สามารถตั้งชื่อคอลัมน์หรือฟิลด์(fields)ได้ มีเงื่อนไขว่าจำนวนแถวของทุกคอลัมน์ต้องเท่ากัน

ตัวอย่างการสร้าง data frames ด้วยฟังก์ชัน `data.frame()`

```
> name <-c("Wattana","Somchai","Arunee")
> gender <- c("Male", "Male","Female")
> age <- c(20,19,25)
> student.dat<-data.frame(name,gender,age)
> student.dat
     name gender age
1 Wattana   Male  20
2 Somchai   Male  19
3  Arunee Female  25
```

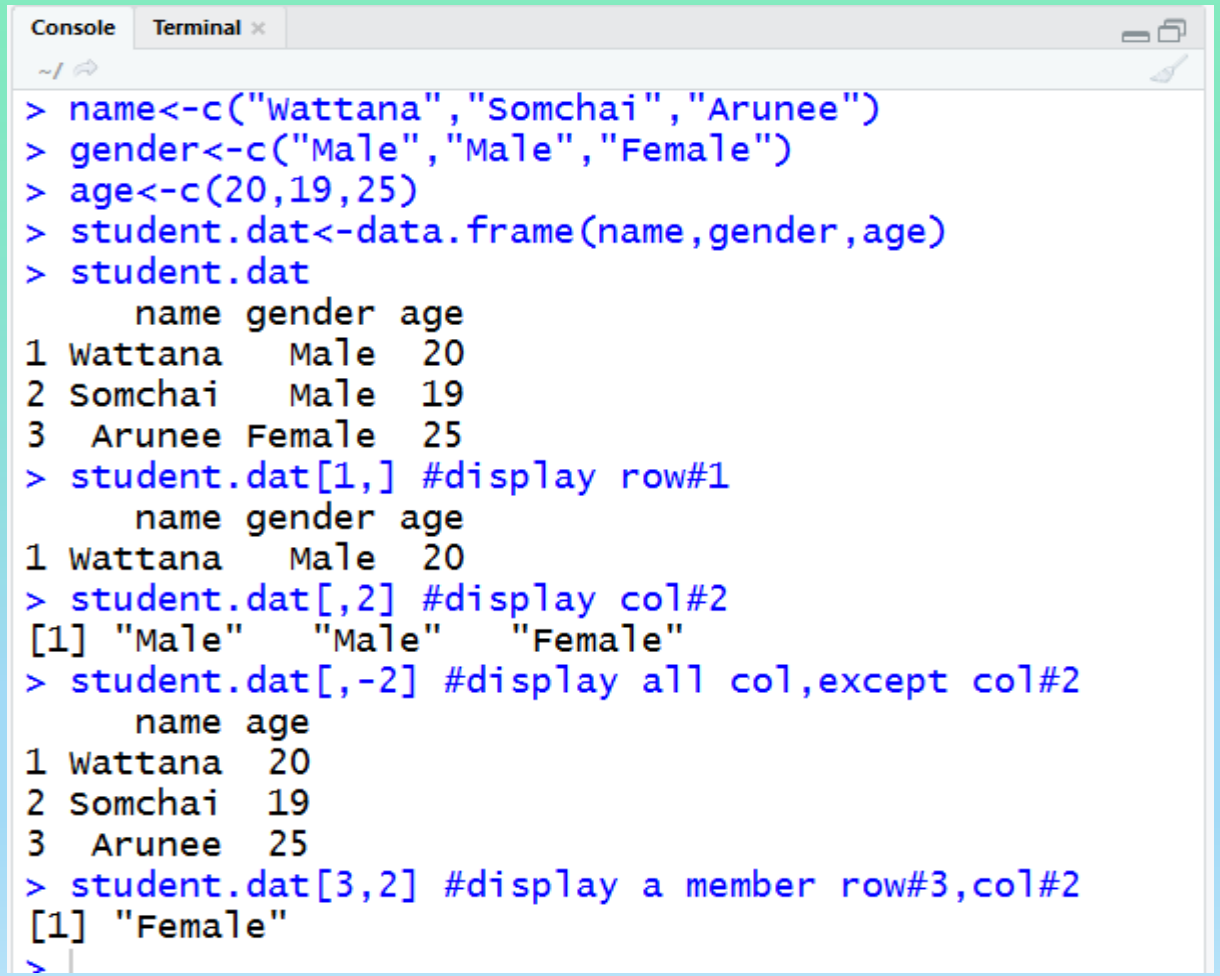
R data objects

การอ้างอิงสมาชิกใน data frames

```
1  ####################
2  #  datafr.r
3  ####################
4  name<-c("Wattana","Somchai","Arunee")
5  gender<-c("Male","Male","Female")
6  age<-c(20,19,25)
7  student.dat<-data.frame(name,gender,age)
8  student.dat
9  student.dat[1,] #display row#1
10 student.dat[,2] #display col#2
11 student.dat[,-2] #display all col,except col#2
12 student.dat[3,2] #display a member row#3,col#2
```

