



DOCUMENTATION:

LANGAGE DE SCRIPT

WINDOWS

Utilisation du langage de commande en ligne
et des fichiers de traitement par lots (fichiers BATCH)
pour administrer un système d'exploitation windows

Auteur: Bernard GIACOMONI
Autoentreprise GIACOMONI Bernard

Version	Date
1.0	08/01/2019

Table des matières

I. INTRODUCTION:	4
I.1. OBJET DU DOCUMENT:	4
I.2. HISTORIQUE RAPIDE:	4
II. L'APPLICATION CMD ET SON LANGAGE DE COMMANDE:	5
II.1. ACTIVATION DE L'APPLICATION CMD:	5
II.1.1. PREMIÈRE SOLUTION: LANCER DIRECTEMENT L'EXÉCUTABLE CMD.EXE:	5
II.1.2. DEUXIÈME SOLUTION: UTILISER LA FENÊTRE D'EXÉCUTION DE WINDOWS:	5
II.1.3. LA FENÊTRE CONSOLE:	6
II.2. DESCRIPTION DU LANGAGE DE COMMANDE:	7
II.2.1. FORMAT GÉNÉRAL:	7
II.2.2. NOTION DE REDIRECTION DES FLUX:	9
II.2.2.A. FLUX DE DONNÉES SORTANT D'UNE COMMANDE:	9
II.2.2.B. FLUX DE DONNÉES "ENTRANT" DANS UNE COMMANDE:	9
II.2.2.C. REDIRECTION DU FLUX DE SORTIE DANS UN FICHER TEXTE:	10
II.2.2.D. REDIRECTION DU CONTENU D'UN FICHER TEXTE DANS UN FLUX D'ENTRÉE :	10
II.2.2.E. REDIRECTION DU FLUX DE SORTIE VERS UN FLUX D'ENTRÉE (PIPE-LINE):	10
II.2.2.F. UTILISATION DU "JOKER" DANS LES RECHERCHES DE FICHERS:	12
II.2.2.G. NAVIGATION ENTRE LES RÉPERTOIRES:	13
II.2.3. PRINCIPALES COMMANDES:	14
II.2.3.A. LA COMMANDE DIR:	14
II.2.3.B. LA COMMANDE TYPE:	15
II.2.3.C. LA COMMANDE CD (Change Directory):	16
II.2.3.D. LA COMMANDE COPY:	17
II.2.3.E. LA COMMANDE ECHO:	18
II.2.3.F. LES COMMANDES DEL, ERASE, RM ET RMDIR:	19
II.2.3.G. LA COMMANDE REN OU RENAME:	20
II.2.3.H. LA COMMANDE D'ÉDITION D'UN FICHER:	21
II.2.3.I. LA COMMANDE MD OU MKDIR:	22
II.2.3.J. LA COMMANDE TIME:	23
II.2.3.K. LA COMMANDE DATE:	23
II.2.3.L. LA COMMANDE IPCONFIG:	24
II.2.3.M. LA COMMANDE NETSTAT:	25
II.2.3.N. LA COMMANDE PING:	25
II.2.3.O. LA COMMANDE ARP:	26
III. FICHERS BATCHES SOUS WINDOWS:	27
III.1. PROGRAMMATION DE FICHERS BATCHES SOUS WINDOWS:	27
III.1.1. REMARQUE PRÉLIMINAIRE:	27
III.1.2. LES FICHERS BATCH SOUS WINDOWS:	27
III.1.3. EXEMPLE DE CRÉATION ET D'UTILISATION D'UN FICHER BATCH:	28
III.1.3.A. CRÉATION D'UN FICHER DE TEST:	28
CRÉATION D'UN FICHER BATCH:	28
III.1.3.B. CRÉATION DU CONTENU DU FICHER BATCH:	28
III.1.3.C. EXÉCUTION DU FICHER BATCH:	29
CONCLUSION:	30
III.1.4. UTILISATION DE VARIABLES DANS UN FICHER BATCH:	30
III.1.4.A. GÉNÉRALITÉS:	30
III.1.4.B. DÉCLARATION DE VARIABLES:	30
III.1.4.C. UTILISATION DES VARIABLES:	31
III.1.4.D. OPÉRATIONS ARITHMÉTIQUES AVEC DES VARIABLES:	31

