

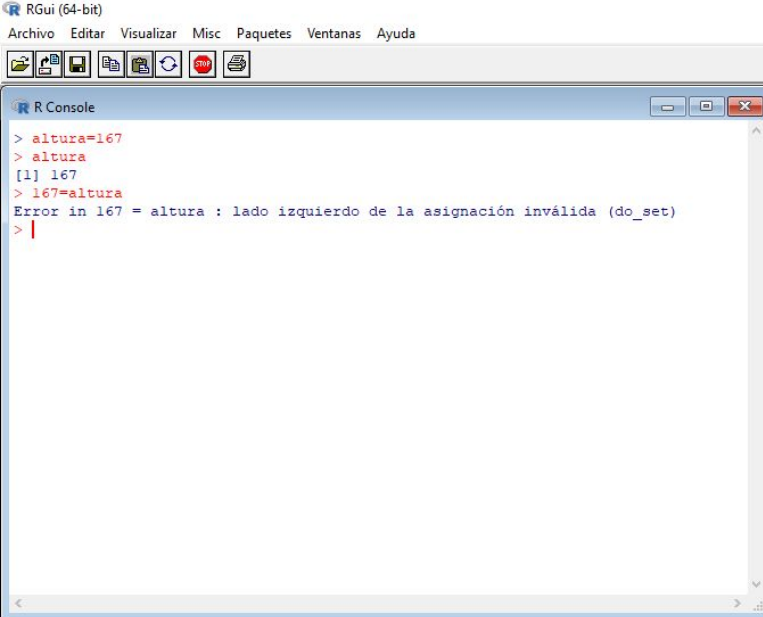
VARIABLES Y OPERACIONES

1. Variables

Para asignar el valor a una variable, tenemos dos opciones:

- altura=167
- altura<-167

¡Cuidado!: Si lo asignamos al revés, el programa nos dará error y nos avisará con un mensaje como este:



```
RGui (64-bit)
Archivo  Editar  Visualizar  Misc  Paquetes  Ventanas  Ayuda
R Console
> altura=167
> altura
[1] 167
> 167=altura
Error in 167 = altura : lado izquierdo de la asignación inválida (do_set)
> |
```

Si el valor que queremos asignar es una palabra, debemos emplear comillas simples (' ') o dobles (" "), ya que si no ponemos las comillas el programa nos dará error.

```
RGui (64-bit)
Archivo  Editar  Visualizar  Misc  Paquetes  Ventanas  Ayuda

R Console
> comida='galletas'
> comida
[1] "galletas"
> comida=galletas
Error: objeto 'galletas' no encontrado
> |
```

Comandos de variables útiles:

- **ls():** Nos indica las variables almacenadas en la memoria.
- **ls.str():** nos permite obtener información sobre los objetos que hay almacenados en la memoria.
- **ls(pat='x')** donde x es un carácter establecido anteriormente: nos indica la variable (objetos) que contiene ese carácter.
- **ls(pat="^x")** (donde x es un carácter cualquiera): nos indica los objetos cuyo nombre empiezan por ese carácter.
- **rm():** nos permite eliminar los elementos (objetos) almacenados en la memoria.
- **rm(yy)** (donde yy es un objeto concreto establecido anteriormente): nos permite eliminar ese objeto concreto.

Operaciones con variables

Operadores aritméticos:

- ❖ Suma: se utiliza el símbolo +
- ❖ Resta: se utiliza el símbolo -
- ❖ División: se utiliza el símbolo /
- ❖ Multiplicación: se utiliza el símbolo *
- ❖ Exponente: se puede utilizar el símbolo ^ y también el símbolo **

En R también se sigue la jerarquía de las operaciones:

1º realiza las potencias.

2º realiza las multiplicaciones y divisiones (esto es siempre así al no ser que haya paréntesis).

