

DATA ANALYSIS WITH R

LES STRUCTURES DE DONNES

Arsène W. KIEMA Fidel KPOGHOMOU

kiemaarsene@gmail.com / kpoghomoufidel@gmail.com

9/5/24

Vecteurs

Les vecteurs sont des séquences ordonnées d'éléments du même type. Ils peuvent être de type numérique, caractère, logique, etc.

Création de vecteurs

```
# Exemples de vecteurs
vec_numerique <- c(1, 2, 3, 4, 5)
vec_caractere <- c("a", "b", "c")
vec_logique <- c(TRUE, FALSE, TRUE)

# Générer une séquence de nombres
sequence <- 1:10
```

Les vecteurs

Accéder aux éléments

```
vecteur <- c(10,15,12,-6,98,NA,20,25,32,12,13,4)
vecteur[1] # renvoie le premier élément du vecteur
```

```
[1] 10
```

```
vecteur[10] # renvoie le 10e élément du vecteur
```

```
[1] 12
```

Les Vecteurs

Accéder aux éléments

```
vecteur[2:5]
```

```
[1] 15 12 -6 98
```

```
vecteur[vecteur>20]
```

```
[1] 98 NA 25 32
```

Les vecteurs

Modifier les éléments d'un vecteurs

```
vecteur[5] <- 0 # remplace la 5e valeur par 0  
vecteur[is.na(vecteur)] <- -99 # remplace les valeurs manquantes par  
print(vecteur)
```

```
[1] 10 15 12 -6 0 -99 20 25 32 12 13 4
```

Les vecteurs

Quelques fonctions vitales

```
# Construction de vecteur  
mon_vec<- c(8, 4 , 5 , 6 , 10 , 21, -8, 32, 41)
```

Les vecteurs

Quelques fonctions vitales

```
# Somme  
sum(mon_vec)  
  
[1] 119
```

Les vecteurs

Quelques fonctions vitales

```
# Somme
```

```
sum(mon_vec)
```

```
[1] 119
```

```
# Moyenne
```

```
mean(mon_vec)
```

```
[1] 13.22222
```


Les vecteurs

Quelques fonctions vitales

```
# Somme
```

```
sum(mon_vec)
```

```
[1] 119
```

```
# Moyenne
```

```
mean(mon_vec)
```

```
[1] 13.22222
```

```
# Minimum
```

```
min(mon_vec)
```

```
[1] -8
```

Les vecteurs

Quelques fonctions vitales

Somme

```
sum(mon_vec)
```

```
[1] 119
```

Moyenne

```
mean(mon_vec)
```

```
[1] 13.22222
```

Minimum

```
min(mon_vec)
```

```
[1] -8
```

Maximum

```
max(mon_vec)
```

```
[1] 41
```

Les vecteurs

Quelques fonctions vitales

Somme

```
sum(mon_vec)
```

```
[1] 119
```

Moyenne

```
mean(mon_vec)
```

```
[1] 13.22222
```

Minimum

```
min(mon_vec)
```

```
[1] -8
```

Maximum

```
max(mon_vec)
```

```
[1] 41
```

Variance

```
var(mon_vec)
```

```
[1] 234.6944
```

Les vecteurs

Quelques fonctions vitales

Somme

```
sum(mon_vec)
```

```
[1] 119
```

Moyenne

```
mean(mon_vec)
```

```
[1] 13.22222
```

Minimum

```
min(mon_vec)
```

```
[1] -8
```

Maximum

```
max(mon_vec)
```

```
[1] 41
```

Variance

```
var(mon_vec)
```

```
[1] 234.6944
```

Ecart-type

```
sd(mon_vec)
```

```
[1] 15.31974
```

Les vecteurs

Autres fonctions

<code>length(x)</code>	# nombre d'éléments dans x
<code>head(x)</code>	# 6 premiers éléments de x
<code>tail(x)</code>	# 6 derniers éléments de x
<code>rev(x)</code>	# éléments de x dans l'ordre inverse
<code>t(x)</code>	# transpose et converti x en matrix
<code>sort(x)</code>	# trie x en ordre croissant
<code>sort(x, decr=T)</code>	# trie x en ordre décroissant
<code>order(x)</code>	# renvoie le rang des éléments de x
<code>unique(x)</code>	# renvoie les éléments uniques contenus dans x

LET'S PRACTICE...



