Elección de subconjuntos de datos

Herramientas de programación para el análisis de datos

Clase 8

Seleccionar parte de la base de datos / tabla / Data Frame

Tome una tabla

col1	col2	col3	col4	col5
1	234	Hola	NA (Si es R)	**True** (Si es Python)
2	124	Qué tal	Colombiano	**True**
1	524	Salut!	Francés	**False**

Qué podemos hacer

col1	col2	col3	col4	col5
1	234	Hola	NA (Si es R)	**True** (Si es Python)
2	124	Qué tal	Colombiano	**True**
1	524	Salut!	Francés	**False**

- Seleccionar columnas (en STATA les decimos variables)
  - Individuales y grupales
- Seleccionar filas (en STATA les decimos observaciones)
  - Individuales y grupales
- Filtrar observaciones: hacer operaciones pero con un **subgrupo** de observaciones
- Repetir operaciones para varios subgrupos
- Crear columnas y reemplazar valores de columnas
- Eliminar filas y columnas

STATA

Base de datos

Seleccionar columnas: varlist

- Normalmente sólo hay que referirse al nombre de la “variable” correspondiente.

  Como en **`sum var3`**

- Pero a veces se pueden usar atajos, por ejemplo:

  `var*`

Significa `var1 var2 var3`

	var1	var2	var3
1	1	12	374
2	1	13	725
3	2	35	355
4	2	32	332
5	3	73	836
6	3	27	247
7	4	82	247
8	5	54	247
9	6	25	353
10	6	53	235
11	7	73	235

Recordemos la sintaxis básica de STATA

comando [*varlist*]**, opciones**

comando [*varlist*] = [nombre]**, opciones**

Etc…

Por ejemplo:

gen nuevaVariable = viejaVariable * 2

sum var2

sum var2**, detail**

Stata

En STATA: hay una diferencia entre expresión y comando

Como comando en el dofile:

```
if $unGlobal == 1 {
    display "hola"
}
```

Como expresión en un comando:

```
sum var2 if(var1 == 1)
```

Stata

Filtrar

```
. sum var2 if(var1 == 1)
```

Variable	Obs	Mean	Std. Dev.	Min	Max
var2	2	12.5	.7071068	12	13

Syntax

```
summarize [varlist] [if] [in] [weight] [, options]
```

options	Description
Main	
detail	display additional statistics
meanonly	suppress the display; calculate only the mean; programmer's option
format	use variable's display format
separator(#)	draw separator line after every # variables; default is **separator(5)**
display_options	control spacing, line width, and base and empty cells

	var1	var2	var3
1	1	12	374
2	1	13	725
3	2	35	355
4	2	32	332
5	3	73	836
6	3	27	247
7	4	82	247
8	5	54	247
9	6	25	353
10	6	53	235
11	7	73	235

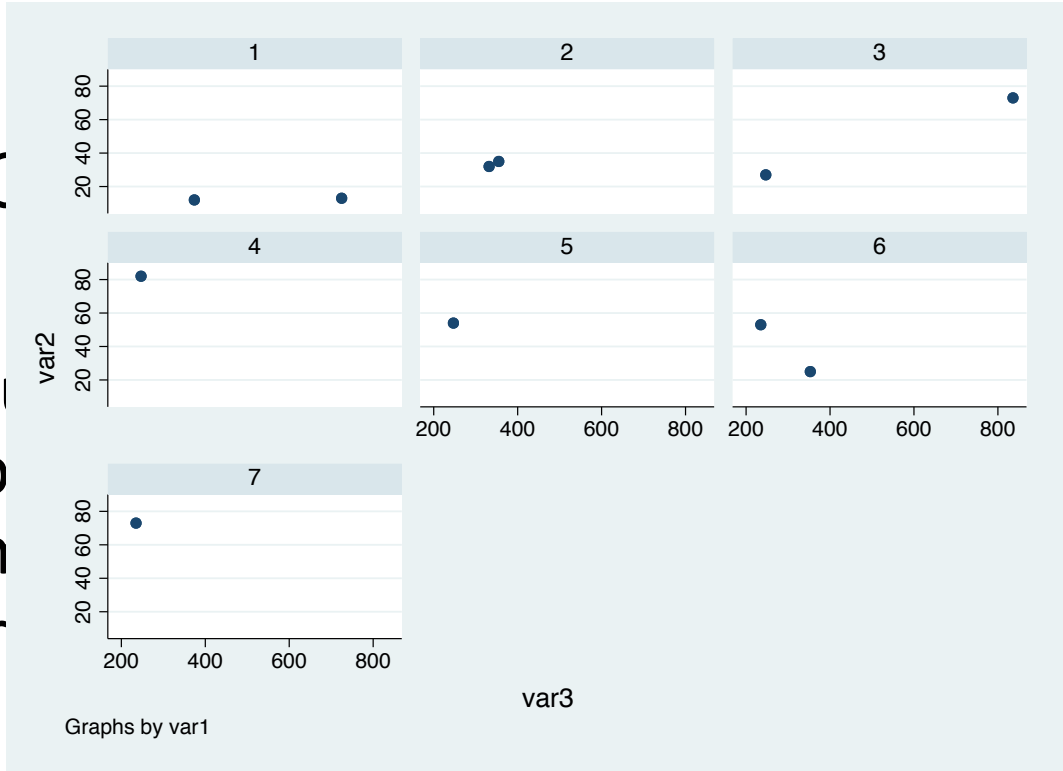
Podemos consultar la documentación de cada comando para filtrar con la expresión `if`

```
sum var2 if(var1 = 1)
```