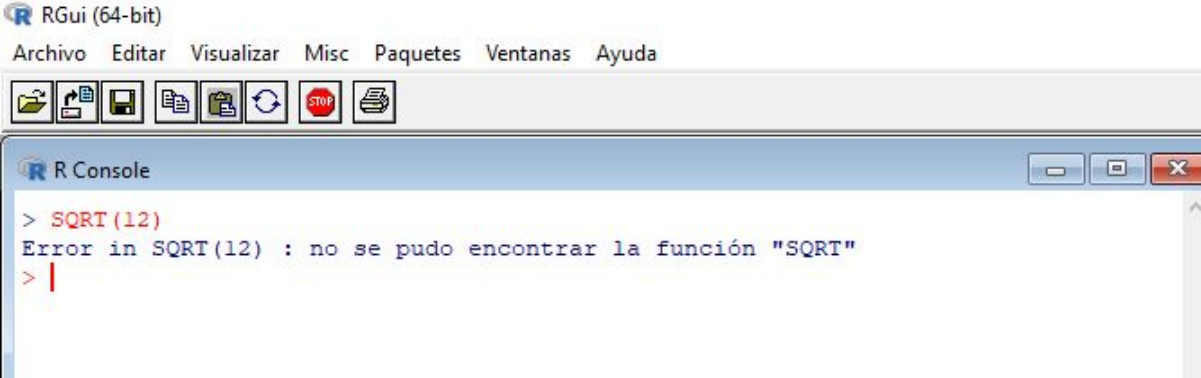
3º realiza las sumas y restas.

Cuando hay varias cosas del mismo rango, como por ejemplo multiplicaciones y divisiones, se va de izquierda a derecha.

Funciones matemáticas

- Funciones logarítmicas:
 - ❖ **log(x)**: logaritmo neperiano
 - ❖ **log10(x)**: logaritmo en base 10
 - ❖ **logb(x)**: logaritmo en cualquier base
- **exp(x)**: función exponencial
- Funciones trigonométricas:
 - ❖ **sin(x)**
 - ❖ **cos(x)**
 - ❖ **tan(x)**
 - ❖ **asin(x)**
 - ❖ **acos(x)**
 - ❖ **atan(x)**
- Otras funciones:
 - ❖ **abs(x)**: valor absoluto
 - ❖ **sqrt(x)**: raíz cuadrada
 - ❖ **factorial(x)**: factorial

¡Cuidado!: es muy importante escribir estas funciones en minúsculas porque si las escribimos en mayúsculas nos daría un error como el siguiente:



```
RGui (64-bit)
Archivo  Editar  Visualizar  Misc  Paquetes  Ventanas  Ayuda
[Icons]
R Console
> Sqrt(12)
Error in Sqrt(12) : no se pudo encontrar la función "Sqrt"
> |
```