III.1.4.E. INITIALISATION AVEC DES DONNÉES SAISIES EN LIGNE:.....	32
III.1.4.F. UTILISATION DES VARIABLES D'ENVIRONNEMENT:.....	32
REMARQUE: Les variables peuvent également être utilisées dans des commandes en ligne.....	32
III.1.4.G. DÉFINITION DE VARIABLES LOCALES À UN SCRIPT BATCH:.....	32
III.1.4.H. DÉSACTIVATION D'UNE VARIABLE:.....	33
III.1.4.I. SUBSTITUTION DE CARACTÈRES DANS LE CONTENU D'UNE VARIABLE:..	33
III.1.5. UTILISATION D'ARGUMENTS D'ENTRÉE DANS UN FICHER BATCH:.....	34
III.1.5.A. PASSAGE D'ARGUMENTS A UN FICHER BATCH:.....	34
III.1.5.B. UTILISATION DES ARGUMENTS DANS UN FICHER BATCH:.....	34
EXEMPLE:.....	34
III.1.6. COMMANDES BATCH RÉALISANT DES RUPTURES DE SÉQUENCE:.....	35
III.1.6.A. RUPTURES CONDITIONNELLE (IF):.....	35
III.1.6.B. RUPTURES INCONDITIONNELLES:.....	38
III.1.6.C. LA BOUCLE FOR:.....	39
IV. DÉCLENCHEMENT AUTOMATIQUE DE TÂCHES :.....	41
IV.1. INTRODUCTION :.....	41
IV.2. COMMANDE DE CRÉATION D'UNE TÂCHE:.....	41
IV.2.1. FORMAT:.....	41
IV.2.2. EXEMPLES:.....	42
IV.3. AFFICHAGE DES TÂCHES PLANIFIÉES:.....	42
IV.4. COMMANDE DE SUPPRESSION D'UNE TACHE:.....	42
IV.5. DÉMONSTRATIONS:.....	42
V. ANNEXES:.....	43
V.1. EXEMPLES PLUS COMPLEXES DE FICHIERS BATCHES:.....	43
V.1.1. Calculatrice effectuant les 4 opérations de base (+, -, *, /):.....	43
V.1.2. Sauvegarde périodique d'un répertoire:.....	44
V.1.3. Procédure permettant d'afficher des informations réseau:.....	45
V.2. COMMANDES DE L'INTERPRÉTEUR CMD:.....	46
V.2.1. LISTE DES COMMANDES:.....	46
V.2.2. AIDE SUR LES COMMANDE:.....	49
V.3. PRINCIPALES VARIABLES D'ENVIRONNEMENT:.....	50

I.INTRODUCTION:

I.1.OBJET DU DOCUMENT:

Cette documentation a pour but d'exposer un ensemble de connaissances et de pratiques nécessaires à l'administration des systèmes informatiques basés sur WINDOWS. Plus spécialement, il traite de l'utilisation des COMMANDES EN LIGNE et des FICHIERS DE TRAITEMENT PAR LOTS (Fichiers BATCHES, encore appelés "scripts batches").

I.2.HISTORIQUE RAPIDE:

Les premiers ordinateurs personnels (Personal Computers ou PC) de MICROSOFT étaient par défaut équipés d'un système d'exploitation appelé MS-DOS (MicroSoft Disk Operating System). Ce système qui n'offrait en configuration de base qu'une interface en lignes de commandes a été jusqu'à la fin des années 1980 le plus utilisé sur les machines compatibles PC.

Le système MS-DOS était MONOTÂCHE (pas d'exécution de tâches en parallèle) et MONO-UTILISATEUR (un seul utilisateur avait accès à toutes les ressources de la machine). Il ignorait la notion de FENÊTRAGE de l'écran: ceci revient à dire que l'écran tout entier constituait la seule fenêtre accessible.

Durant cette période, les principaux rivaux des systèmes Microsoft étaient les systèmes à base d'OS UNIX ou compatibles qui, eux aussi, n'offraient en configuration de base que des interfaces en ligne de commande (les SHELLS unix). Cependant, UNIX intégrait le concept de multi-fenêtrage (les "terminaux" Unix) et pouvait gérer simultanément plusieurs utilisateurs.

A la fin des années 1980, la réussite commerciale des systèmes McIntosh (devenus depuis Apple) équipés d'interfaces graphiques basés sur le multi-fenêtrage et les menus en cascades a poussé Microsoft à équiper ses système d'une interface graphique. De ce fait, le système MS-DOS a été abandonné au profit du système WINDOWS.

Cependant, pour des raisons de compatibilité, le nouveau système a intégré l'application CMD.EXE dont le rôle est de SIMULER le fonctionnement de l'interface en ligne de MS-DOS dans une fenêtre WINDOWS de type "console" (fenêtre alphanumérique gérant des interactions en lignes de commandes). L'application CMD.exe est donc un INTERPRÉTEUR DE COMMANDES MS-DOS.

REMARQUE : Il est incorrect d'appeler les fenêtres CMD "fenêtres MS-DOS", comme on l'entend souvent, suggérant par là que WINDOWS englobe toujours un "noyau" MS-DOS: en fait, les systèmes WINDOWS s'appuient sur les noyaux WIN** (WIN32, puis WIN64). Le noyau MS-DOS est, lui, simplement simulé.