Stata

Rep varios subgrupos

- Alg
  exp
  con inidos
  por



Por ejemplo:

```
twoway scatter var2 var3, by(var1)
```

	var1	var2	var3
1	1	12	374
2	1	13	725
3	2	35	355
4	2	32	332
5	3	73	836
6	3	27	247
7	4	82	247
8	5	54	247
9	6	25	353
10	6	53	235
11	7	73	235

Stata

Crear columnas nuevas

- Para crear columnas nuevas tenemos dos comandos:

gen

egen

Gen permite crear columnas con fórmulas generales como:

```
gen var4 = var2 + 1
```

	var1	var2	var3	var4
1	1	12	374	13
2	1	13	725	14
3	2	35	355	36
4	2	32	332	33
5	3	73	836	74
6	3	27	247	28
7	4	82	247	83
8	5	54	247	55
9	6	25	353	26
10	6	53	235	54
11	7	73	235	74

Crear columnas nuevas

- Para crear columnas nuevas tenemos dos comandos:

gen

egen

Egen permite usar funciones matemáticas:

```
egen var4 = min(var3)
```

	var1	var2	var3	var4
1	1	12	374	235
2	1	13	725	235
3	2	35	355	235
4	2	32	332	235
5	3	73	836	235
6	3	27	247	235
7	4	82	247	235
8	5	54	247	235
9	6	25	353	235
10	6	53	235	235
11	7	73	235	235

Crear columnas nuevas

- Para crear columnas nuevas tenemos dos comandos:

gen

egen

Además, egen permite aplicar asignaciones por subgrupos

```
egen var4 = min(var3), by(var1)
```

	var1	var2	var3	var4
1	1	12	374	374
2	1	13	725	374
3	2	35	355	332
4	2	32	332	332
5	3	73	836	247
6	3	27	247	247
7	4	82	247	247
8	5	54	247	247
9	6	25	353	235
10	6	53	235	235
11	7	73	235	235

Stata

Reemplazar valores de una columna

- Es igual que gen pero con:

```
replace
```

Por ejemplo,

```
replace var3 = var2
```

Note: no hay un “ereplace”

	var1	var2	var3
1	1	12	12
2	1	13	13
3	2	35	35
4	2	32	32
5	3	73	73
6	3	27	27
7	4	82	82
8	5	54	54
9	6	25	25
10	6	53	53
11	7	73	73

Stata

Eliminar filas y columnas

Para eliminar columnas usamos el comando

```
drop [varlist]
```

```
drop var2
```

También podemos usar keep para quedarnos con un conjunto de variables

```
keep var1 var3
```

		var1	var3	
1	1	1	374	74
2	2	1	725	25
3	3	2	355	55
4	4	2	332	32
5	5	3	836	36
6	6	3	247	47
7	7	4	247	47
8	8	5	247	47
9	9	6	353	53
10	10	6	235	35
11	11	7	235	35