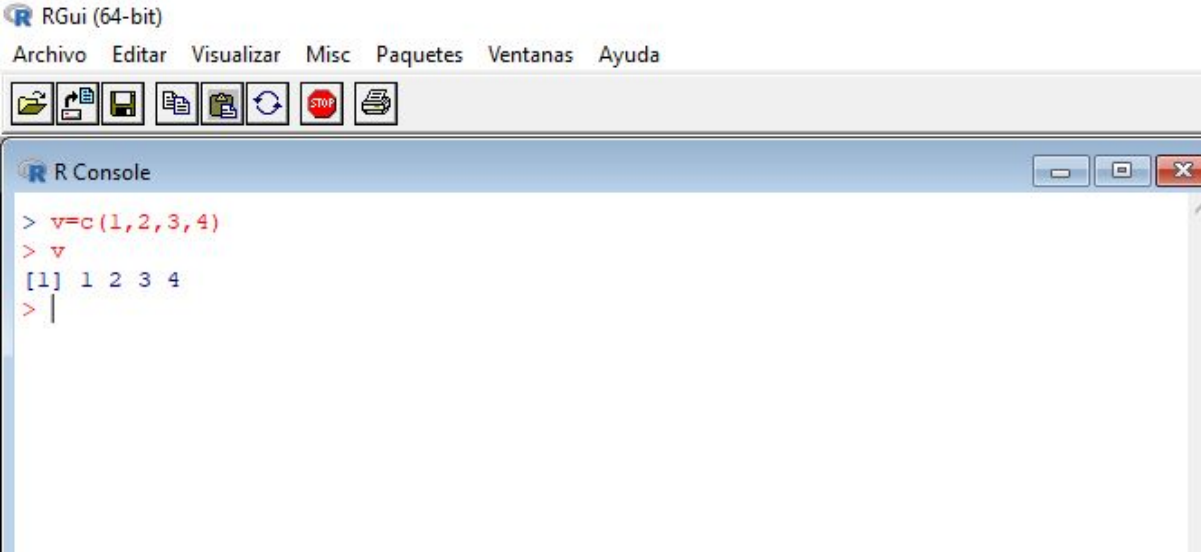
2. Vectores

Para escribir un vector, debemos hacerlo de la siguiente manera:

- Poner las componentes del vector entre paréntesis.
- Escribir la letra "c" después del igual.
- Si la variable que queremos introducir es una palabra, debemos ponerla entre comillas.

❖ (ejemplo: fruta=('manzana' , 'fresa' , 'kiwi'))

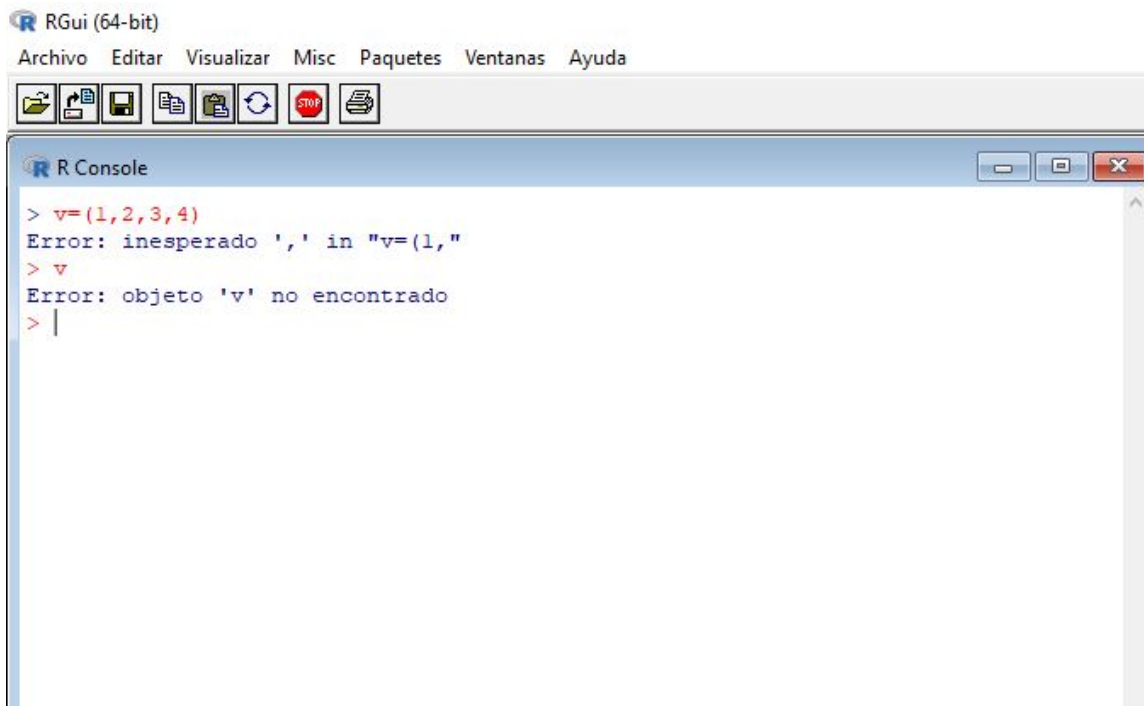
- **Ejemplo:** tenemos el siguiente vector $v=(1,2,3,4)$ y queremos escribirlo en R, para ello debemos hacerlo de la siguiente manera:



```
RGui (64-bit)
Archivo  Editar  Visualizar  Misc  Paquetes  Ventanas  Ayuda
[Icons: Home, Copy, Paste, Save, Print, Refresh, Stop, Print]

R Console
> v=c(1,2,3,4)
> v
[1] 1 2 3 4
> |
```