Les Vecteurs

Exercice 1 : Création de Séquences Complexes

Créez un vecteur "sequence" contenant une séquence de nombres de 10 à 100, en incrémentant de 5 à chaque étape.

Exercice 2 : Moyenne des Températures

- 1- Créez un vecteur "temperatures" contenant les températures moyennes en Celsius pour chaque jour d'une semaine : `c(20, 22, 23, 25, 21, 18, 19)`.
- 2- Calculez la moyenne des températures pour cette semaine.

Exercice 3 : Conversion de Devises

- 1-Créez un vecteur "euros" contenant des montants en euros : `c(100, 50, 75, 200, 30)`. Supposons que le taux de change est de 1 euro = 650 F CFA.
- 2-Convertissez les montants en F CFA.

Les vecteurs

Exercice 4 : Extraction de Sous-Vecteur

- 1- Créez un vecteur "elements" contenant des caractères : `c("chat", "chien", "oiseau", "poisson", "souris")`.
- 2-Extrayez les éléments du deuxième au quatrième élément pour obtenir un sous-vecteur.

Exercice 5 : Filtrage et Indexation

- 1-Créez un vecteur "nombres" contenant des nombres de 1 à 20.
- 2-Affichez les nombres pairs du vecteur.

Construction de matrix

Les matrices sont des tableaux bidimensionnels de données, avec des lignes et des colonnes. Tous les éléments d'une matrice doivent être du même type.

```
matrice <- matrix(data = 1:20,nrow = 4,ncol = 5,byrow = TRUE)
```

- data : les données à utiliser pour remplir la matrix -nrow: nombre de lignes de la matrix

Construction de matrix

Les matrices sont des tableaux bidimensionnels de données, avec des lignes et des colonnes. Tous les éléments d'une matrice doivent être du même type.

```
matrice <- matrix(data = 1:20, nrow = 4, ncol = 5, byrow = TRUE)
```

- data : les données à utiliser pour remplir la matrix -nrow: nombre de lignes de la matrix
- ncol: nombre de colonnes de la matrix

Construction de matrix

Les matrices sont des tableaux bidimensionnels de données, avec des lignes et des colonnes. Tous les éléments d'une matrice doivent être du même type.

```
matrice <- matrix(data = 1:20,nrow = 4,ncol = 5,byrow = TRUE)
```

- data : les données à utiliser pour remplir la matrix -nrow: nombre de lignes de la matrix
- ncol: nombre de colonnes de la matrix
- byrow: indique si le remplissage doit se faire par ligne (TRUE) ou par colonne (FALSE)

Matrix

Construction de matrix

```
mat1 <- matrix(data = 1:20,nrow = 4,ncol = 5,byrow = TRUE)
View(mat1) # visualiser la matrix
head(mat1) # visualiser les 6 premiers éléments de la matrix
tail(mat1)# visualiser les 6 derniers éléments de la matrix
```

Les dimensions de la matrix

```
# Vérifier les dimensions d'une matrix  
nrow(mat1) # Renvoie le nombre de ligne de la matrix  
  
[1] 4
```

Matrix

Les dimensions de la matrix

```
# Vérifier les dimensions d'une matrix
```

```
nrow(mat1) # Renvoie le nombre de ligne de la matrix
```

```
[1] 4
```

```
ncol(mat1) # Renvoie le nombre de colonne de la matrix
```

```
[1] 5
```

Matrix

Les dimensions de la matrix

```
# Vérifier les dimensions d'une matrix
```

```
nrow(mat1) # Renvoie le nombre de ligne de la matrix
```

```
[1] 4
```

```
ncol(mat1) # Renvoie le nombre de colonne de la matrix
```

```
[1] 5
```

```
dim(mat1) # Renvoie le nombre de ligne et colonne de la matrix
```

```
[1] 4 5
```

Matrix

Extraire des éléments dans une matrix

```
# Extraire une cellule  
print(mat1)
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	1	2	3	4	5
[2,]	6	7	8	9	10
[3,]	11	12	13	14	15
[4,]	16	17	18	19	20