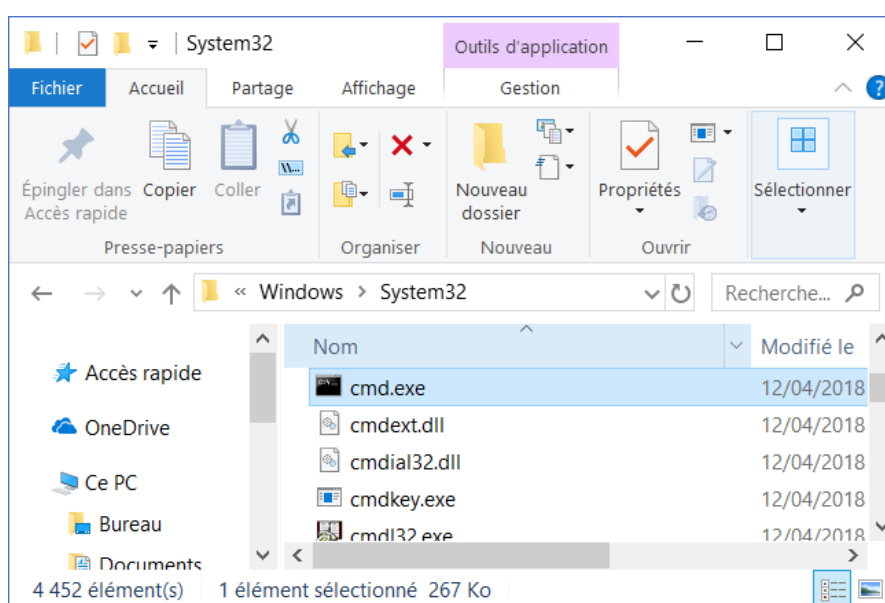
II.L'APPLICATION CMD ET SON LANGAGE DE COMMANDE:

II.1.ACTIVATION DE L'APPLICATION CMD:

Suivant la version de windows utilisée, il existe plusieurs solutions pour ouvrir une fenêtre CMD. Les deux solutions indiquées ci-dessous sont disponibles pour toutes les versions.

II.1.1.PREMIÈRE SOLUTION: LANCER DIRECTEMENT L'EXÉCUTABLE CMD.EXE:

Le fichier exécutable de cette application se trouve dans le sous-répertoire system32 du répertoire c:\windows:



Il suffit alors de faire un double clic sur cmd.exe pour ouvrir une fenêtre console.

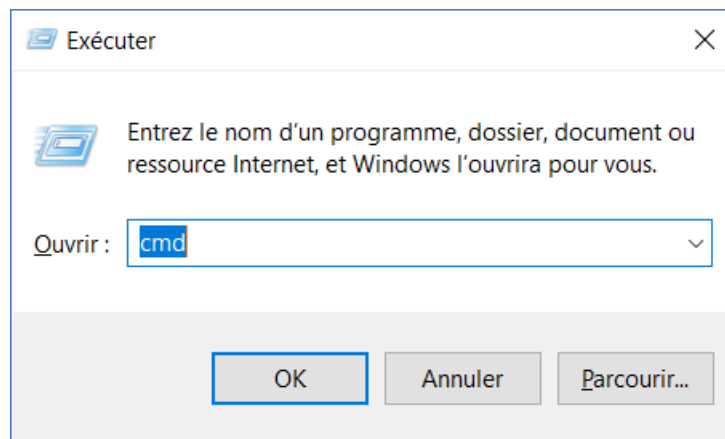
REMARQUES:

1. Il est possible en faisant un clic droit sur cmd.exe et en choisissant l'option "Exécuter en tant qu'administrateur", d'ouvrir une console avec les privilèges d'un administrateurs. Cette option est parfois nécessaire car certaines commandes ne peuvent être exécutées qu'en mode administrateur;
2. Lorsqu'une commande qui exige le mode administrateur est lancée dans une console non ouverte en mode administrateur, le message "L'opération demandée nécessite une élévation" est retourné. Il faut alors rouvrir la fenêtre console en mode administrateur comme il est indiqué à l'alinéa précédent.

II.1.2.DEUXIÈME SOLUTION: UTILISER LA FENÊTRE D'EXÉCUTION DE WINDOWS:

Dans un système WINDOWS, l'application CMD.exe peut être lancée à partir de la

fenêtre "exécuter" qui peut être ouverte en cliquant (clic droit) sur l'icône située dans le coin inférieur gauche de l'écran (icône de démarrage), puis en activant l'option "exécuter" du menu qui s'est ouvert. Son activation provoque l'affichage de la fenêtre suivante:



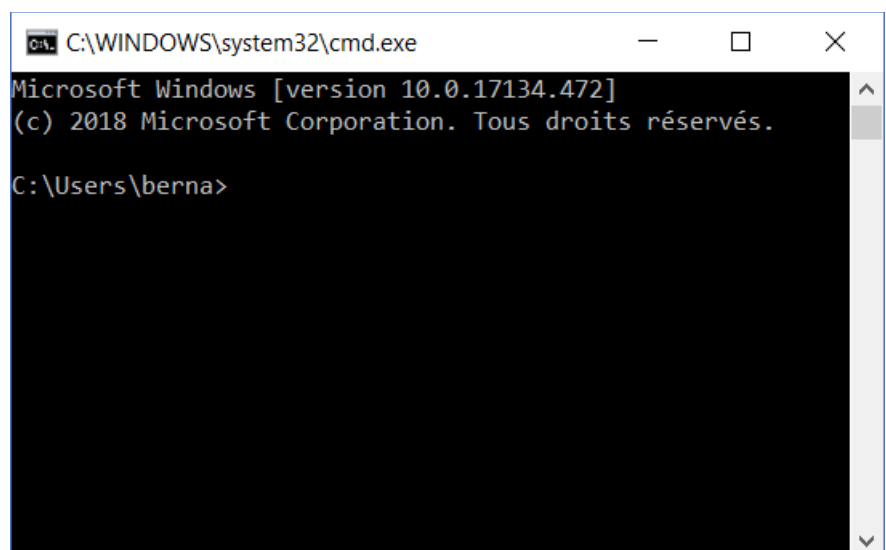
La saisie du nom d'application "cmd" déclenche alors l'ouverture d'une fenêtre dite "console" dans laquelle les commandes en ligne peuvent être saisies.

II.1.3.LA FENÊTRE CONSOLE:

REMARQUE :

**Si la console est lancée depuis la fenêtre d'exécution, l'invite de commande (appelée aussi "prompt") est par défaut constitué du chemin d'accès au répertoire courant : ici, il s'agit du répertoire racine de l'utilisateur "berna". Pour un autre utilisateur, le prompt serait différent.*

**Si la console est lancée depuis c:\windows\system32, le prompt d'ouverture sera:
C:\windows\system32.*



Cette "console" fonctionne donc comme l'écran d'une machine munie d'un système MS-DOS: en particulier l'interpréteur accepte toutes les commandes MS-DOS et répond de la même manière que ce système.

WINDOWS étant un système multi-fenêtré, plusieurs consoles peuvent être ouvertes simultanément.

II.2.DESCRPTION DU LANGAGE DE COMMANDE:

II.2.1.FORMAT GÉNÉRAL:

Le format général des commandes peut être décrit comme suit:

`<nom de commande> [arguments] /<attribut 1> [parametres] ... /<attribut n> [paramètre]>`

- Le nom de commande correspond au nom d'un fichier exécutable (extension .exe ou .bat) appartenant à l'un des répertoires dont les chemins d'accès sont indiqués par la variable d'environnement %PATH%;
- La partie "arguments" dépend de la commande. Il peut s'agir, par exemple, du nom et du chemin d'accès du fichier sur lequel la commande doit agir;
- Les ensembles attribut-paramètre permettent de paramétrer les informations de sortie de la fonction (par exemple, préciser le contenu ou le format d'affichage de ces sorties). Certains attributs peuvent avoir un paramètre associé, d'autres n'en ont pas.

REMARQUES:

- Toutes les commandes de l'ancien O.S. MS-DOS sont acceptées par l'interpréteur CMD.
- La commande `> help <nom de la commande>` permet en général d'afficher une description détaillée de la commande et de son emploi. Cependant, pour certaines commandes particulières (commandes réseau, par exemple), une aide s'obtient par `><nom de commande> /?`;
- La touche clavier F3 rappelle la dernière commande saisie.

EXEMPLE:

Une des commandes CMD les plus utilisées est la commande "dir". Celle-ci permet de lister les fichiers et sous-répertoires d'un répertoire donné. La syntaxe particulière de la commande dir est la suivante:

`dir [< arguments >] <liste d'attributs>`

- La partie <arguments> est utilisée pour préciser le répertoire que l'on veut lister (par exemple: `c:\Users\berna\TestCMD` permet de lister les éléments du répertoire TestCMD);
- Cette commande peut être dotée d'un très grand nombre d'attributs. Ils permettent de sélectionner les éléments qui doivent être listés ou de choisir la présentation ou les informations accompagnant chaque fichier.

Les deux exemples ci-dessous illustrent l'effet de certains des attributs:

La commande

`>dir c:\Users\berna\TestCMD`
 provoque l'affichage du contenu
 du répertoire
`c:\users\berna\TestCMD`

En l'absence d'attributs, chaque
 élément est accompagné de la
 date et de l'heure de sa création
 et de sa taille en octets

```

Sélection C:\WINDOWS\system32\cmd.exe
C:\Users\berna>dir c:\Users\berna\TestCMD
Le volume dans le lecteur C s'appelle Acer
Le numéro de série du volume est E8B4-0DFE

Répertoire de c:\Users\berna\TestCMD

06/01/2019  15:41    <DIR>          .
06/01/2019  15:41    <DIR>          ..
06/01/2019  16:02                440 AfficheDatesSemaine.bat
29/12/2018  18:47                252 AfficheFichierSemaine.bat
23/12/2018  15:44                2 130 AfficheTablesBDD.bat
10/12/2018  18:48                 3 auteurs_livres
24/11/2018  17:54                7 639 bibliotheque.sql
10/12/2018  15:09                1 100 InfosReseau.bat
04/12/2018  18:06                313 Jointure_3_Tables.bat
05/01/2019  18:06                 20 liste.txt
06/01/2019  15:44                 57 semaine.txt
23/12/2018  15:44                 65 Warning
                10 fichier(s)          12 019 octets
                2 Rép(s)   398 953 615 360 octets libres

C:\Users\berna>
    
```