¡Cuidado!: si no ponemos la c antes de escribir las componentes del vector nos dará el siguiente error:



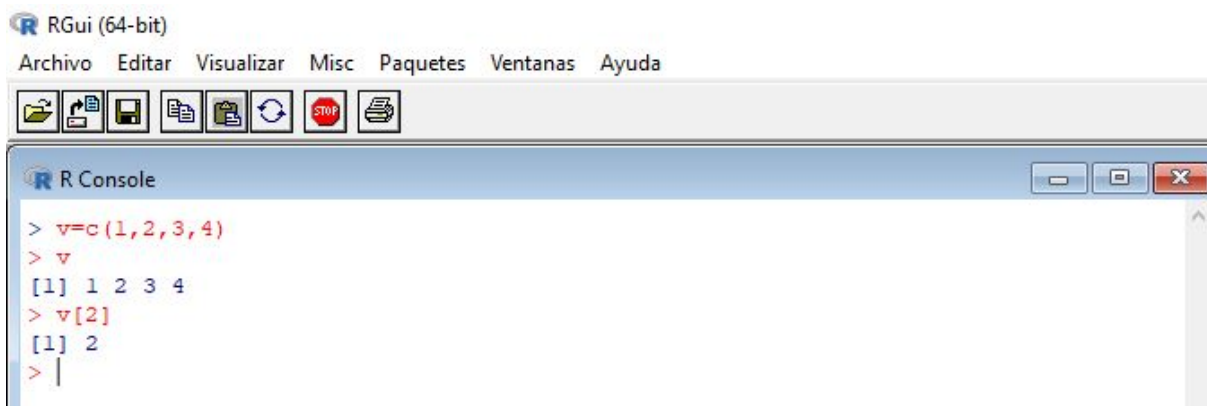
Operaciones con vectores:

- Para mostrar la componente de un vector (como en el ejemplo anterior, que está formado por cuatro componentes) debemos escribirlo de la siguiente manera:

v[posición de la componente del vector]

◆ **Ejemplo:**

v[2]=2 (el resultado obtenido es 2 porque corresponde a la segunda variable del vector).



- Para realizar la suma, resta y división utilizamos los operadores de siempre (+,-,/)
- Para realizar el producto escalar, debemos emplear la siguiente notación: $v \% \% w$ (donde v es el 1º vector y w el 2º)
- Para realizar el producto vectorial (producto de los dos vectores, componente a componente, usamos *).

❖ **Ejemplo:** tenemos los siguientes vectores y tenemos que realizar su producto escalar y el producto componente a componente: $v=(1,2,3,4)$ y $w=(5.67, \sqrt{17}, -14, 0)$. Debemos hacerlo de la siguiente manera:

```

RGui (64-bit)
Archivo  Editar  Visualizar  Misc  Paquetes  Ventanas  Ayuda
[Icons]
R Console
> #Producto escalar de dos vectores
> v=c(1,2,3,4)
> v
[1] 1 2 3 4
> w=c(5.67,sqrt(17),-14,0)
> w
[1] 5.670000 4.123106 -14.000000 0.000000
> v%%w
      [,1]
[1,] -28.08379
> #Producto Vectorial de dos vectores (producto de los dos vectores componente a componente)
> v*w
[1] 5.670000 8.246211 -42.000000 0.000000
> |