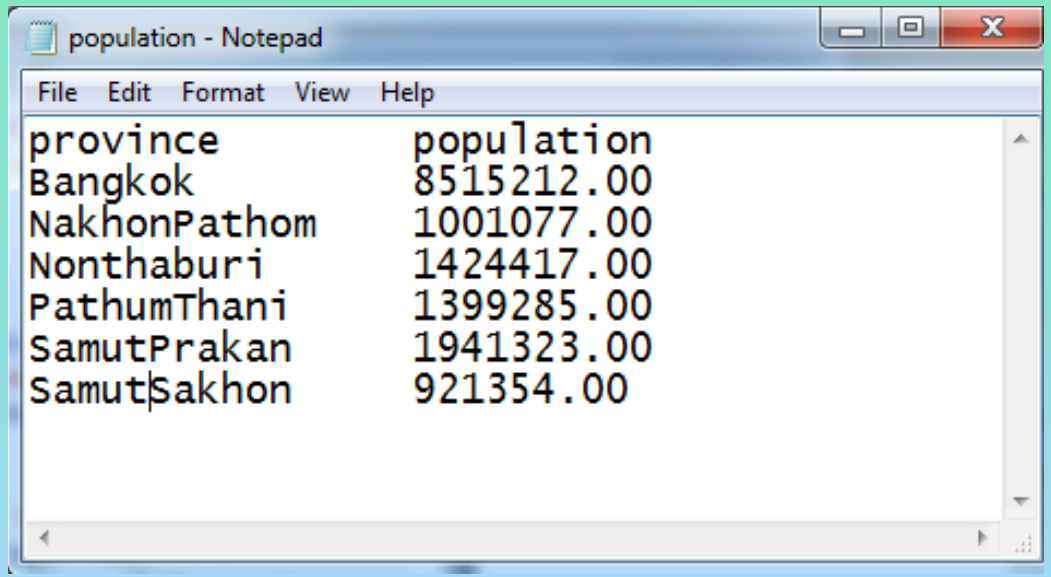
R data objects

ผลลัพธ์ที่ได้



การอ่านข้อมูลจากแฟ้มข้อมูลชนิด text

ก่อนที่จะอ่านข้อมูลให้เตรียมไฟล์ชนิด txt (Text documents) จากโปรแกรม Notepad ดังรูป



ข้อควรระวัง การพิมพ์ข้อมูลชื่อจังหวัดต้องไม่เว้นวรรค และการพิมพ์ตัวเลขประชากรต้องไม่ใส่เครื่องหมาย , (comma)

การอ่านข้อมูลจากแฟ้มข้อมูลชนิด text

ใช้ฟังก์ชัน `read.table()` อ่านข้อมูลจากแฟ้มชนิด txt เข้ามาในโปรแกรม R ดังนี้

```
> read.table("population.txt",header = TRUE)
      province population
1       Bangkok    8515212
2  NakhonPathom    1001077
3    Nonthaburi    1424417
4   PathumThani    1399285
5   SamutPrakan    1941323
6   SamutSakhon     921354
>
```

ข้อควรระวัง ในฟังก์ชัน `read.table()` เมื่อระบุ `header=TRUE` แสดงว่าให้ใช้ข้อมูลแถวแรกในไฟล์ `population.txt` มาเป็นชื่อคอลัมน์ โดยปกติฟังก์ชันนี้จะกำหนดให้ `header=FALSE`