Stata

Eliminar filas y columnas

Para eliminar filas usamos drop, pero con la opción `if`

```
drop if exp
```

```
drop if var3 < 300
```

	var1	var2	var3
1	1	12	374
2	1	13	725

	var1	var2	var3
1	1	12	374
2	1	13	725
3	2	35	355
4	2	32	332
5	3	73	836
6	6	25	353
10	6	53	235
11	7	73	235

Ejercicio en STATA

Python

Pandas DataFrame

Ahora tenemos un DataFrame de Pandas

Llamado “datos”

[5]:

	var1	var2	var3
0	1	12	374
1	1	13	725
2	2	35	355
3	2	32	332
4	3	73	836
5	3	27	247
6	4	82	247
7	5	54	247
8	6	25	353
9	6	53	235
10	7	73	235

Python

Python

Seleccionar columnas

Para seleccionar columnas podemos usar el nombre de la columna:

```
nuevosdatos = datos["var2"]+1
```

[5]:

	var1	var2	var3
0	1	12	374

[8]:
```
nuevosdatos = datos["var2"]+1
nuevosdatos
```

[8]:
```
0     13
1     14
2     36
3     33
4     74
5     28
6     83
7     55
8     26
9     54
10    74
Name: var2, dtype: int64
```

9	6	53	235
10	7	73	235

Python

Seleccionar columnas en grupo:

Para seleccionar columnas podemos usar el nombre de la columna:

```
datos2 = datos[["var1", "var3"]]
```

```
[9]: datos2 = datos[["var1", "var3"]]
     datos2
```

[9]:

	var1	var3
0	1	374
1	1	725
2	2	355
3	2	332
4	3	836
5	3	247
6	4	247
7	5	247
8	6	353
9	6	235
10	7	235

Seleccionar filas individuales

Llamado “datos”

[5]:

	var1	var2	var3
0	1	12	374
1	1	13	725
2	2	35	355
3	2	32	332
4	3	73	836
5	3	27	247
6	4	82	247
7	5	54	247
8	6	25	353
9	6	53	235
10	7	73	235

Python

Seleccionar filas

Para modificar de acuerdo a
podemos usar la función w
numpy

Así podemos modificar
simultáneamente según la
lógica.

```
[28]: import numpy as np
datos["var2"] = np.where(datos["var2"] < 30, "bajo", "alto")
datos
```

[28]:

	var1	var2	var3
0	1	bajo	374
1	1	bajo	725
2	2	alto	355
3	2	alto	332
4	3	alto	836
5	3	bajo	247
6	4	alto	247
7	5	alto	247
8	6	bajo	353
9	6	alto	235
10	7	alto	235

Python

Filtrar observaciones al hacer cálculos

Para desarrollar cálculos con un subgrupo de observaciones, seleccionamos primero y luego llamamos la función sobre el objeto DataFrame resultante (todo en una línea):

```
datos[datos["var3"] > 300].mean()
```

[10]:

	var1	var2	var3
0	1	12	374
1	1	13	725
2	2	35	355
3	2	32	332
4	3	73	836
8	6	25	353
5	3	27	247

[12]:
```
datos[datos["var3"] > 300].mean()
```

[12]:
```
var1      2.500000
var2     31.666667
var3    495.833333
dtype: float64
```

10	7	73	235

Python

Repetir operaciones por subgrupos de filas

En pandas empleamos

`groupby`

1. Definimos el grupo
2. Asignamos según la función y el grupo correspondientes

[41]:
```
# Primero definimos el grupo y la columna que vamos a usar
gb = datos.groupby("var1")["var3"]
# Luego asignamos según el grupo
datos["var4"] = gb.transform("mean")
datos
```

[41]:

	var1	var2	var3	var4
0	1	12	374	549.5
1	1	13	725	549.5
2	2	35	355	343.5
3	2	32	332	343.5
4	3	73	836	541.5
5	3	27	247	541.5
6	4	82	247	247.0
7	5	54	247	247.0
8	6	25	353	294.0
9	6	53	235	294.0
10	7	73	235	235.0

Python

Crear columnas

Para crear columnas solo hay que asignarlas con un nuevo nombre en el dataframe:

```
datos["nueva"] = 1
```

```
[43]: datos["nueva"] = 1
      datos
```

[43]:

	var1	var2	var3	nueva
0	1	12	374	1
1	1	13	725	1
2	2	35	355	1
3	2	32	332	1
4	3	73	836	1
5	3	27	247	1
6	4	82	247	1
7	5	54	247	1
8	6	25	353	1
9	6	53	235	1
10	7	73	235	1