```

3. Matrices

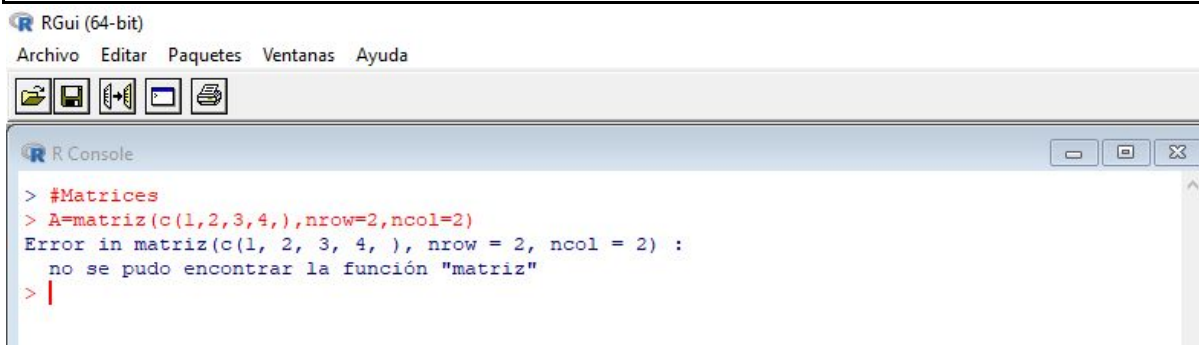
Para escribir una matriz, debemos hacerlo de la siguiente manera:

A=matrix(c(1,2,3,4), nrow=2, ncol=2)

donde los valores 1,2,3 y 4 son los datos que deben aparecer en la matriz y estos pueden ser cualquier número.

- **nrow**: es el número de filas que queremos que tenga la matriz.
- **ncol**: es el número de columnas que queremos que tenga la matriz.

¡Cuidado! Hay que escribir **matrix** de la forma correcta, puesto que si ponemos **matriz** con **z** nos saldrá un error como el siguiente.



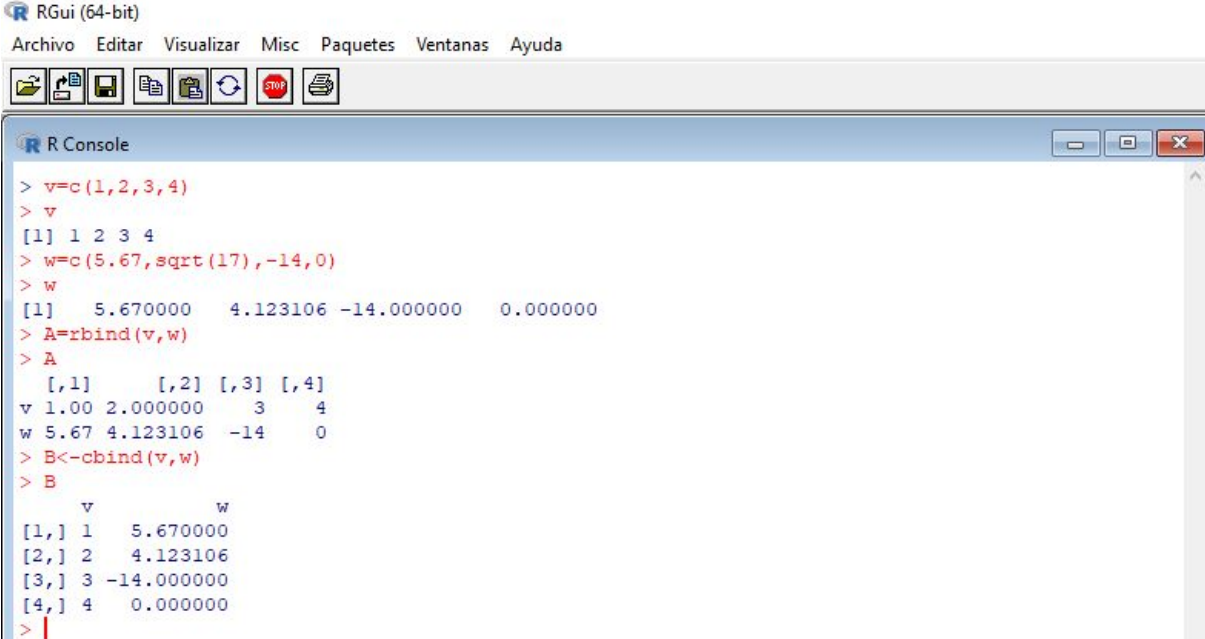
```
> #Matrices
> A=matriz(c(1,2,3,4),nrow=2,ncol=2)
Error in matriz(c(1, 2, 3, 4, ), nrow = 2, ncol = 2) :
  no se pudo encontrar la función "matriz"
> |
```