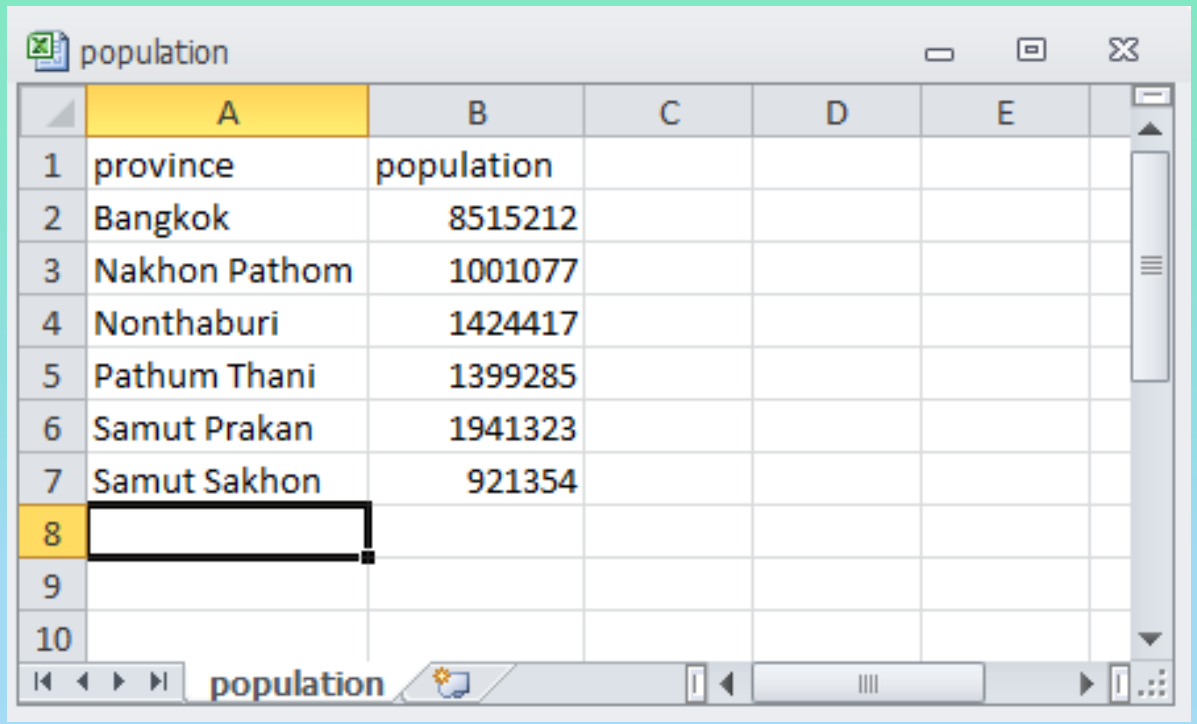
การอ่านข้อมูลจากแฟ้มข้อมูลชนิด text

การอ่านข้อมูลเข้ามาในโปรแกรม Rแล้วควรเก็บไว้ในตัวแปรชนิด data frames เพื่อความสะดวกในการเรียกใช้งานภายหลังดังนี้

```
> popu.dat<-data.frame(read.table("population.txt",
header = TRUE))
> popu.dat
      province population
1       Bangkok    8515212
2 NakhonPathom    1001077
3    Nonthaburi    1424417
4  PathumThani    1399285
5  SamutPrakan    1941323
6  SamutSakhon     921354
>
```

การอ่านข้อมูลจากแฟ้มข้อมูลชนิด csv

ก่อนที่จะอ่านข้อมูลให้เตรียมไฟล์ชนิด csv (Comma delimited)จากโปรแกรม Microsoft Excel ดังนี้



การอ่านข้อมูลจากแฟ้มข้อมูลชนิด csv

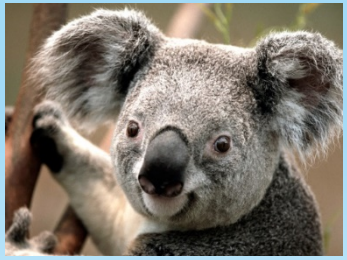
ใช้ฟังก์ชัน read.csv() อ่านข้อมูลจากแฟ้มชนิด csv เข้ามาในโปรแกรม R ดังนี้

```
> read.csv("population.csv", header = TRUE)
       province population
1        Bangkok    8515212
2 Nakhon Pathom    1001077
3     Nonthaburi    1424417
4   Pathum Thani    1399285
5   Samut Prakan    1941323
6   Samut Sakhon     921354
>
```

การอ่านข้อมูลจากแฟ้มข้อมูลชนิด csv

การอ่านข้อมูลเข้ามาในโปรแกรม Rแล้วควรเก็บไว้ในตัวแปรชนิด data frames เพื่อความสะดวกในการเรียกใช้งานภายหลังดังนี้

```
> popucsv.dat<-data.frame(read.csv("population.csv",
header = TRUE))
> popucsv.dat
       province population
1       Bangkok    8515212
2 Nakhon Pathom    1001077
3    Nonthaburi    1424417
4  Pathum Thani    1399285
5  Samut Prakan    1941323
6  Samut Sakhon     921354
>
```