```
mat1[2,3] # Extraire une cellule  
[1] 8
```


Matrix

Extraire des éléments dans une matrix

```
# Extraire une cellule  
print(mat1)
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	1	2	3	4	5
[2,]	6	7	8	9	10
[3,]	11	12	13	14	15
[4,]	16	17	18	19	20

```
# Extraire des lignes  
mat1[2,]
```

[1]	6	7	8	9	10
-----	---	---	---	---	----

```
mat1[c(1,3),]
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	1	2	3	4	5
[2,]	11	12	13	14	15

Matrix

Extraire des éléments dans une matrix

```
print(mat1)
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	1	2	3	4	5
[2,]	6	7	8	9	10
[3,]	11	12	13	14	15
[4,]	16	17	18	19	20

Matrix

Extraire des éléments dans une matrix

```
print(mat1)
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	1	2	3	4	5
[2,]	6	7	8	9	10
[3,]	11	12	13	14	15
[4,]	16	17	18	19	20

```
# Extraire des colonnes
```

```
mat1[,3]
```

```
[1] 3 8 13 18
```

```
mat1[,1:2]
```

	[,1]	[,2]
[1,]	1	2
[2,]	6	7
[3,]	11	12
[4,]	16	17

Matrix

Extraire des éléments dans une matrix

```
print(mat1)
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	1	2	3	4	5
[2,]	6	7	8	9	10
[3,]	11	12	13	14	15
[4,]	16	17	18	19	20

Matrix

Extraire des éléments dans une matrix

```
print(mat1)
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	1	2	3	4	5
[2,]	6	7	8	9	10
[3,]	11	12	13	14	15
[4,]	16	17	18	19	20

```
# Extraire des lignes et colonnes  
mat1[c(1,3),c(2,4)]
```

	[,1]	[,2]
[1,]	2	4
[2,]	12	14

Matrix

Nommer les dimensions d'une matrix

```
set.seed(42)
mat.1 <- matrix(nrow = 6, ncol = 5,
                 data = sample(5:50, 30))
print(mat.1)
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	41	28	7	34	47
[2,]	5	11	50	42	15
[3,]	29	46	31	6	19
[4,]	14	48	8	12	37
[5,]	40	24	9	38	35
[6,]	22	30	17	49	45

Matrix

Nommer les dimensions d'une matrix

```
set.seed(42)
mat.1 <- matrix(nrow = 6, ncol = 5,
                 data = sample(5:50, 30))
print(mat.1)
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	41	28	7	34	47
[2,]	5	11	50	42	15
[3,]	29	46	31	6	19
[4,]	14	48	8	12	37
[5,]	40	24	9	38	35
[6,]	22	30	17	49	45

```
set.seed(52)
mat.2 <- matrix(sample(-20:20, 30),
                 6, 5)
print(mat.2)
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	-3	-18	2	0	-20
[2,]	-2	-4	15	-8	-6
[3,]	3	9	6	-17	-9
[4,]	-16	20	16	17	5
[5,]	-7	-13	8	-14	13
[6,]	7	10	18	-19	1

Matrix

Nommer les dimensions d'une matrix

```
colnames(mat.1) <- c(paste0("col",1:5))  
rownames(mat.1) <- c(paste0("line",1:6))  
print(mat.1)
```

	col1	col2	col3	col4	col5
line1	41	28	7	34	47
line2	5	11	50	42	15
line3	29	46	31	6	19
line4	14	48	8	12	37
line5	40	24	9	38	35
line6	22	30	17	49	45

Matrix

Nommer les dimensions d'une matrix

```
colnames(mat.1) <- c(paste0("col",1:5))  
rownames(mat.1) <- c(paste0("line",1:6))  
print(mat.1)  
colnames(mat.2) <- c(paste0("col_",1:5))  
rownames(mat.2) <- c(paste0("line_",1:6))  
print(mat.2)
```

	col1	col2	col3	col4	col5
line1	41	28	7	34	47
line2	5	11	50	42	15
line3	29	46	31	6	19
line4	14	48	8	12	37
line5	40	24	9	38	35
line6	22	30	17	49	45

	col_1	col_2	col_3	col_4	col_5
line_1	-3	-18	2	0	-20
line_2	-2	-4	15	-8	-6
line_3	3	9	6	-17	-9
line_4	-16	20	16	17	5
line_5	-7	-13	8	-14	13
line_6	7	10	18	-19	1