Operaciones con matrices:

- Para realizar el determinante, emplearemos: **det(A)**
- **t(A)**: permite calcular la transpuesta de la matriz A.
- **solve(A,b)**: nos permite obtener la solución del sistema de ecuaciones $Ax=b$.
- **solve(A)**: nos permite obtener la inversa de la matriz A.
- **svd(A)**: nos permite descomponer la matriz en valores singulares.
- **eigen(A)**: nos permite obtener valores y vectores propios de la matriz.
- **diag(b)**: sirve para calcular la matriz diagonal (b es un vector).
- **diag(A)**: sirve para calcular la matriz diagonal (A es una matriz).
- Suma: empleamos el **+**
- Resta: empleamos el **-**
- División: empleamos el **/**
- Producto de dos matrices: empleamos el siguiente operador **%*%** (para que se pueda realizar el producto es necesario que coincidan el número de columnas de la primera matriz con el número de filas de la segunda).
- Producto de matrices elemento a elemento: para ello empleamos el siguiente operador *****.

También podemos generar una matriz, a partir de varios vectores empleando los siguientes comandos:

- **rbind**= se utiliza para generar una matriz combinando los vectores en forma de filas respectivamente.
- **cbind**= se utiliza para generar una matriz combinando los vectores en forma de columnas.
 - ❖ **Ejemplo:** tenemos estos dos vectores: $v=(1,2,3,4)$ y $w=(5.67, \sqrt{17}, -14, 0)$ y queremos construir dos matrices a partir de ellos usando los comandos `cbind` y `rbind`.



```
RGui (64-bit)
Archivo Editar Visualizar Misc Paquetes Ventanas Ayuda

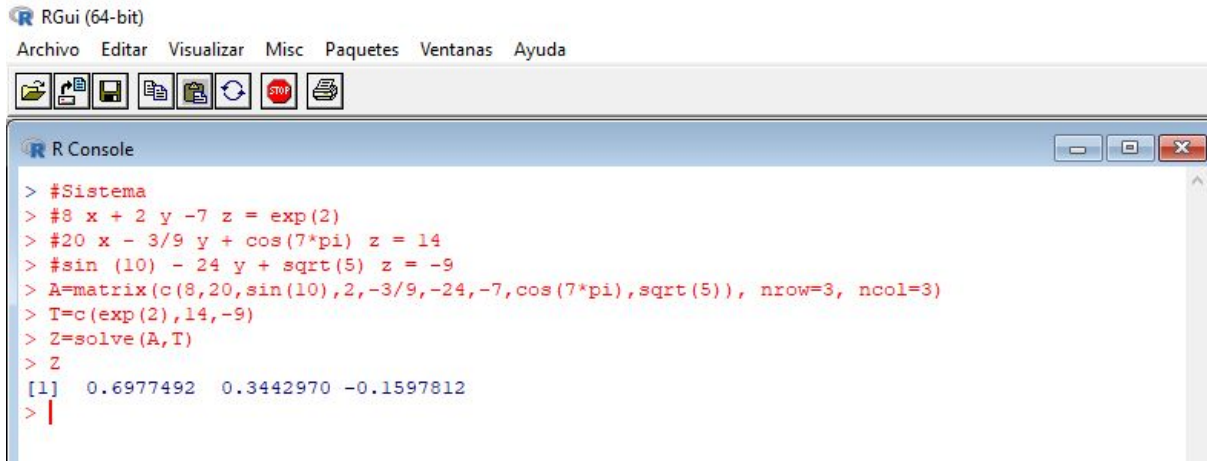
R Console
> v=c(1,2,3,4)
> v
[1] 1 2 3 4
> w=c(5.67,sqrt(17),-14,0)
> w
[1] 5.670000 4.123106 -14.000000 0.000000
> A=rbind(v,w)
> A
  [,1] [,2] [,3] [,4]
v 1.00 2.000000 3 4
w 5.67 4.123106 -14 0
> B<-cbind(v,w)
> B
  v w
[1,] 1 5.670000
[2,] 2 4.123106
[3,] 3 -14.000000
[4,] 4 0.000000
> |
```