การบันทึกข้อมูลลงแฟ้มข้อมูลชนิด text

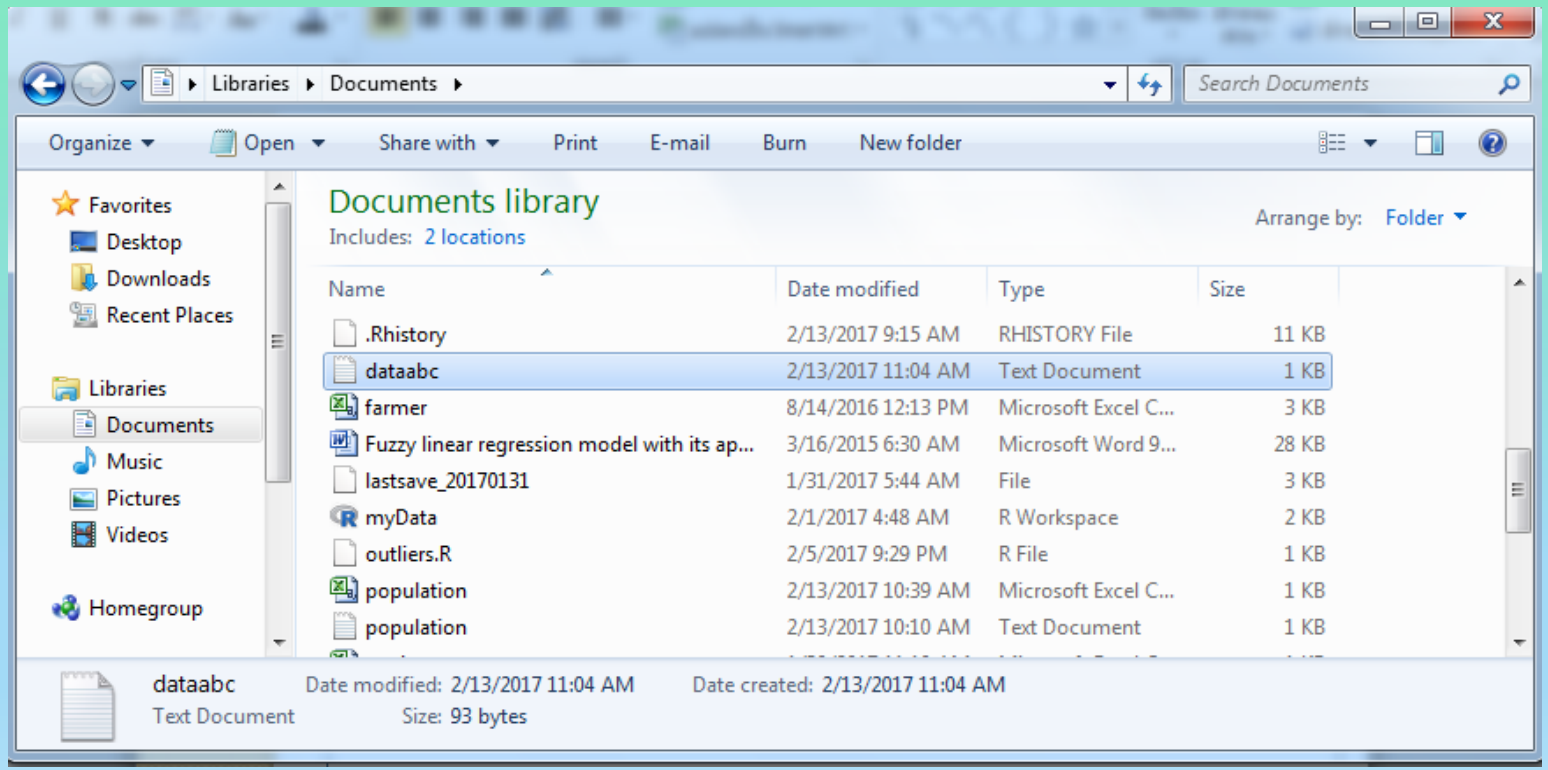
โปรแกรม R สามารถใช้ตัวแปร data frames เพื่อบันทึกข้อมูลลงแฟ้ม csv และ txt ได้ดังนี้

ตัวอย่างการบันทึกข้อมูลลงแฟ้ม txt ดังนี้

```
> number<-1:5
> number
[1] 1 2 3 4 5
> lowerletter<-letters[1:5]
> lowerletter
[1] "a" "b" "c" "d" "e"
> upperletter<-LETTERS[1:5]
> upperletter
[1] "A" "B" "C" "D" "E"
> data1<-data.frame(number,lowerletter,upperletter)
> data1
  number lowerletter upperletter
1      1           a           A
2      2           b           B
3      3           c           C
4      4           d           D
5      5           e           E
> write.table(data1,"dataabc.txt", row.names = FALSE)
>
```