Reemplazar valores en columnas

Simplemente tomamos la columna y le asignamos valores nuevos

```
datos["var3"] = 0
```

```
[46]: datos["var3"] = 0
      datos
```

[46]:

	var1	var2	var3
0	1	12	0
1	1	13	0
2	2	35	0
3	2	32	0
4	3	73	0
5	3	27	0
6	4	82	0
7	5	54	0
8	6	25	0
9	6	53	0
10	7	73	0

Python

Eliminar columnas

Para eliminar columnas podemos usar el comando `del` y dirigirlo a nuestra columna del DataFrame

```
del datos["var2"]
```

```
[48]: del datos["var2"]
      datos
```

[48]:

	var1	var3
0	1	374
1	1	725
2	2	355
3	2	332
4	3	836
5	3	247
6	4	247
7	5	247
8	6	353
9	6	235
10	7	235

Python

Eliminar filas

Podemos asignar al mismo dataframe una versión filtrada del mismo dataframe:

```
datos = datos[datos["var3"] > 300]
```

[5]:

	var1	var2	var3

[50]:

```
datos = datos[datos["var3"] > 300]
datos
```

[50]:

	var1	var2	var3
0	1	12	374
1	1	13	725
2	2	35	355
3	2	32	332
4	3	73	836
8	6	25	353
8	6	25	353
9	6	53	235
10	7	73	235

R

DataFrames

Ahora tenemos un DataFrame de R

Llamado “datos”

R

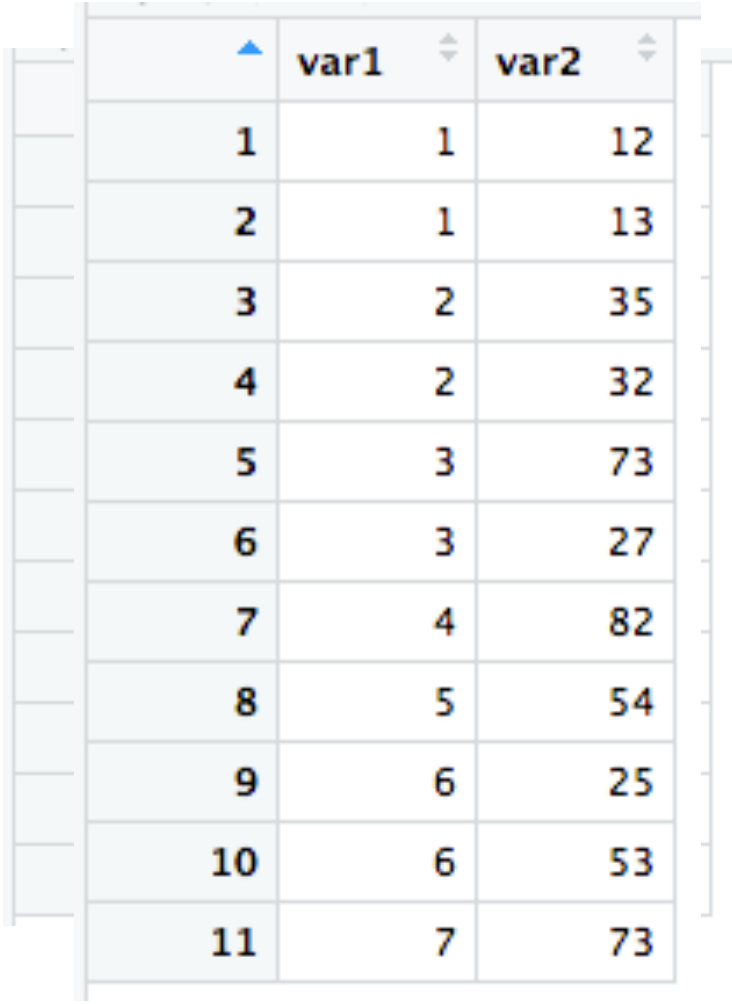
	var1	var2	var3
1	1	12	374
2	1	13	725
3	2	35	355
4	2	32	332
5	3	73	836
6	3	27	247
7	4	82	247
8	5	54	247
9	6	25	353
10	6	53	235
11	7	73	235

