2. FUNCIONES EN R

2.1 FUNCIONES ARITMÉTICAS

El programa R utiliza un lenguaje similar al de una calculadora: por pantalla se puede digitar la operación que se requiere, así, se puede realizar desde una suma hasta calcular un logaritmo determinado. En la tabla 1 se muestran algunas operaciones con su respectiva instrucción y un ejemplo:

Tabla 1. Funciones aritméticas básicas

Operación	Símbolo	Ejemplo	Descripción
Suma y Resta	+ y -	> 3 +2	Realiza la suma o resta de dos cantidades
Multiplicación	*	> 6 * 5	Realiza la multiplicación de dos cantidades
División	/	> 9 / 3	Realiza la división entre dos cantidades
Potenciación	^	> 3 ^ 5	Devuelve la potencia de un número
Raíz cuadrada	sqrt()	> sqrt(81)	Devuelve la raíz cuadrada de un número
Logaritmo	log(x,base)	> log(64,8)	Calcula el logaritmo de x en base "base"
Seno	sin()	> sin(radianes)	Calcula el seno para un ángulo determinado
Coseno	cos()	> cos(radianes)	Calcula el coseno para un ángulo determinado
Tangente	tan()	> tan(radianes)	Calcula la tangente para un ángulo determinado

2.2 COMANDOS ESPECIALES

2.2.1 Comando help(). Permite obtener ayuda sobre funciones específicas; se necesita tener el nombre de la función sobre la cual se desea obtener información; para utilizar esta ayuda se procede así:

help(función) o alternativamente **?función**

2.2.2 Asignación. Consiste en dar un nombre a un valor o a una determinada función, de tal manera que esta pueda ser utilizada más adelante en otras operaciones o con otras funciones más complicadas. La estructura para realizar la asignación es la siguiente:

Nombre<-valor o función
X<- 5 * 9 + 2

Nótese que en el anterior comando se escribe (<-) después del nombre al que se quiere asignar el valor o la función, esto es equivalente a utilizar (=); para ilustrar esto, en algunos comandos más adelante se trabaja con el (=) en lugar de (<-). Con lo anterior se ha asignado a un nombre específico un valor o función que se almacena en la memoria del computador, y si se quiere observar esta asignación en la pantalla se invoca colocando nuevamente el nombre de la asignación y dando un enter. La asignación puede realizarse también mediante la función assign(). Una forma equivalente de realizar la asignación anterior es

assign("X",5 * 9 + 2)

2.2.3 Comando c(). R utiliza diferentes estructuras de datos. La estructura más simple es el vector, que es una colección ordenada de números. Para crear un vector columna de cualquier dimensión, escriba los números dentro de los paréntesis del comando separados por comas; por ejemplo, si se desea un vector columna de tamaño cinco se procede así:

Y=c(1,2,3,4,5) o assign("Y",c(1,2,3,4,5))

2.2.4 Comando seq(). Permite generar sucesiones de números; para ello se debe indicar el número inicial, el número final y el incremento que se desee; la estructura para este comando es como sigue:

seq(valor inicial, valor final, incremento)
seq(1,5,0.5) "genera la sucesión de números del
1 al 5 con incrementos de 0.5"

```
>
> # Secuencia de 1 a 5 incrementando en 0.5
> seq(1,5,0.5)
[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
>
```

Imagen 8. Salida R para generar secuencias

Si se necesita que esta sucesión se almacene dentro de un vector, lo anterior se escribe dentro del comando **c()** y se le asigna un nombre, para su utilización más adelante.

2.2.5 Comando rep(). Permite repetir el mismo número tantas veces como se desee; si se requiere obtener el resultado del comando dentro de un vector, entonces se utiliza el comando **c()** con su asignación respectiva, ejemplo:

2.3 VECTORES

2.3.1 Definición. Un vector es todo elemento de un espacio vectorial. Los vectores surgen como elementos de ciertas estructuras matemáticas previamente definidas. R utiliza diferentes estructuras de datos; la estructura más simple es el vector, que R considera como una colección ordenada de números. Un número, por sí mismo, se considera un vector de longitud uno.