Matrix

Fusion horizontale

```
cbind(mat.1,mat.2)
```

	col1	col2	col3	col4	col5	col_1	col_2	col_3	col_4	col_5
line1	41	28	7	34	47	-3	-18	2	0	-20
line2	5	11	50	42	15	-2	-4	15	-8	-6
line3	29	46	31	6	19	3	9	6	-17	-9
line4	14	48	8	12	37	-16	20	16	17	5
line5	40	24	9	38	35	-7	-13	8	-14	13
line6	22	30	17	49	45	7	10	18	-19	1

Matrix

Fusion verticale

```
rbind(mat.1,mat.2)
```

	col1	col2	col3	col4	col5
line1	41	28	7	34	47
line2	5	11	50	42	15
line3	29	46	31	6	19
line4	14	48	8	12	37
line5	40	24	9	38	35
line6	22	30	17	49	45
line_1	-3	-18	2	0	-20
line_2	-2	-4	15	-8	-6
line_3	3	9	6	-17	-9
line_4	-16	20	16	17	5
line_5	-7	-13	8	-14	13
line_6	7	10	18	-19	1

Opérations sur les lines

```
mat.3 <- rbind(mat.1,mat.2)
mat.4 <- mat.3[c(sample(1:12,5)),]
print(mat.4)
```

	col1	col2	col3	col4	col5
line2	5	11	50	42	15
line6	22	30	17	49	45
line5	40	24	9	38	35
line3	29	46	31	6	19
line_2	-2	-4	15	-8	-6

Matrix

Opérations sur les lignes

```
mat.3 <- rbind(mat.1,mat.2)
mat.4 <- mat.3[c(sample(1:12,5)),]
print(mat.4)
```

	col1	col2	col3	col4	col5
line2	5	11	50	42	15
line6	22	30	17	49	45
line5	40	24	9	38	35
line3	29	46	31	6	19
line_2	-2	-4	15	-8	-6

```
# Moyenne par ligne
apply(mat.4,1,mean)
```

line2	line6	line5	line3	line_2
24.6	32.6	29.2	26.2	-1.0

Matrix

Opérations sur les lignes

```
mat.3 <- rbind(mat.1,mat.2)
mat.4 <- mat.3[c(sample(1:12,5)),]
print(mat.4)
```

	col1	col2	col3	col4	col5
line2	5	11	50	42	15
line6	22	30	17	49	45
line5	40	24	9	38	35
line3	29	46	31	6	19
line_2	-2	-4	15	-8	-6

```
# Moyenne par ligne
apply(mat.4,1,mean)
```

line2	line6	line5	line3	line_2
24.6	32.6	29.2	26.2	-1.0

```
# Somme par ligne
apply(mat.4,1,sum)
```

line2	line6	line5	line3	line_2
123	163	146	131	-5

Matrix

Opérations sur les colonnes

```
print(mat.4)
```

	col1	col2	col3	col4	col5
line2	5	11	50	42	15
line6	22	30	17	49	45
line5	40	24	9	38	35
line3	29	46	31	6	19
line_2	-2	-4	15	-8	-6

Matrix

Opérations sur les colonnes

```
print(mat.4)
```

	col1	col2	col3	col4	col5
line2	5	11	50	42	15
line6	22	30	17	49	45
line5	40	24	9	38	35
line3	29	46	31	6	19
line_2	-2	-4	15	-8	-6

```
# Moyenne par colonne
```

```
apply(mat.4,2,mean)
```

col1	col2	col3	col4	col5
18.8	21.4	24.4	25.4	21.6

Matrix

Opérations sur les colonnes

```
print(mat.4)
```

	col1	col2	col3	col4	col5
line2	5	11	50	42	15
line6	22	30	17	49	45
line5	40	24	9	38	35
line3	29	46	31	6	19
line_2	-2	-4	15	-8	-6

```
# Moyenne par colonne
```

```
apply(mat.4,2,mean)
```