Seleccionar columnas

Para seleccionar columnas podemos usar el nombre de la columna:

R

```
variables <- c("var1", "var2")
nuevosDatos <- datos[variables]
```



	var1	var2
1	1	12
2	1	13
3	2	35
4	2	32
5	3	73
6	3	27
7	4	82
8	5	54
9	6	25
10	6	53
11	7	73

Seleccionar columnas

Una selección simple de una columna funciona mediante $

R

```
datos$var3
```

```
> datos$var3
 [1] 374 725 355 332 836 247 247 247 353 235 235
```

	var1	var2	var3
1	1	12	374
2	1	13	725
3	2	35	355
4	2	32	332
5	3	73	836
6	3	27	247
7	4	82	247
8	5	54	247
9	6	25	353
10	6	53	235
11	7	73	235

Seleccionar filas individuales

R

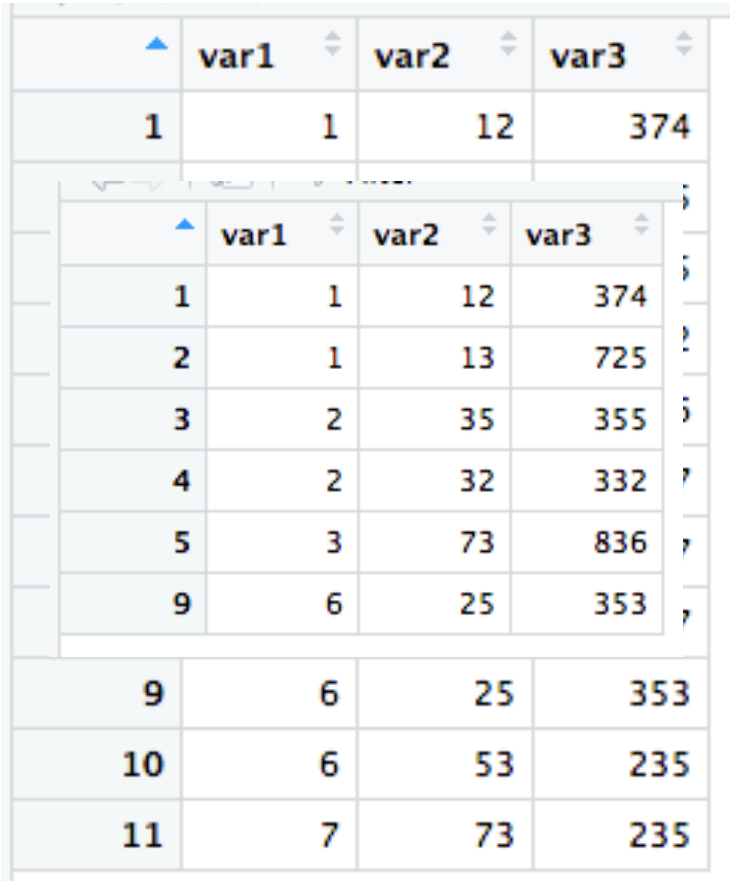
Podemos seleccionar con filas y columnas

```
datos[filas, columnas]
```

así:

```
datos[1:5, ]
```

Significa filas del 1 al 5 y columnas todas. También puedo:

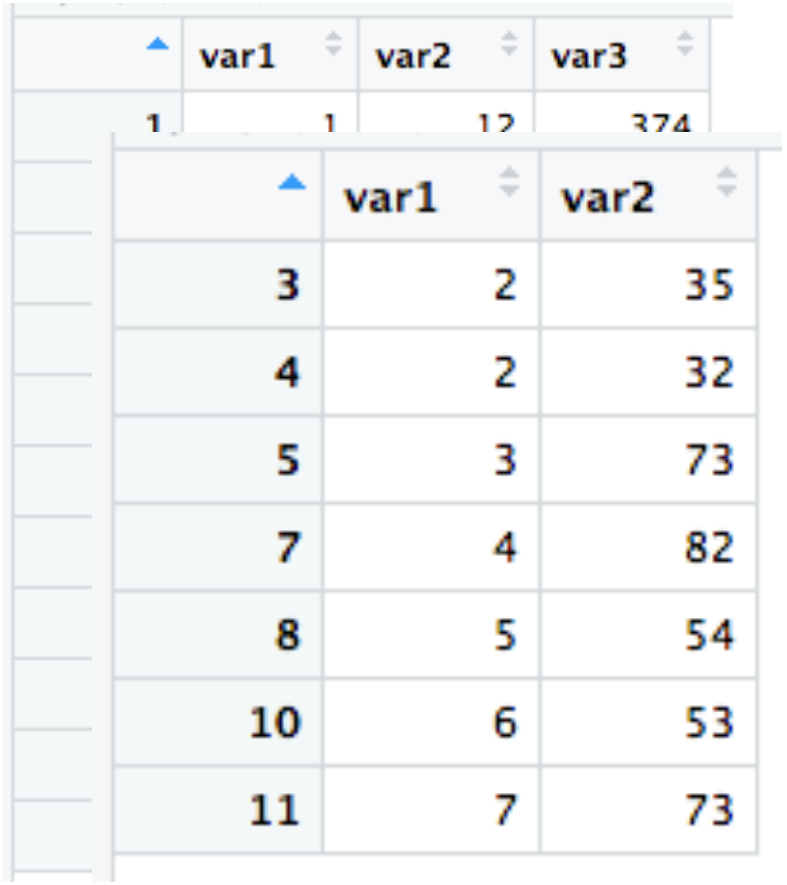


	var1	var2	var3
1	1	12	374
2	1	13	725
3	2	35	355
4	2	32	332
5	3	73	836
9	6	25	353
9	6	25	353
10	6	53	235
11	7	73	235

```
nuevosDatos <- datos[ which(datos$var3>300) , ]
```

Seleccionar filas grupales

La forma más práctica de seleccionar es mediante la función subset:



	var1	var2	var3
1	1	12	374

	var1	var2
3	2	35
4	2	32
5	3	73
7	4	82
8	5	54
10	6	53
11	7	73

```
nuevosDatos <- subset(datos, var2 >= 30, select=c(var1, var2))
```

R

Filtrar observaciones

Para filtrar observaciones al hacer cálculos, podemos simplemente pasar como parámetro a las funciones los DataFrames ya filtrados:

	var1	var2	var3
1	1	12	374
2	1	13	725
3	2	35	355
4	2	32	332
9	6	25	353
10	6	53	235
11	7	73	235

```
> sapply(datos, mean, na.rm=TRUE)
     var1       var2       var3
 3.636364  43.545455 380.545455
> sapply(datos[ which(datos$var3>300) , ], mean, na.rm=TRUE)
    var1      var2      var3
 2.50000  31.66667 495.83333
```

```
sapply(datos, mean, na.rm=TRUE)

sapply(datos[ which(datos$var3>300) , ], mean, na.rm=TRUE)
```

R

Crear columnas

Al igual que en Python, podemos simplemente referirnos a una columna con un nombre nuevo y asisgnarla

```
datos$var4 <- 2
```

R

	var1	var2	var3	var4
1	1	12	374	2
2	1	13	725	2
3	2	35	355	2
4	2	32	332	2
5	3	73	836	2
6	3	27	247	2
7	4	82	247	2
8	5	54	247	2
9	6	25	353	2
10	6	53	235	2
11	7	73	235	2

Reemplazar valores en columnas

Asignamos refiriéndonos a las columnas correspondientes.

R

```
datos$var3 <- datos$var3 > 300
```

	var1	var2	var3
1	1	12	TRUE
2	1	13	TRUE
3	2	35	TRUE
4	2	32	TRUE
5	3	73	TRUE
6	3	27	FALSE
7	4	82	FALSE
8	5	54	FALSE
9	6	25	TRUE
10	6	53	FALSE
11	7	73	FALSE

Eliminar filas

Asignamos un subconjunto de filas al mismo DataFrame:

R

```
datos <- datos[ which(datos$var3>300) , ]
```

	var1	var2	var3
1	1	12	374
2	1	13	725
3	2	35	355
4	2	32	332
5	3	73	836
9	6	25	353
10	6	53	235
11	7	73	235

Eliminar columnas

Asignamos NULL a la columna correspondiente

```
datos$var2 <- NULL
```

También puedo asignar un subconjunto con el símbolo -c(columna)

```
datos <- subset(datos, select = -c(var2) )
```

	var1	var3
1	1	374
2	1	725
3	2	355
4	2	332
5	3	836
6	3	247
7	4	247
8	5	247
9	6	353
10	6	235
11	7	235

R