Y=c(rep(x,número de veces))
Unos=c(rep(1,5)) "repite el 1 cinco veces"

```
>
> Unos=c(rep(1,5))
> Unos
[1] 1 1 1 1 1
>
```

Imagen 9. Salida R para repeticiones

2.3.2 Aritmética vectorial. Los vectores pueden usarse en expresiones aritméticas, en cuyo caso las operaciones se realizan elemento tras elemento. Dos vectores que se utilizan en la misma expresión no tienen por qué ser de la misma longitud. Si no lo son, el resultado será un vector de la longitud del más largo, y el más corto será reciclado, repitiéndolo tantas veces como sea necesario (puede que no un número exacto de veces) hasta que coincida con el más largo.

Los operadores aritméticos elementales son los habituales +, -, *, / y ^, para elevar a una potencia. Además, están disponibles las funciones log, exp, sin, cos, tan, sqrt, entre otras; estas se utilizan en operaciones aritméticas, y, además, se aplican a todos los elementos del vector. Si se tiene un vector llamado X, es posible definir las siguientes funciones:

Tabla 2. Funciones aritméticas vectoriales

Comando	Descripción
sum(x)	Suma de los elementos del vector
prod(x)	Multiplica los elementos del vector
max(x)	Valor máximo del vector
min(x)	Valor mínimo del vector
range(x)	Rango del vector o c(min(x),max(x))
length(x)	Número de elementos del vector
sort(x)	Ordena de menor a mayor los elementos del vector
rev(sort(x))	Ordena de mayor a menor los elementos del vector
round(vector,n)	Redondea los elementos del vector a n cifras decimales
cumsum(x)	Vector donde cada elemento es la suma de él y sus cifras anteriores

A continuación se ilustran algunos de los comandos anteriores con ejemplos:

Ejemplo: Determinar el número de elementos de un vector

B=c(6,1,2,5,4,8,7,9,6,3)	`>`
	`> B=c(6,1,2,5,4,6,7,9,6,3)`
	`> length(B)`
	`[1] 10`
	`>`

Imagen 10. Salida R para determinar el tamaño de un vector

Ejemplo: Realizar la suma de cada número en un vector con los números que le anteceden.

```
>
> X=c(1,2,3,4,5,6,7,8,9,10)
> cumsum(X)
[1]  1  3  6 10 15 21 28 36 45 55
>
```

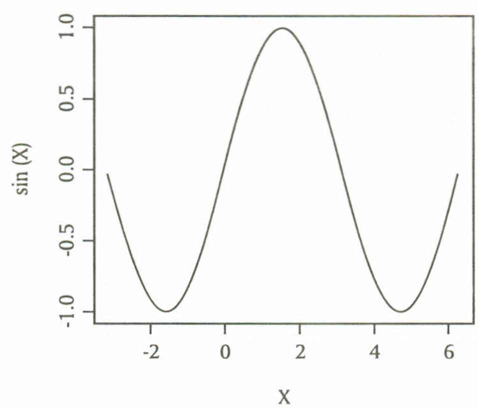
Imagen 11. Salida R para el comando vectorial cumsum

2.3.3 Gráficas de funciones. En R también es posible realizar gráficas de funciones conocidas como: seno (sin), coseno (cos), tangente (tan), exponenciales (exp) y casi cualquier clase de funciones; a continuación se muestran algunos ejemplos de estas gráficas.

Para realizar una gráfica de una función trigonométrica se utiliza el siguiente comando **plot(función, Rango)**; El argumento después de la función corresponde al rango, ejemplo:

Función Seno

```
>
> plot(sin, -pi, 2*pi)
>
```

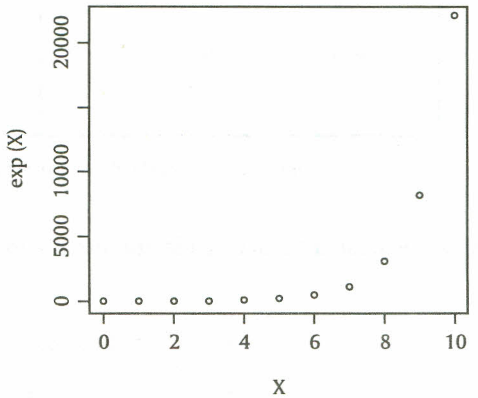


Imágenes 12 y 13. Salida R para creación de función Seno

Para las demás funciones es necesario construir un vector con los valores iniciales que se quieren evaluar mediante una función específica; el comando utilizado es **plot(vector inicial, función)**.