La commande

`>dir c:\Users\berna\TestCMD /b`
 provoque aussi l'affichage du
 contenu du répertoire
`c:\users\berna\TestCMD`

Cependant, l'attribut `/b` permet
 de supprimer des sorties tout ce
 qui n'est pas le nom des fichiers.

```

C:\WINDOWS\system32\cmd.exe
C:\Users\berna>dir c:\Users\berna\TestCMD /b
AfficheDatesSemaine.bat
AfficheFichierSemaine.bat
AfficheTablesBDD.bat
auteurs_livres
bibliotheque.sql
InfosReseau.bat
Jointure_3_Tables.bat
liste.txt
semaine.txt
Warning

C:\Users\berna>
    
```

Enfin, la commande `dir` sans
 préciser le répertoire à lister
 permet de lister le répertoire
 courant (ici, c'est le répertoire
`c:\Users`):

```

C:\WINDOWS\system32\cmd.exe
C:\Users>dir
Le volume dans le lecteur C s'appelle Acer
Le numéro de série du volume est E8B4-0DFE

Répertoire de C:\Users

29/09/2018  06:43    <DIR>          .
29/09/2018  06:43    <DIR>          ..
28/12/2018  21:51    <DIR>          berna
06/10/2017  16:08    <DIR>          defaultuser0
06/06/2018  20:23    <DIR>          Public
                0 fichier(s)          0 octets
                5 Rép(s)   398 951 133 184 octets libres

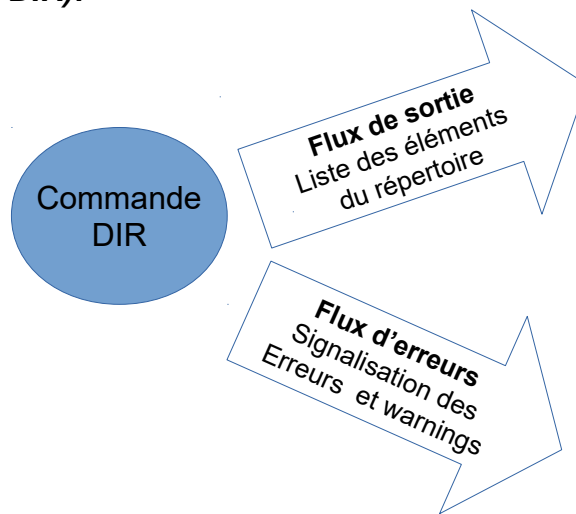
C:\Users>
    
```


II.2.2. NOTION DE REDIRECTION DES FLUX:

II.2.2.A. FLUX DE DONNÉES SORTANT D'UNE COMMANDE:

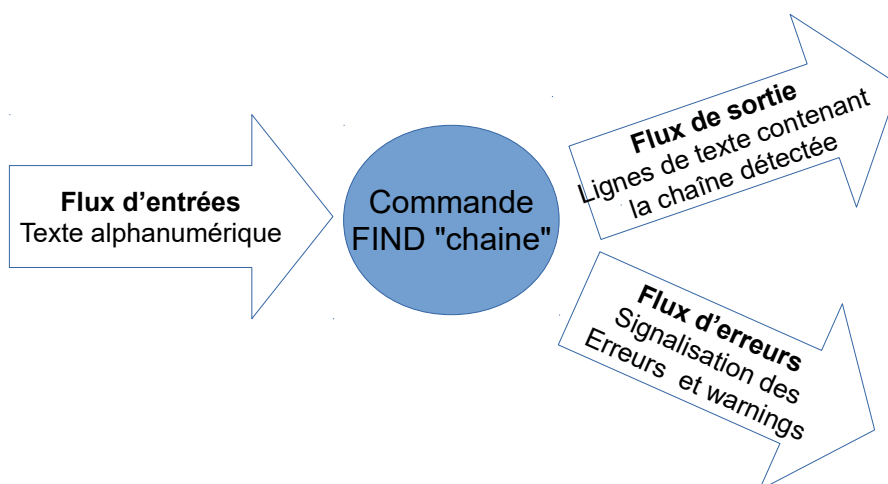
En s'exécutant, une commande produit des données textuelles qui constituent le FLUX DE SORTIE de cette commande. Par défaut, ces données sont affichées dans la fenêtre d'exécution. La commande produit également un FLUX D'ERREURS, composé des divers messages d'avertissement ou d'erreurs qui peuvent survenir en cours d'exécution. Par défaut, ce flux d'erreurs s'affiche également dans la fenêtre d'exécution.

