# Feuille de TP #1 : *R* pour débutants

## 1. Demarrer *R*

On lance ***R*** en tapant dans la fenêtre *Unix* : `R`
Et on le quitte en tapant :

> **`quit()`**

ou

> **`q()`**

ou bien

> **`Control D.`**

Le prompt de ***R*** est par défaut le caractère'>', signifie qu'il est en attente d'une commande.
Les objets courants sont sauvés dans le fichier `'.RData'`.
Dans R, vous pouvez voir le répertoire de travail courant par :

> getwd()

Pour le modifier, on peut aussi utiliser la commande **`setwd (`**directoire**`)`** où directoire correspond au répertoire de travail. Ce qui va permettre au R de lire directement les fichiers de ce dossier, de conserver les résultats dans ce dossier et permettra de reprendre le travail dans ce dossier.

## 2. L'aide sous *R*

Pour obtenir l'aide pour une certaine fonction, par exemple **mean** (la fonction qui calcule la moyenne arithmétique), tapez :

```
> help (mean)
```

Les deux autres fonctions de recherche **`help.search()`** et **`apropos()`** nous permettent d'effectuer une recherche plus précise. Par exemple :

```
> help.search("matrix")
```

Ce qui énumère toutes les fonctions dont les pages help ont le titre ou des alias où le mot «matrix« apparaît.

```
> apropos("matrix")
```

Ce qui énumère tout les noms de fonctions qui incluent le texte «matrix«.

## 3. Syntaxe générale

On sépare les différentes commandes par des points virgules ' ; ' ou par des sauts de ligne.

> x <- 6 ; y <- 13

L'opérateur d'affectation est constitué de deux caractères '<' et '-' placés côte à côte et pointant sur l'objet recevant la valeur. Cet opérateur peut être orienté dans l'autre sens.
Exemple :
> a<-5 ;
ou
> 5->a ;

Pour ajouter des commentaires, on utilise le symbole dièse '#'
> a # Indique la valeur de a

## 4. Importer, exporter les données

- Pour lire des fichier de données on va utiliser la fonction **`read.table()`** qui a pour effet de créer un tableau de données. Si par exemple on a un fichier nommé *data.txt*, pour lire ce fichier on va utiliser la commande suivante :
> res <- read.table("data.txt")
Cette commande va créer un tableau de données nommé *res*, et les variables par défaut nommées V1,V2…, pourront être accédées individuellement par mydata$V1, mydata$V2 ,…, ou par mydata["V1"], mydata["V2"],…

Regardez le help des fonctions **`scan()`**, et **`read.fwf()`**.

- Pour enregistrer les données on va utiliser la fonction **`write.table()`** qui écrit dans un fichier un objet , d'habitude c'est un tableau de données mais cela peut très bien être un autre type d'objet (matrice, vecteur, … ).
Exemple :
write (x, file="resultat.txt"),
où *x* est le nom de l'objet, et *resultat.txt* est le nom du fichier où on va écrire les résultats.

## 5. Ecrire une fonction

La syntaxe générale de définition d'une fonction est la suivante :
```
nom_fonction <- function(arg1[=expr1], arg2[=expr2], …) {
        Blocs d'instruction
}
```

Exemple d'une fonction qui calcule la somme des éléments d'un vecteur :
```
> somme.vecteur <- function(x){
   s<-sum(x) ;  # Somme des éléments de x
   return(s)
 }
```

## 6. Les différents types d'objets

### a) Les vecteurs

Pour construire des vecteurs plusieurs méthodes sont disponibles dont voici les principales :

- Construction par la fonction collecteur **c** :

> x<- c(5.6, 2, 3.4,-4,9.8) # vecteur numérique à 5 éléments

```
> x
[1]  5.6  2.0  3.4 -4.0  9.8
```

- Construction par l'opérateur séquence » : » :

```
> 1:7
[1] 1 2 3 4 5 6 7
```

- Construction par la fonction **seq()** :

```
> seq(1,5,by=0.7)
[1] 1.0 1.7 2.4 3.1 3.8 4.5
> seq(1,5,length=6)
[1] 1.0 1.8 2.6 3.4 4.2 5.0
```