Función Exponencial

```
>
> x=c(0,1,2,3,4,5,6,7,8,9,10)
> plot(x,y=exp(x))
```

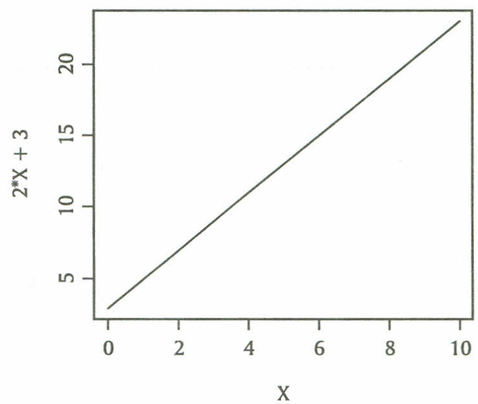


Imágenes 14 y 15. Salida R para creación de función Exponencial

Si el interés está en que los puntos de la gráfica aparezcan conectados mediante una línea, en el comando **plot()** se agrega la instrucción "l". Si se desea realizar la gráfica de la recta **y = 2 x + 3** utilizando el vector **x** del ejemplo anterior se procede así:

Recta y = 2 x + 3

```
>
> plot(x,y=2*x+3,"l")
> .
```



Imágenes 16 y 17. Salida R para creación de función lineal

2.3.4 Vectores lógicos. Los elementos de un vector lógico solo pueden tomar dos valores: FALSE (falso) y TRUE (verdadero). Los vectores lógicos aparecen al utilizar condiciones; los operadores lógicos son < (menor), = (menor o igual), > (mayor), = (mayor o igual), == (igual), y != (distinto). Además, si c1 y c2 son expresiones lógicas, entonces c1 & c2 es su intersección (conjunción), c1 | c2 es su unión (disyunción) y !c1 es la negación de c1. Por ejemplo, dado un vector (1, 2, 3, 4, 5) se establece la condición x > 3.

```
>
> X=c(1,2,3,4,5)
> mayores=x>3
> mayores
[1] FALSE FALSE FALSE TRUE TRUE
>
```

Imagen 18. Salida R para evaluación de valores lógicos

2.3.5 Vectores de caracteres. Las cadenas de caracteres, o frases, también son utilizadas en R; por ejemplo, para etiquetar gráficos. Una cadena de caracteres se construye escribiendo entre comillas la sucesión de caracteres que la define; por ejemplo: "Altura" o "Peso". La función **paste()** une todos los vectores de caracteres que se le suministran y construye una sola cadena de caracteres.

2.4 MATRICES

2.4.1 Definición. Si m y n son enteros positivos, entonces una matriz mxn (que se lee "m por n") es un arreglo rectangular. Una matriz mxn tiene m filas (líneas horizontales) y n columnas (líneas verticales).

$$A = [ij] = \begin{bmatrix} a_{11} & a_{12} & . & . & a_{1n} \\ a_{21} & a_{22} & . & . & a_{2n} \\ . & . & & & . \\ . & . & & & . \\ a_{m1} & a_{m2} & . & . & a_{mn} \end{bmatrix}$$

Imagen 19. Matriz

En R, una matriz es realmente un vector con un atributo adicional, dimensión (dim), el cual a su vez es un vector numérico de longitud 2, que define el número de filas y columnas de la matriz; además, el tamaño del vector debe ser igual al producto del número de filas por el número de columnas. Una matriz se puede crear con la función **matrix()**, teniendo los datos dentro de un vector, así:

matrix(vector de datos, #filas, #columnas)

Por defecto, R ordena los elementos del vector de datos en términos de vectores columna; si se desea realizar el ordenamiento del vector de datos por fila, se incorpora dentro del comando anterior la instrucción byrow=T.

```
>
> X=c(1,2,3,4,5,6)
> matrix(X,2,3)
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> |
```

```
>
> matrix(X,2,3,byrow=T)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
> .
```