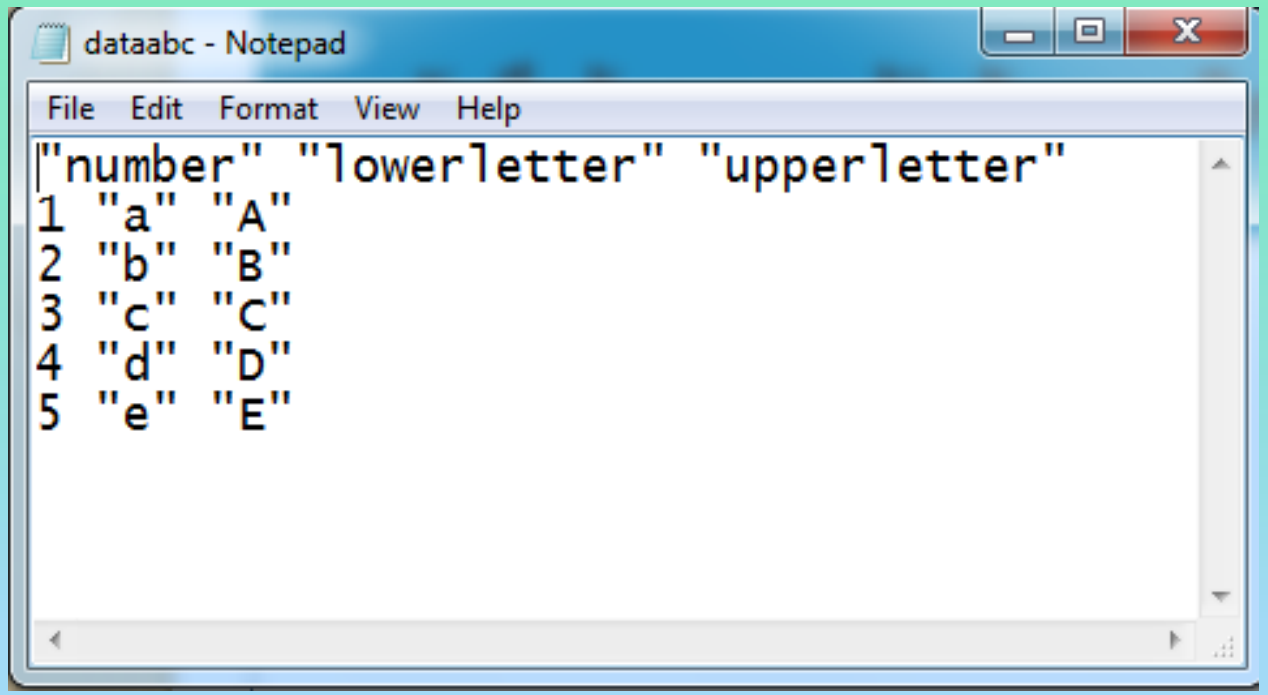
การบันทึกข้อมูลลงแฟ้มข้อมูลชนิด text

ผลลัพธ์ที่ได้คือแฟ้ม `dataabc.txt` ซึ่งเก็บอยู่ที่ working directories ของโปรแกรม R ดังนี้