col1	col2	col3	col4	col5
18.8	21.4	24.4	25.4	21.6

```
# Somme par colonne
```

```
apply(mat.4,2,sum)
```

col1	col2	col3	col4	col5
94	107	122	127	108

Matrix

Transposition de matrix

```
print(mat.4)
```

	col1	col2	col3	col4	col5
line2	5	11	50	42	15
line6	22	30	17	49	45
line5	40	24	9	38	35
line3	29	46	31	6	19
line_2	-2	-4	15	-8	-6

```
mat.5 = t(mat.4)
```

```
print(mat.5)
```

	line2	line6	line5	line3	line_2
col1	5	22	40	29	-2
col2	11	30	24	46	-4
col3	50	17	9	31	15
col4	42	49	38	6	-8
col5	15	45	35	19	-6

LET'S PRACTICE...



Exercice 1 : Création et Affichage

- 1-Créez une matrice `mat` de dimensions 5x4 contenant des nombres entiers.
- 2-Affichez la matrice `mat` dans la console.

Exercice 2 : Indexation et Extraction

- 1-Utilisez la matrice de l'exercice précédent (`mat`).
- 2-Accédez à l'élément situé à la deuxième ligne et troisième colonne.
- 3-Extrait la première ligne de la matrice.
- 4-Extrait les colonnes 2 et 4 de la matrice.
- 5- Extraire les valeurs supérieurs à 10 de la 2e colonne

Exercice 3 : Transposition

Transposez la matrice `mat` de l'exercice précédent et stockez le résultat dans `mat_transposed`.

Exercice 4 : Opérations Vectorielles sur les Colonnes

1- Utilisez la matrice `mat_transposed` de l'exercice précédent. 2- Calculez la somme des éléments de chaque colonne et stockez-les dans un vecteur.

Data Frame

Description

Un data frame est un tableau de donnée à deux dimensions. C'est d'une généralisation de la matrix.

A la différence d'une matrix, un data frame peut contenir des données plusieurs types (chaîne de caractère, numérique, facteur, date, etc.).

Les data frame sont beaucoup plus souple que les matrix. La plupart des fonctions qui s'appliquent sur les matrix peuvent être appliquées sur les data frame.

Il sera toujours possible de contenir une matrix en data frame mais l'inverse n'est pas toujours possible.

Data frame

Construction d'un data frame

La construction d'un data frame se fait avec la fonction `data.frame`.

```
df1 <- data.frame(Nom=paste0("Eleve",1:5),  
                  Math= c(12,14,8,15,18),  
                  Physique= c(05,16,8,11,15),  
                  Chimie=c(19,14,10,12,15))
```

Data frame

Construction d'un data frame

La construction d'un data frame se fait avec la fonction `data.frame`.

```
df1 <- data.frame(Nom=paste0("Eleve",1:5),  
                  Math= c(12,14,8,15,18),  
                  Physique= c(05,16,8,11,15),  
                  Chimie=c(19,14,10,12,15))  
print(df1)
```

	Nom	Math	Physique	Chimie
1	Eleve1	12	5	19
2	Eleve2	14	16	14
3	Eleve3	8	8	10
4	Eleve4	15	11	12
5	Eleve5	18	15	15

Data frame

Extraction de données dans un data frame

En plus de la méthode d'indice (indiciage) vu précédemment sur les matrix, il existe d'autres méthodes pour récupérer des données dans un data frame.

```
# Nommer les lignes
rownames(df1) <- c(paste0("line",1:5))

# Extraire à partir des noms des lignes
df1[c("line1","line5"),]
```

	Nom	Math	Physique	Chimie
line1	Eleve1	12	5	19
line5	Eleve5	18	15	15

Data frame

Extraction de données dans un data frame

En plus de la méthode d'indice (indiciage) vu précédemment sur les matrix, il existe d'autres méthodes pour récupérer des données dans un data frame.

```
# Nommer les lignes
```

```
rownames(df1) <- c(paste0("line",1:5))
```

```
# Extraire à partir des noms des lignes
```

```
df1[c("line1","line5"),]
```