EXEMPLE (Commande DIR):



II.2.2.B. FLUX DE DONNÉES "ENTRANT" DANS UNE COMMANDE:

D'autre part, certaines commandes consomment des données "en entrée". C'est le cas, par exemple, de la commande FIND qui permet de détecter dans son flux d'entrée des chaînes de caractères alphanumériques :



Ces flux de données entrants, qu'il ne faut pas confondre avec les arguments ou paramètres des commandes, peuvent être constitués soit de données contenues dans des fichiers, soit de flux de sorties d'autres commandes.

REMARQUE : ces flux sont souvent appelés "stdin" (entrée), "stdout" (sortie) et "stderr" (erreurs).

II.2.2.C. REDIRECTION DU FLUX DE SORTIE DANS UN FICHIER TEXTE:

Par défaut, une commande affiche ses sorties textuelles dans la fenêtre de commande. Il est possible de rediriger ce flux dans un fichier de texte grâce aux opérateurs ">" et ">>".

Exemples:

- `dir /b > text.txt` provoque la redirection de la liste des éléments contenus dans le répertoire courant vers le fichier text.txt. Si le fichier n'existe pas, il est créé. S'il existe, son contenu antérieur est ÉCRASÉ par la sortie de dir;
- `dir c:\Users\berna /b > text.txt` provoque la redirection de la liste des éléments contenus dans le répertoire `c:\Users\berna` vers le fichier text.txt. Si le fichier n'existe pas, il est créé. S'il existe, son contenu antérieur est ÉCRASÉ par la sortie de dir;
- `dir /b >> text.txt` provoque également la redirection de la liste des éléments contenus dans le répertoire courant vers le fichier text.txt. Si le fichier n'existe pas, il est créé et chargé par la sortie de dir. S'il existe, la sortie de dir est AJOUTÉE au contenu antérieur.

Remarque : La redirection peut être effectuée vers un fichier n'appartenant pas au lecteur courant. Ainsi, la ligne de commande `dir > f:\text.txt` crée le fichier text.txt sur le lecteur f (en général, c'est une clef usb) et y enregistre le résultat de la commande dir.

II.2.2.D. REDIRECTION DU CONTENU D'UN FICHIER TEXTE DANS UN FLUX D'ENTRÉE :

Il est possible de diriger le contenu d'un fichier texte dans le flux d'entrée d'une commande, grâce à l'opérateurs "<". Par exemple, la ligne de commandes :

```
find "abc" < fichier.txt
```

permet d'injecter le texte contenu dans le fichier fichier.txt dans le flux d'entrée de la commande `find "abc"` : cette commande va alors filtrer les lignes de texte contenues dans ce fichier pour n'afficher dans son flux de sortie que celles qui contiennent la chaîne "abc".

Remarque : La redirection peut être effectuée à partir d'un fichier qui n'appartient pas au lecteur courant. Ainsi, la ligne de commande `find ".bat" < f:\fichier.txt` affiche les lignes du fichier \fichier.txt du lecteur f qui incluent la chaîne ".bat".

II.2.2.E. REDIRECTION DU FLUX DE SORTIE VERS UN FLUX D'ENTRÉE (PIPE-LINE):

Le flux de sortie d'une commande peut également être redirigé vers le flux d'entrée d'une autre commande. L'opérateur "|" (appelé pipe-line ou tube) permet de réaliser cette

opération. Par exemple, la ligne de commandes :

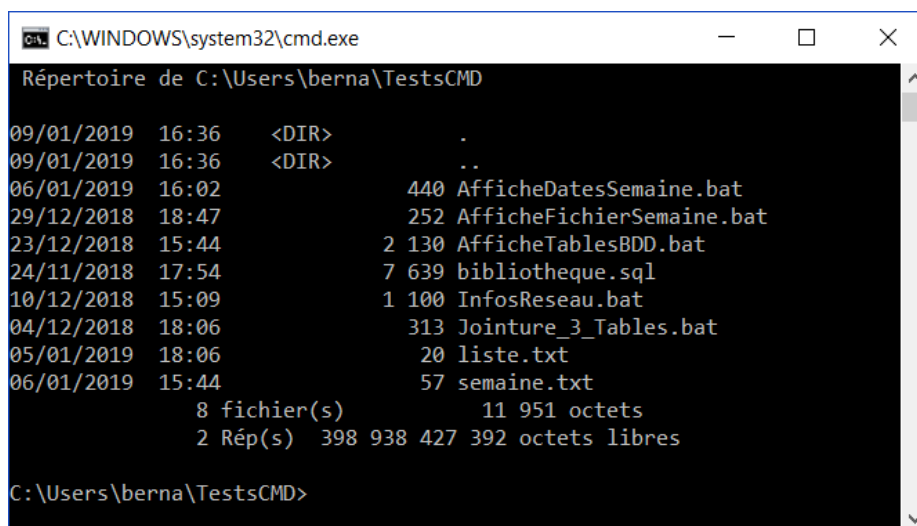
```
<commande 1> | <commande 2> | <commande 3> .....
```

Fonctionne comme suit :

- La sortie de la commande n°1 est injectée dans l'entrée de la commande n° 2;
- La sortie de la commande n°2 est injectée dans l'entrée de la commande n°3;
- Et ainsi de suite.