การบันทึกข้อมูลลงแฟ้มข้อมูลชนิด text

เมื่อเปิดไฟล์ `dataabc.txt` จะเห็นข้อมูลดังรูป



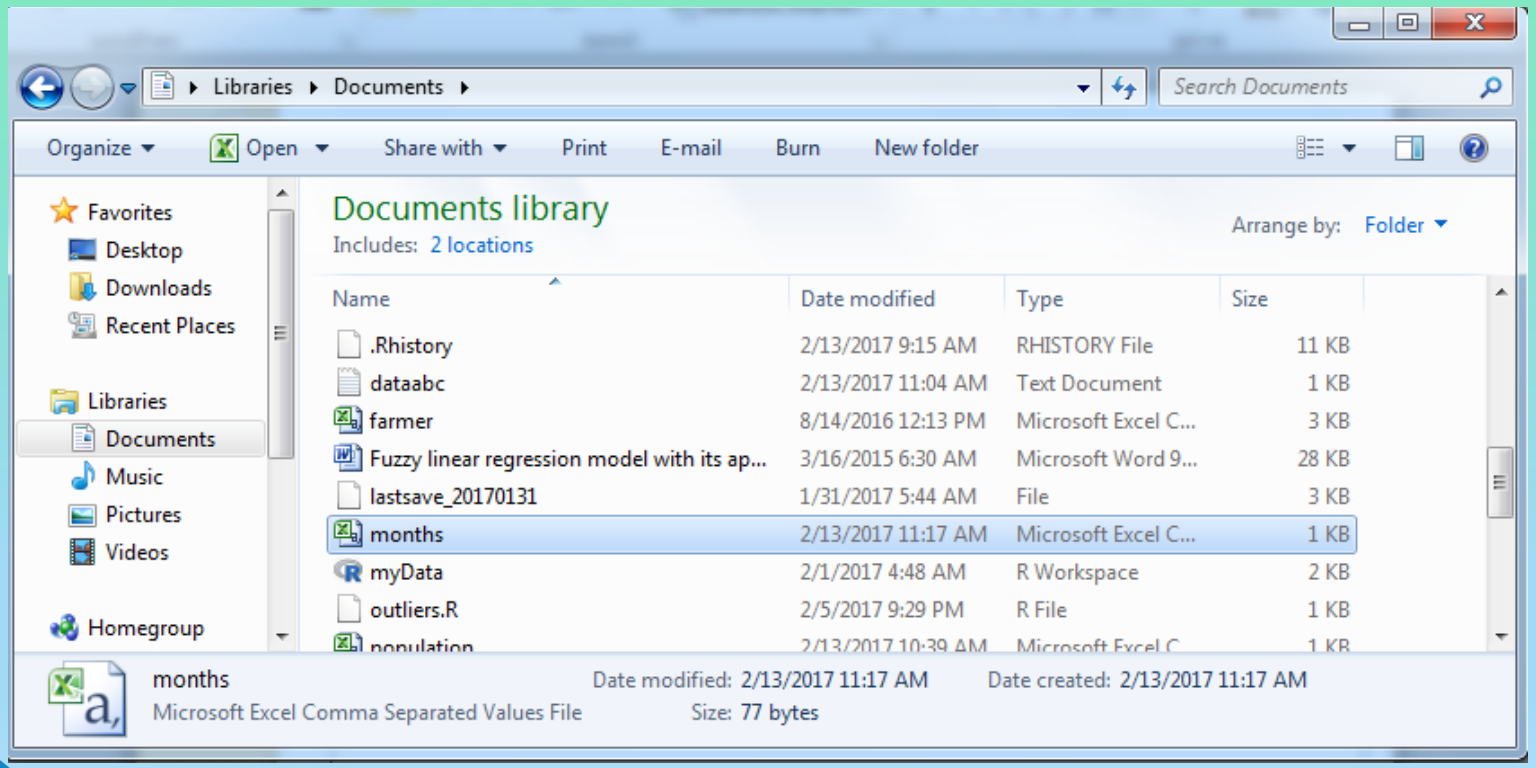
การบันทึกข้อมูลลงแฟ้มข้อมูลชนิด csv

ตัวอย่างการบันทึกข้อมูลลงแฟ้ม csv ดังนี้

```
> number2<-1:5
> number2
[1] 1 2 3 4 5
> Month<-month.name[1:5]
> Month
[1] "January"  "February" "March"    "April"    "May"
> data2<-data.frame(number2,Month)
> data2
  number2    Month
1       1  January
2       2 February
3       3    March
4       4    April
5       5      May
> write.csv(data2,"months.csv", row.names = FALSE)
>
```