	Nom	Math	Physique	Chimie
line1	Eleve1	12	5	19
line5	Eleve5	18	15	15

```
# Extraire à partir des noms de colonnes
```

```
df1[,c("Chimie","Physique")]
```

	Chimie	Physique
line1	19	5
line2	14	16
line3	10	8
line4	12	11
line5	15	15

Data frame

Extraction de données dans un data frame

En plus de la méthode d'indice (indiaçage) vu précédemment sur les matrix, il existe d'autres méthodes pour récupérer des données dans un data frame.

```
# Extraire une colonne  
#à partir du symbole $  
df1$Physique
```

```
[1]  5 16  8 11 15
```

Data frame

Extraction de données dans un data frame

En plus de la méthode d'indice (indixage) vu précédemment sur les matrix, il existe d'autres méthodes pour récupérer des données dans un data frame.

```
# Extraire une colonne  
#à partir du symbole $  
df1$Physique
```

```
[1]  5 16  8 11 15
```

```
# Extraire à partir d'une condition  
  
df1[df1$Math>=13.5,]
```

	Nom	Math	Physique	Chimie
line2	Eleve2	14	16	14
line4	Eleve4	15	11	12
line5	Eleve5	18	15	15

Data frame

Opération sur les lignes

```
df2 <- df1[,2:4]
```

```
df2
```

	Math	Physique	Chimie
line1	12	5	19
line2	14	16	14
line3	8	8	10
line4	15	11	12
line5	18	15	15

Data frame

Opération sur les lignes

```
df2 <- df1[,2:4]
```

```
df2
```

```
# moyenne en ligne
```

```
round(rowMeans(df2),2)
```

	Math	Physique	Chimie	line1	line2	line3	line4	line5
line1	12	5	19	12.00	14.67	8.67	12.67	16.00
line2	14	16	14					
line3	8	8	10					
line4	15	11	12					
line5	18	15	15					

Data frame

Opération sur les lignes

```
df2 <- df1[,2:4]
```

```
df2
```

```
# moyenne en ligne
```

```
round(rowMeans(df2),2)
```

	Math	Physique	Chimie
line1	12	5	19
line2	14	16	14
line3	8	8	10
line4	15	11	12
line5	18	15	15

line1	line2	line3	line4	line5
12.00	14.67	8.67	12.67	16.00

```
rowSums(df2) # sommee en ligne
```

line1	line2	line3	line4	line5
36	44	26	38	48

Data frame

Opération sur les colonnes

```
df2 <- df1[,2:4]
```

```
df2
```

	Math	Physique	Chimie
line1	12	5	19
line2	14	16	14
line3	8	8	10
line4	15	11	12
line5	18	15	15

Data frame

Opération sur les colonnes

```
df2 <- df1[,2:4]
```

```
df2
```

```
# moyenne en ligne
```

```
round(colMeans(df2),2)
```

	Math	Physique	Chimie
line1	12	5	19
line2	14	16	14
line3	8	8	10
line4	15	11	12
line5	18	15	15

Math	Physique	Chimie
13.4	11.0	14.0

```
colSums(df2) # sommee en ligne
```

Math	Physique	Chimie
67	55	70

Data frame

Opération sur les colonnes

```
# calcul l'écart-type par ligne  
sapply(df2,sd)
```

	Math	Physique	Chimie
	3.714835	4.636809	3.391165

sapply(X, FUN)

Data frame

Opération sur les colonnes

```
# calcul l'écart-type par ligne  
sapply(df2,sd)
```

	Math	Physique	Chimie
	3.714835	4.636809	3.391165

sapply(X, FUN)

donnée →

Data frame

Opération sur les colonnes

```
# calcul l'écart-type par ligne  
sapply(df2,sd)
```

	Math	Physique	Chimie
	3.714835	4.636809	3.391165

fonction

sapply(X, FUN)

donnée

Data frame

Remarque

Les fonctions `rowMeans`, `rowSums`, `colMeans`, `colSums` s'appliquent également sur les matrices.

LET'S PRACTICE...