Sistemas de ecuaciones:

Para resolver sistemas de ecuaciones emplearemos matrices. Los pasos a seguir son los siguientes:

1. Crearemos una matriz que contenga los coeficientes de las ecuaciones:
 $A = \text{matrix}(c(\text{coeficientes de las } x, \text{coeficientes de las } y, \text{coeficientes de las } z), \text{nrow} = n^\circ \text{ de filas}, \text{ncol} = n^\circ \text{ de columnas})$.
2. Creamos otro vector **T** que contenga los términos independientes de las ecuaciones.
3. Escribimos un nuevo vector (por ejemplo **Z**) que incluya el comando **$\text{solve}(A, T)$** ; lo debemos escribir de la siguiente manera: **$Z = \text{solve}(A, T)$** .
4. Ejecutar.

- ❖ **Ejemplo:** dado el sistema de ecuaciones representado en la foto, resolverlo empleando el comando solve(A,T).



```
> #Sistema
> #8 x + 2 y -7 z = exp(2)
> #20 x - 3/9 y + cos(7*pi) z = 14
> #sin(10) - 24 y + sqrt(5) z = -9
> A=matrix(c(8,20,sin(10),2,-3/9,-24,-7,cos(7*pi),sqrt(5)), nrow=3, ncol=3)
> T=c(exp(2),14,-9)
> Z=solve(A,T)
> Z
[1] 0.6977492 0.3442970 -0.1597812
> |
```

¡Cuidado! hay que poner # delante de las ecuaciones porque sino nos da error.

❑ **Comando data.frame**

Este comando nos permite realizar tablas en R, ya que nos permite almacenar diferentes tipos de datos.

- ❖ **Ejemplo:** tenemos los siguientes tres vectores y con ellos queremos generar una tabla empleando el comando data.frame:
niños=('Juan','Luis','Claudia','Andrea','Hector')
edad=(12,7,9,5,14)
altura=(165,167,160,172,180)

Tenemos que escribir lo siguiente **colegio=data.frame('niños','edad','altura')** para poder realizar la tabla.

Los pasos a seguir son los representados en la fotografía:

- 1º Escribir los tres vectores
- 2º Escribir el objeto colegio (pero podríamos haber escrito otro nombre) para almacenar en él el comando data.frame y que nos permita almacenar los datos de los tres vectores.
- 3º Ejecutarlo.

RGui (64-bit)

Archivo Editar Paquetes Ventanas Ayuda



R Console

```
> #Data.frame
> alumnos=c('Juan','Luis', 'Claudia', 'Andrea', 'Héctor')
> edad=c(12,7,9,5,14)
> altura=c(165,167,160,172,180)
> Colegio=data.frame(alumnos,edad,altura)
> Colegio
  alumnos edad altura
1   Juan   12   165
2   Luis    7   167
3 Claudia    9   160
4  Andrea    5   172
5 Héctor   14   180
> |
```