Imágenes 20 y 21. Salida R para creación de matrices

En R existe otra forma para crear una matriz; la creación de dicha matriz es posible si se tiene un conjunto de vectores que, luego, con el comando **cbind()**, se encadenan en un arreglo rectangular, en donde cada columna representa un vector. Si se requiere que los vectores representen una fila de la matriz se utiliza el comando **rbind()**.

```
>
> x1=c(1,2,3)
> x2=c(4,5,6)
> x3=c(7,8,9)
> X=cbind(x1,x2,x3)
> X
     x1 x2 x3
[1,]  1  4  7
[2,]  2  5  8
[3,]  3  6  9
> .
```

Imagen 22. Salida R para creación de matrices con cbind

2.4.2 Matriz Identidad. Es aquella cuyos elementos en su diagonal principal son unos y los demás elementos son ceros. Para crear esta matriz en R se procede de la siguiente forma:

diag(n), donde n indica el orden de la matriz

```
>
> Y=diag(3)
> Y
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
> .
```

Imagen 23. Salida R para creación de matriz Identidad

También, al trabajar con la función diag(), si su argumento es una matriz, **diag(matriz)**, devuelve un vector formado por los elementos de la diagonal de esta. Si, por el contrario, su argumento es un vector (de longitud mayor que uno), **diag(vector)**, lo transforma en una matriz diagonal cuyos elementos en la diagonal principal son los elementos del vector. Por último, si se requiere una matriz diagonal diferente a la identidad, entonces dentro del comando **diag()** se escribe primero el número que debe aparecer en la diagonal y luego el número que corresponde al orden de la matriz, así por ejemplo:

Construcción de una matriz diagonal de tamaño cinco con el número cuatro en su diagonal

```
>
> diag(4,5)
     [,1] [,2] [,3] [,4] [,5]
[1,]    4    0    0    0    0
[2,]    0    4    0    0    0
[3,]    0    0    4    0    0
[4,]    0    0    0    4    0
[5,]    0    0    0    0    4
> .
```

Imagen 24. Salida R para creación de matriz diagonal

2.5 OPERACIONES ARITMÉTICAS CON MATRICES

Las matrices pueden utilizarse en expresiones aritméticas, y el resultado es una matriz formada a partir de las operaciones elemento tras elemento de las matrices involucradas. Las dimensiones de los operadores deben ser iguales en general y coincidirán con la dimensión de la matriz resultado; luego se tiene:

A + B, realiza la suma elemento a elemento de las matrices A y B
A - B, realiza la resta elemento a elemento de las matrices A y B
A * B, realiza la multiplicación elemento a elemento de las matrices A y B
A / B, realiza la división elemento a elemento de las matrices A y B

$$\begin{bmatrix} 2 & 5 & 6 \\ 3 & 2 & 1 \\ 8 & 4 & 5 \end{bmatrix} + \begin{bmatrix} 6 & 1 & 4 \\ 2 & 1 & 1 \\ 1 & 3 & 2 \end{bmatrix} = \begin{bmatrix} 8 & 6 & 10 \\ 5 & 3 & 2 \\ 9 & 7 & 7 \end{bmatrix}$$

```
>
> X=matrix(c(2,3,8,5,2,4,6,1,5),3)
> Y=matrix(c(6,2,1,1,1,3,4,1,2),3)
> Z=X+Y
> Z
     [,1] [,2] [,3]
[1,]    8    6   10
[2,]    5    3    2
[3,]    9    7    7
>
```

Imágenes 25 y 26. Salida R para suma de matrices

2.5.1 Multiplicación por un escalar. Los números suelen denominarse escalares; estos serán números reales, a no ser que se determine otra cosa; la multiplicación de un escalar por una matriz se realiza al multiplicar el escalar por cada uno de los elementos que componen la matriz. En R, la multiplicación de una matriz por un escalar se lleva a cabo mediante la siguiente secuencia:

Teniendo en cuenta la matriz Z del ejemplo anterior:
L = 5 * Z , realiza la multiplicación del escalar 5 por la matriz Z

```
> L = 5*Z
> L
     [,1] [,2] [,3]
[1,]   40   30   50
[2,]   25   15   10
[3,]   45   35   35
>
```