Data Frame

Exercice 1 : Création et Affichage

- 1- Créez un data frame `donnees` avec trois colonnes : `nom`, `age` et `ville`.
- 2- Remplissez le data frame avec des informations factices pour au moins 5 personnes.
- 3- Affichez le data frame dans la console.

Exercice 2 : Tri et Sélection de Colonnes

- 1- Utilisez le data frame `donnees` de l'exercice 1.
- 2- Triez le data frame par ordre croissant d'âge.
- 3- Sélectionnez uniquement la colonne `nom` du data frame trié.

Data Frame

Exercice 3 : Ajout et Suppression de Lignes

- 1- Utilisez le data frame donnees de l'exercice 1.
- 2- Ajoutez une nouvelle personne au data frame.
- 3- Supprimez la dernière ligne du data frame.

Exercice 4 : Création de Variables Calculées

- 1- Utilisez le data frame donnees de l'exercice 1.
- 2- Calculez une nouvelle variable `annee_naissance` en soustrayant l'âge de 2023.
- 3- Ajoutez la variable calculée au data frame.

Liste

La liste un objet générique. C'est même un *super-objet*. Il permet de stocker des objets hétérogènes.

La fonction de base pour créer une liste est `list()`

Construction d'une

```
my_ls <- list(vect1=c("Math","Physique","Anglais"),  
             vect2=seq(10,50,5),  
             mat1=matrix(data =sample(1:100,20),  
                          nrow = 4,ncol = 5))
```

Quelques attributs de la liste

```
mode(my_ls) # mode de l'objet
```

```
[1] "list"
```

Quelques attributs de la liste

```
mode(my_ls) # mode de l'objet
```

```
[1] "list"
```

```
names(my_ls) # Noms des composantes de la liste
```

```
[1] "vect1" "vect2" "mat1"
```

Liste

Quelques attributs de la liste

```
mode(my_ls) # mode de l'objet
```

```
[1] "list"
```

```
names(my_ls) # Noms des composantes de la liste
```

```
[1] "vect1" "vect2" "mat1"
```

```
length(my_ls) # Nombre d'éléments qui constitue de la liste
```

```
[1] 3
```

Extraire des données dans une liste

L'extraction des données dans liste est approximativement identique aux data frame ou matrix à la différence qu'il faut utiliser deux crochet imbriqués plutôt qu'un.

```
my_ls[[i]] # extrait l'élément d'indice i
```

Extraire des données dans une liste

L'extraction des données dans liste est approximativement identique aux data frame ou matrix à la différence qu'il faut utiliser deux crochet imbriqués plutôt qu'un.

```
my_ls[[i]] # extrait l'élément d'indice i
```

```
my_ls[[1]] # extrait l'élément d'indice 1
```

```
[1] "Math"      "Physique" "Anglais"
```

Extraire des données dans une liste

On peut également extraire des éléments dans une liste en utilisant les noms des composantes ou le symbole \$

```
#Extraction à partir des noms  
my_ls[["mat1"]]
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	84	2	58	89	15
[2,]	19	72	80	30	70
[3,]	86	35	44	32	97
[4,]	92	36	59	56	96

Liste

Extraire des données dans une liste

On peut également extraire des éléments dans une liste en utilisant les noms des composantes ou le symbole \$

```
#Extraction à partir des noms  
my_ls[["mat1"]]
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	84	2	58	89	15
[2,]	19	72	80	30	70
[3,]	86	35	44	32	97
[4,]	92	36	59	56	96

```
# Extraction avec le symbole $  
my_ls$mat1
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	84	2	58	89	15
[2,]	19	72	80	30	70
[3,]	86	35	44	32	97
[4,]	92	36	59	56	96

Exercice 1 : Création et Affichage

1- Créez une liste appelée `ma_liste` contenant les éléments suivants : - Un vecteur numérique - Un vecteur de caractères - Une matrice - Un data frame 2- Affichez le contenu de la liste dans la console.

Exercice 2 : Indexation et Extraction

1- Utilisez la liste `ma_liste` de l'exercice précédent.
2- Accédez à l'élément matrice de la liste.
3- Extraire la deuxième valeur du vecteur de caractères.

**THANK YOU FOR YOUR
ATTENTATION**