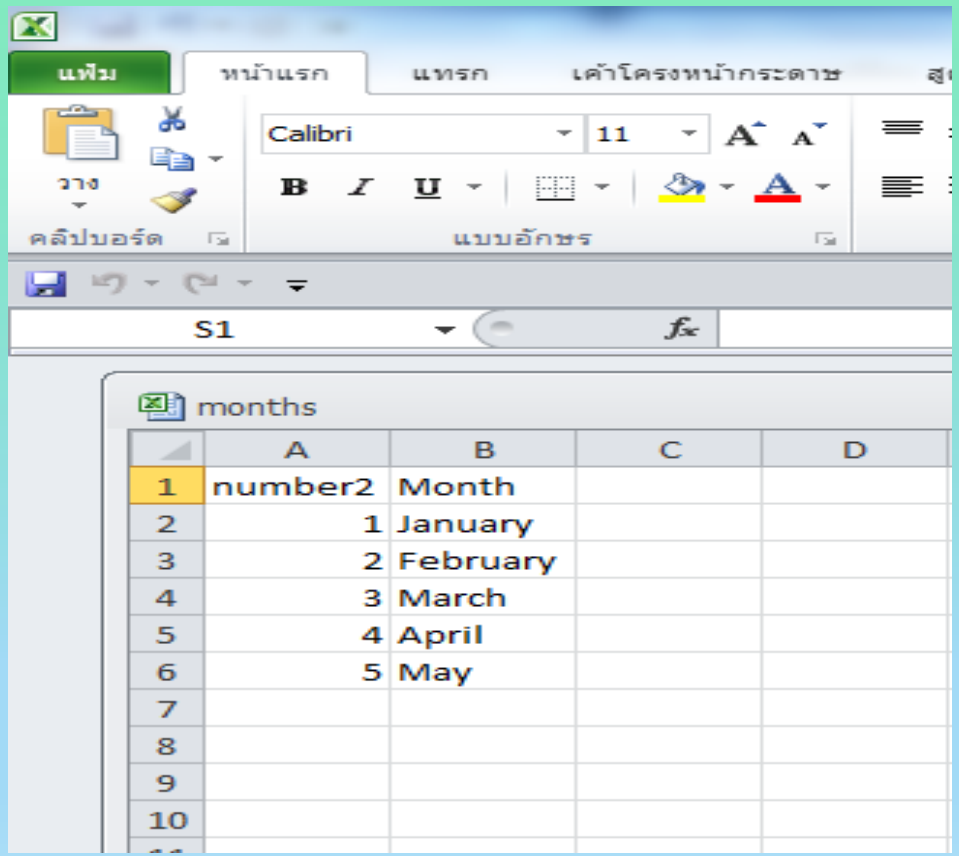
การบันทึกข้อมูลลงแฟ้มข้อมูลชนิด csv

ผลลัพธ์ที่ได้คือแฟ้ม `months.csv` ซึ่งเก็บอยู่ที่ working directories ของโปรแกรม R ดังนี้



การบันทึกข้อมูลลงแฟ้มข้อมูลชนิด csv

- เมื่อเปิดไฟล์ `months.csv` จะเห็นข้อมูลดังรูป



	A	B	C	D
1	number2	Month		
2	1	January		
3	2	February		
4	3	March		
5	4	April		
6	5	May		
7				
8				
9				
10				

การคำนวณค่าสถิติเบื้องต้น

ฟังก์ชันทางสถิติที่ใช้คำนวณค่าสถิติเบื้องต้นของข้อมูลมีดังนี้

ฟังก์ชัน	คำอธิบาย
mean()	หาค่าเฉลี่ยเลขคณิตของข้อมูลในเวกเตอร์
median()	หามัธยฐานของข้อมูลในเวกเตอร์
sum()	หาค่าผลรวมของข้อมูลในเวกเตอร์
length()	หาจำนวนข้อมูลในเวกเตอร์
min()	หาข้อมูลในเวกเตอร์ที่มีค่าน้อยที่สุด
max()	หาข้อมูลในเวกเตอร์ที่มีค่ามากที่สุด

การคำนวณค่าสถิติเบื้องต้น

ตัวอย่างการใช้ฟังก์ชันทางสถิติ

```
> x<-c(1,3,4,7,6,9)
> x
[1] 1 3 4 7 6 9
> mean.x<-mean(x)
> mean.x
[1] 5
> median.x<-median(x)
> median.x
[1] 5
> sum.x<-sum(x)
> sum.x
[1] 30
> n<-length(x)
> n
[1] 6
```

การคำนวณค่าสถิติเบื้องต้น

ถ้าต้องการหาค่าฐานนิยมของข้อมูล สามารถหาได้ดังนี้

```
> y<-c(3,4,4,4,5,6,8)
> y
[1] 3 4 4 4 5 6 8
> which.max(table(y))
4
2
```

หมายความว่า mode คือ 4 ส่วนตัวเลข 2 เป็นลำดับที่ของ mode หลังจากเรียงข้อมูลจากน้อยไปมากด้วยฟังก์ชัน `table()`

ถ้าใช้ฟังก์ชัน `mode()` จะเป็นการตรวจสอบชนิดของข้อมูล

```
> y<-c(3,4,4,4,5,6,8)
> mode(y)
[1] "numeric"
```

การวัดการกระจายข้อมูล

ฟังก์ชันทางสถิติที่ใช้วัดการกระจายของข้อมูลมีดังนี้

ฟังก์ชัน	คำอธิบาย
range()	หาค่าสูงสุดและต่ำสุดของข้อมูลในเวกเตอร์
diff()	หาผลต่างของข้อมูล
sd()	หาค่าส่วนเบี่ยงเบนมาตรฐานของข้อมูลตัวอย่าง
var()	หาค่าความแปรปรวนของข้อมูลตัวอย่าง

การวัดการกระจายข้อมูล

ตัวอย่างการใช้ฟังก์ชันวัดการกระจายของข้อมูล

```
> z<-1:10
> z
 [1]  1  2  3  4  5  6  7  8  9 10
> range(z)                    #หาค่าต่ำสุดและค่าสูงสุดของข้อมูล
[1]  1 10
> diff(range(z))              #หาค่าพิสัยของข้อมูล
[1] 9
> sd(z)                       #หาค่าส่วนเบี่ยงเบนมาตรฐานของข้อมูลตัวอย่าง
[1] 3.02765
> var(z)                      #หาความแปรปรวนของข้อมูลตัวอย่าง
[1] 9.166667
```

การจัดการค่าสูญหาย (missing values)

โปรแกรม R ใช้ฟังก์ชัน na.omit( ) เพื่อลบค่าสูญหายออกไปจากเวกเตอร์หรือ data frames