Imagen 27. Salida R para la multiplicación por escalar

2.5.2 Multiplicación de matrices. Si A = $[a_{ij}]$ es una matriz mxn y B = $[b_{ij}]$ es una matriz nxp, entonces, el producto AB es una matriz mxp. Nótese que para que el producto esté definido, el número de columnas de la primera matriz debe ser igual al número de filas de la segunda matriz. En R, el operador "%*%" permite realizar el producto entre dos matrices; un ejemplo del uso de este operador se muestra a continuación:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 2 \end{bmatrix} \text{ y } B = \begin{bmatrix} 1 & 2 & 0 \\ 2 & 1 & 1 \end{bmatrix}$$

```
>
> A=matrix(c(1,3,2,2),2)
> B=matrix(c(1,2,2,1,0,1),2,3)
> C=A%*%B
> C
     [,1] [,2] [,3]
[1,]    5    4    2
[2,]    7    8    2
>
```

Imágenes 28 y 29. Salida R para el producto de matrices

Una matriz de nx1 ó 1xn puede ser utilizada como un vector n dimensional en caso necesario. Análogamente, R puede usar automáticamente un vector en una operación matricial, convirtiéndolo en una matriz fila o una matriz columna cuando ello es posible.

2.5.3 Traspuesta de una matriz. La traspuesta de una matriz se forma al escribir sus columnas como filas. Por ejemplo, si A es la matriz de orden mxn, entonces la traspuesta, denotada por A^t, es la matriz de orden nxm. El comando que permite calcular la traspuesta de una matriz en R es **t(nombre de la matriz)**.

Matriz C

```
> C
     [,1] [,2] [,3]
[1,]    5    4    2
[2,]    7    8    2
>
```

Traspuesta de C

```
> t(C)
     [,1] [,2]
[1,]    5    7
[2,]    4    8
[3,]    2    2
>
```

Imágenes 30 y 31. Salida R para obtener la traspuesta

2.5.4 Inversa de una matriz. Una matriz A nxn es invertible (o no singular) si hay una matriz B nxn tal que AB = BA = In, donde In es la matriz identidad de orden n. La matriz B se denomina inversa multiplicativa de A. La inversa de A se denota A^{-1}. Una matriz que no tiene inversa se denomina no invertible (o singular). El comando utilizado en R para encontrar la matriz inversa es **solve(matriz)**. A continuación se ilustra esto.

$$Sea\ D = \begin{bmatrix} 1 & 3 & 4 \\ 2 & 5 & 1 \\ 3 & 1 & 2 \end{bmatrix}$$

```
>
> D=matrix(c(1,2,3,3,5,1,4,1,2),3)
> Inv=solve(D)
> Inv
            [,1]        [,2]        [,3]
[1,] -0.19565217  0.04347826  0.36956522
[2,]  0.02173913  0.21739130 -0.15217391
[3,]  0.28260870 -0.17391304  0.02173913
> .
```

Imágenes 32 y 33. Salida R para calcular la matriz Inversa

2.5.5 Determinante. Toda matriz cuadrada puede asociarse con un número real denominado su determinante. Si A es una matriz cuadrada (de orden mayor o igual a 2), entonces el determinante de A es la suma de los elementos en el primer renglón de A multiplicados por sus cofactores. En R, el comando que permite calcular el determinante de una matriz cuadrada es **det(nombre de la matriz)**; recuerde que si el determinante de una matriz es cero, la matriz se denomina singular.

$$Sea\ A = \begin{bmatrix} 4 & 1 \\ 3 & 2 \end{bmatrix}$$

```
>
> A=matrix(c(4,3,1,2),2)
> det(A)
[1] 5
>
```

Imágenes 34 y 35, Salida R para repeticiones

2.5.6 Valores y vectores propios. En álgebra lineal, los vectores propios, autovectores o eigenvectores de un operador lineal son los vectores no nulos, que cuando son transformados por el operador dan lugar a un múltiplo escalar de sí mismos, con lo que no cambian su dirección. Este escalar ë recibe el nombre de valor propio, autovalor, valor característico o eigenvalor. A menudo, una transformación queda completamente determinada por sus vectores propios y valores propios. Un espacio propio, autoespacio o eigenespacio, es el conjunto de vectores propios con un valor propio común. En R, el comando que permite calcular los valores y vectores propios de una matriz es **eigen(matriz)**; si se desea que solo aparezcan los valores propios dentro del comando anterior luego de la matriz, se le da la instrucción **only.values=TRUE**, es decir: **eigen(matriz,only.values=TRUE)**, otra forma equivalente es **eigen(matriz) $val**. Por defecto, el comando anterior arroja los vectores normalizados, si se requiere se puede pedir que estos vectores estén sin normalizar, incluyendo dentro del comando la instrucción **EISPACK=TRUE**. Por ejemplo, al considerar la matriz A utilizada anteriormente, se tiene:

Vectores normalizados	Vectores sin normalizar

<pre>>
> A=matrix(c(4,3,1,2),2)
> eigen(A)
$Values
[1] 5 1
 	
$vectors
          [,1]   [,2]
[1,] 0.7071068 -0.3162278
[2,] 0.7071068 0.9486833</pre>
	<pre>> eigen(A,EISPACK=TRUE)
	$values
	[1] 5 1
	
	$vectors
	          [,1] [,2]
	[1,] 0.7071068 -0.3535534
	[2,] 0.7071068 1.0606602</pre>

Imágenes 36 y 37: Salida R para valores y vectores propios

2.5.7 Comando apply(). Este comando permite aplicar una función específica a las filas o columnas de una matriz; esta selección se puede realizar en el comando, mediante la dimensión (1 = fila, 2 = columna); ejemplo:

Apply (matriz, dimensión, función) Apply (B,2,sum), calcula la suma de cada columna de la matriz B	<pre>>
	> B=matrix(c(1,2,3,4,5,6,7,8,9,10,11,12),4,3)
	> B
	     [,1] [,2] [,2]
	[1,] 1 5 9
	[2,] 2 6 10
	[3,] 3 7 11
	[4,] 4 8 12
	> apply(B,2,sum)
	[1] 10 26 42
	></pre>

Imagen 38. Salida R para aplicación comando apply

2.5.8 Partición de matrices. En algunos casos se tienen variables dentro de un arreglo matricial y se desea trabajar solamente con una de estas variables; para esto se hace necesario particionar la matriz; en R es posible tomar un solo elemento de la matriz, una columna, una fila o un arreglo matricial de menor dimensión que la matriz inicial. A continuación se muestra el comando utilizado para realizar lo descrito anteriormente; considérese la siguiente matriz:

```
>
> A=matrix(c(1,2,3,4,5,6,7,8,9),3)
> A
     [,1] [,2] [,2]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
>
```

Imagen 39. Salida R para creación de matriz

A continuación se presentan algunas particiones de la matriz A:

```
> # Elemento primera fila, segunda columna
>
> A[1,2]
[1] 4
>
> # Tomar una columna de la matriz
> # Columna número 3
>
> A[,3]
[1] 7 8 9
>
> # Tomar una fila de la matriz
> # Fila número 2
>
> A[2,]
[1] 2 5 8
>


>
> # Tomar un arreglo matricial
> # filas 1 y 2, columnas 2y3.
>
> A[1:2.2:3]
     [,1] [,2]
[1,]    4    7
[2,]    5    8
```

Imagen 40. Salida R para particionar una matriz

A continuación se presentan ejercicios para ser desarrollados en R, y en el siguiente capítulo se presentan algunos temas de estadística descriptiva que pueden ser trabajados en R.

2.6 EJERCICIOS

2.6.1 Con los siguientes datos construya un vector y asígnele la letra N.

43 76 30 90 12 38 74 46 15 45 25 67 53

2.6.2 Genere una secuencia de números que inicie en 8 y termine en 100, con incrementos de 2.

2.6.3 Los siguientes datos hacen referencia al peso de algunos estudiantes encuestados:

58 54 40 47 60 61 43 50 42 49 53 39 41
51 70 48 43 55 44 60 52 46 49 42 52 48

Obtenga:

a. El número de estudiantes encuestados
b. La suma de los pesos de los estudiantes
c. El peso menor y el mayor
d. Ordene los pesos de menor a mayor

2.6.4 Dada la Matriz A

$$\begin{bmatrix} 2 & 7 & 0 \\ 4 & 3 & 1 \\ 6 & 8 & 2 \end{bmatrix}$$

a. Construya esta matriz en R
b. Multiplique la matriz A por el escalar -3
c. Construya la matriz traspuesta de A
d. Calcule A^{-1}
e. Calcule el determinante de A
f. Realice el producto matricial entre A y A^{-1}