## Question 1 :

Exécutez le code suivant et interprétez ce qui s'affiche :

```
> x<-c(2:4,9:13)
> y<-c("b", "c", "E")
> x[5]
> y[2:3]
> y[c(2,2,3)]
> x[50]
> x[-5]
> x[3]
> x[c(2,2,5 :7)]
> x[6 :1]
> x[-(1 : 4)]
> x [-c(1,4)]
```

## Question 2 :

Créez une fonction permettant de calculer la variance d'un vecteur.

## b) Les matrices

Une matrice peut être créée avec la fonction **`matrix( )`**.
Regarder le help de la fonction **`matrix( )`**.

Exécutez le code suivant et interprétez ce qui s'affiche :

```
> vector=1:10
> matrice1=matrix(vector, ncol=2)
> matrice1
> matrice2=matrix(1:10,nrow=2,byrow=T)
> matrice2
> m=matrix(1:4,nrow=3,ncol=3)
> m
> print(matrice2)
```

## 7. Operations sur les matrices

- Les fonctions **`dim(), ncol(), nrow()`** indiquent les dimensions d'une matrice :
> dim(matrice1)
> ncol(matrice1)
> nrow(matrice1)
- Le produit de deux matrices s'écrit avec l'opérateur **`%*%`**.
> resultat= matrice1 %*% matrice2
- La fonction **`t()`** transpose une matrice
> t (matrice1)
- La fonction **`diag()`** permet d'extraire la diagonale d'une matrice carrée ou de construire une matrice diagonale à partir d'un vecteur.
>diag (resultat)
>diag(c(3,2,4))
- Les fonctions **`rbind()`** et **`cbind()`** permettent de concaténer par ligne ou par colonne des vecteurs ou des matrices.
> vecteur1= c(8,3,2)
>vecteur2=c(23,6,9)
> res=rbind(vecteur1,vecteur2)
> vecteur3=c(2,4)
> cbind(res,vecteur3)
- Pour diagonaliser une matrice carrée, on utilise la fonction **`eigen( )`**, qui fournit comme résultat une liste de deux composantes : **`$values`** qui contient les valeurs propres, **`$vectors`** qui contient les vecteurs propres normés.
> eigen(resultat)

## Question 3

Exécutez le code suivant et interprétez ce qui s'affiche :

---

> resultat[1,]
> resultat[, c(2,2,1)]
> resultat[-1, ]
> resultat [1 :2,-1]
> resultat [resultat>51]
> matrix(vector,nrow=2)
> matrix(vector,nrow=2, byrow=T)

---

## Question 4

1) Construire la matrice Y suivante :

$$Y = \begin{pmatrix} 1 & 2 & 3 & 5 \\ 10 & 12 & 13 & 22 \\ 5 & 9 & 8 & 34 \\ 7 & 1 & 4 & 3 \end{pmatrix}$$

2) Afficher l'élément de Y contenu dans :
- La troisième ligne et deuxième colonne

- La deuxième ligne de Y
- La quatrième colonne de Y
- La matrice obtenu après l'élimination de la première ligne et la deuxième colonne

## Question 5

Ecrire une fonction qui permet la lecture de données dans un fichier suivi d'un graphe.

## 8. Calcul sur la base « airquality »

1) Charger le fichier de données *airquality*.
2) Expliquez les 6 variables.
3) Calculer les statistiques de bases à l'aide de *summary*
4) Calculer séparément la moyenne, la médiane et l'étendu pour la variable Temp à l'aide des commandes appropriées.
5) Calculer la variance et écrire une fonction pour le calcul de l'écart type.
6) Extraire :
   a) la deuxième ligne
   b) la troisième colonne
   c) les lignes 1, 2 et 4 avec une seule commande c( )
   d) les lignes 2 à 6 avec la commande ' : '
e) tout sauf les colonnes 1 et 2
f) toutes les lignes ayant les températures supérieure à 90.