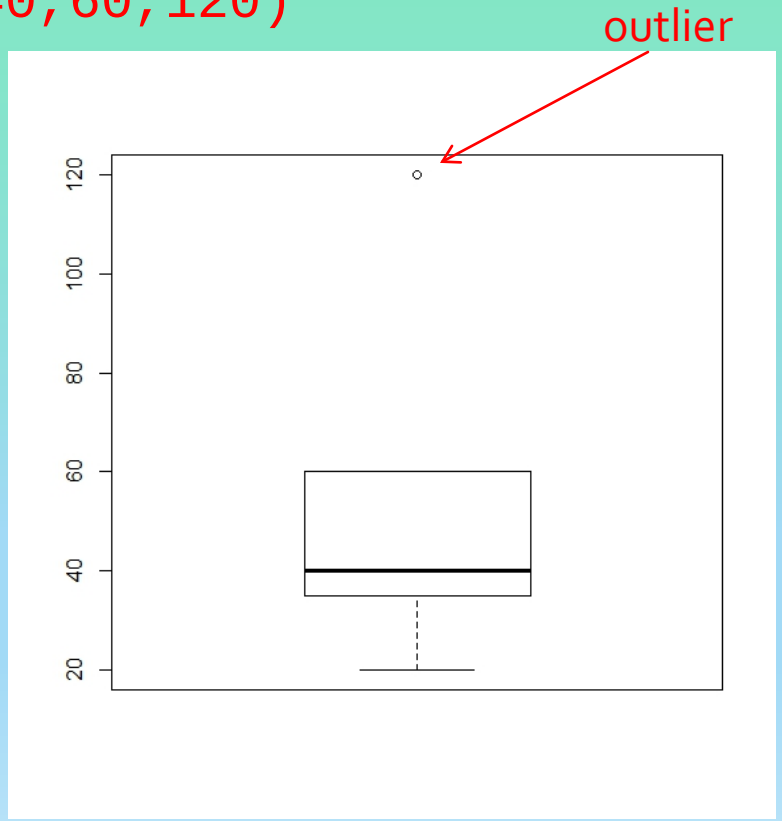
```
> x<-c(1,5,7,4,NA)
> x
[1]  1  5  7  4 NA
> x.new<-na.omit(x)
> x.new
[1] 1 5 7 4
attr(,"na.action")
[1] 5
attr(,"class")
[1] "omit"
>
```

การตรวจสอบค่าผิดปกติ (outliers)

ค่าผิดปกติในเวกเตอร์ สามารถตรวจสอบได้ดังนี้

ใช้แผนภาพ boxplot ตรวจดูค่าที่กระจายออกจากกลุ่ม

```
> x<-c(20,35,40,60,120)
> boxplot(x)
>
```



การตรวจสอบค่าผิดปกติ (outliers)

นอกจากนี้ยังสามารถใช้ R package ชื่อ outliers หาค่าผิดปกติได้

ตัวอย่างการตรวจสอบค่าผิดปกติ

```
> x<-c(20,35,40,60,120)
> x
[1]  20  35  40  60 120
> library("outliers") #load R package outliers
> outlier(x)              # เรียกใช้ฟังก์ชัน outlier
[1] 120
>
```

ค่าผิดปกติ คือข้อมูลที่มีค่าเท่ากับ 120

หมายเหตุ: ฟังก์ชัน `outlier()` ใช้หาระยะห่างที่มากที่สุดระหว่างข้อมูลค่าที่ต้องการตรวจสอบกับค่าเฉลี่ยเลขคณิตที่ตัดข้อมูลค่านั้นออก

การตรวจสอบค่าผิดปกติ (outliers)

หรือเราเขียนคำสั่งขึ้นเองเพื่อตรวจสอบค่าผิดปกติ

ตัวอย่างการตรวจสอบค่าผิดปกติ

```
> x<-c(20,35,40,60,120)
> x
[1]  20  35  40  60 120
> meanx.1<-mean(c(35,40,60,120))
> meanx.2<-mean(c(20,40,60,120))
> meanx.3<-mean(c(20,35,60,120))
> meanx.4<-mean(c(20,35,40,120))
> meanx.5<-mean(c(20,35,40,60))
> meanx<-
c(meanx.1,meanx.2,meanx.3,meanx.4,meanx.5)
> distance<-x-meanx
> cbind(x,meanx,distance)
        x meanx distance
[1,]  20 63.75   -43.75
[2,]  35 60.00   -25.00
[3,]  40 58.75   -18.75
[4,]  60 53.75     6.25
[5,] 120 38.75    81.25
```

ระยะห่างมากที่สุด

ค่าผิดปกติ