EXEMPLES:

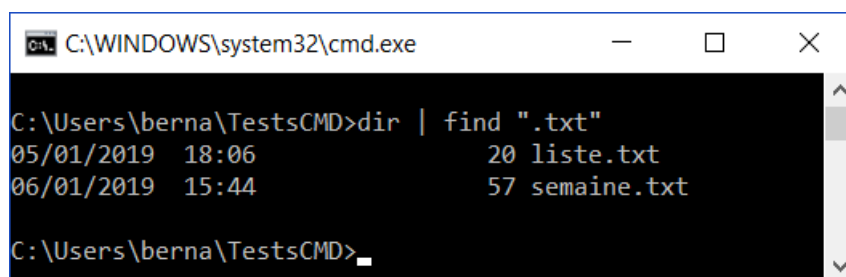
La commande "find" accepte en entrée des données constituées de lignes de textes. Elle agit comme un filtre sur ces données en permettant de n'envoyer dans son flux de sortie que les lignes possédant certaines caractéristiques. Par exemple, supposons que le répertoire C:\Users\berna\TestsCMD ait le contenu suivant:



```
C:\WINDOWS\system32\cmd.exe
Répertoire de C:\Users\berna\TestsCMD
09/01/2019  16:36    <DIR>          .
09/01/2019  16:36    <DIR>          ..
06/01/2019  16:02                440 AfficheDatesSemaine.bat
29/12/2018  18:47                252 AfficheFichierSemaine.bat
23/12/2018  15:44                 2 130 AfficheTablesBDD.bat
24/11/2018  17:54                 7 639 bibliotheque.sql
10/12/2018  15:09                 1 100 InfosReseau.bat
04/12/2018  18:06                313 Jointure_3_Tables.bat
05/01/2019  18:06                 20 liste.txt
06/01/2019  15:44                 57 semaine.txt
            8 fichier(s)          11 951 octets
            2 Rép(s)  398 938 427 392 octets libres

C:\Users\berna\TestsCMD>
```

la commande: `dir | find ".txt"` ne répercutera dans son flux de sortie que les lignes contenant la chaîne ".txt":



```
C:\WINDOWS\system32\cmd.exe
C:\Users\berna\TestsCMD>dir | find ".txt"
05/01/2019  18:06                 20 liste.txt
06/01/2019  15:44                 57 semaine.txt

C:\Users\berna\TestsCMD>
```

Les pipe-lines peuvent être enchaînés: supposons que nous voulions sélectionner les fichiers dont l'extension est ".bat" et qui contiennent la chaîne "Affiche". Nous utiliserons la commande: `dir | find ".bat" | find "Affiche"`:

```

C:\WINDOWS\system32\cmd.exe
C:\Users\berna\TestsCMD>dir | find ".bat" | find "Affiche"
06/01/2019  16:02                440 AfficheDatesSemaine.bat
29/12/2018  18:47                252 AfficheFichierSemaine.bat
23/12/2018  15:44                 2 130 AfficheTablesBDD.bat

C:\Users\berna\TestsCMD>_

```

II.2.2.F.UTILISATION DU "JOKER" DANS LES RECHERCHES DE FICHIERS:

Le joker est le caractère "*" (astérisque). Dans le cadre d'une recherche ou d'une sélection, celui-ci permet de masquer une partie de la chaîne de sélection que l'on ne veut pas préciser. Par exemple:

- > dir A* permet de n'afficher que les fichiers ou sous-répertoires du répertoire courant dont les noms contiennent le caractère A en début;
- > dir *.bat permet de n'afficher que les fichiers du répertoire courant dont les noms ont pour extension ".bat".

EXEMPLE 1: La commande > dir *.txt n'affiche que les éléments du répertoire courant possédant l'extension ".txt":

```

cmd.exe - Raccourci
c:\Users\berna\TestsCMD>dir *.txt
Le volume dans le lecteur C s'appelle Acer
Le numéro de série du volume est E8B4-0DFE

Répertoire de c:\Users\berna\TestsCMD

10/01/2019  17:37                857 Fichier.txt
05/01/2019  18:06                 20 liste.txt
06/01/2019  15:44                 57 semaine.txt
                3 fichier(s)                934 octets
                0 Rép(s) 398 987 300 864 octets libres

c:\Users\berna\TestsCMD>

```

EXEMPLE 2: La commande > dir | find "2018" n'affiche que les éléments du répertoire courant créés en 2018:

```

cmd.exe - Raccourci
c:\Users\berna\TestsCMD>dir | find "2018"
29/12/2018  18:47                252 AfficheFichierSemaine.bat
23/12/2018  15:44                 2 130 AfficheTablesBDD.bat
24/11/2018  17:54                 7 639 bibliotheque.sql
10/12/2018  15:09                 1 100 InfosReseau.bat
04/12/2018  18:06                 313 Jointure_3_Tables.bat

c:\Users\berna\TestsCMD>

```

II.2.2.G.NAVIGATION ENTRE LES RÉPERTOIRES:

Les opérateurs "." et ".." permettent de naviguer dans l'arborescence des répertoires à partir du répertoire courant de la console:

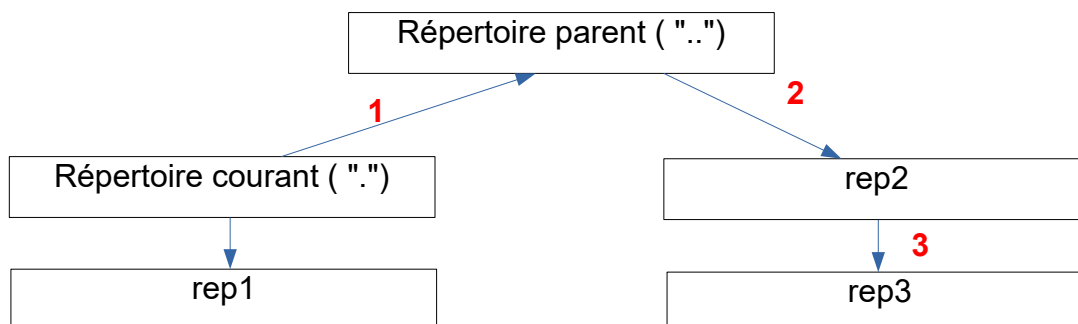
- Le caractère "." dans un chemin d'accès désigne le répertoire courant;
- Le double caractère ".." dans un chemin d'accès désigne le répertoire parent du répertoire courant.

EXEMPLES:

- > dir . Signifie "lister le répertoire courant";
- > dir ./rep1 Signifie "lister le sous-répertoire rep1 du répertoire courant";
- > dir ../rep2/rep3 Signifie "lister le sous-répertoire rep3 du sous-répertoire rep2 du répertoire parent du répertoire courant.

Dans le chemin d'accès ../rep2/rep3:

1. "../" permet de monter du répertoire courant vers le répertoire parent;
2. "rep2/" permet de redescendre du répertoire parent vers le répertoire rep2;
3. "rep3/" permet de descendre du répertoire rep2 vers le répertoire rep3.



II.2.3.PRINCIPALES COMMANDES:

Il n'est pas question ici de décrire l'ensemble des commandes disponibles. Nous nous contenterons de décrire succinctement la vingtaine de commandes qu'il est indispensable de connaître. Pour le reste, nous invitons le lecteur à consulter les annexes de ce document et l'aide en ligne.

II.2.3.A.LA COMMANDE DIR:**FONCTION:**

Elle permet d'afficher à l'écran la liste des fichiers et sous-répertoires d'un répertoire donné.

FORMAT:

`dir [[lecteur:][chemin][nom_de_fichier]] <Liste d'attributs>`

EXEMPLES:

Commande	Effet
<code>dir</code> (sans attribut ni argument)	déclenche l'affichage de la liste des éléments du répertoire courant avec date et volume en octets.
<code>dir /p</code>	permet d'afficher la même chose, mais "page par page" avec une pause entre les pages.
<code>dir /b</code>	permet d'afficher uniquement les noms et extensions des éléments.
<code>dir c:\windows*.exe</code>	déclenche l'affichage de la liste des fichiers exécutables du répertoire windows (fichiers dont l'extension est ".exe") .
<code>dir ../../*.bat</code>	déclenche l'affichage de la liste des fichiers ayant l'extension ".bat" du répertoire grand parent du répertoire courant.
<code>dir . find "2019"</code>	Déclenche l'affichage des fichiers ou sous-répertoires du répertoire courant (paramètre ".") qui ont été créés en 2019.

II.2.3.B.LA COMMANDE TYPE:**FONCTION:**

Elle permet d'afficher le contenu d'un fichier de texte.

FORMAT:

type [lecteur:][chemin d'accès]nom_de_fichier

EXEMPLES:

Commande	Effet
type MonFichierTexte.txt	Affiche le contenu du fichier MonFichierTexte.txt du répertoire courant dans le flux de sortie courant (écran)
type c:\Users\berna\TestCMD\MonFichierTexte.txt	Affiche le contenu du fichier MonFichierTexte.txt du répertoire c:\Users\berna\TestCMD\ dans le flux de sortie courant (écran)
type MonFichierTexte.txt more	permet d'afficher le fichier "page par page" avec une pause entre les pages.

II.2.3.C.LA COMMANDE CD (Change Directory):**FONCTION:**

Cette commande permet de "naviguer" entre les répertoires (changer de répertoire courant, soit directement, soit en remontant ou descendant l'arborescence).

FORMAT:

`cd [/d] <nom + chemin d'accès du nouveau répertoire courant>`

EXEMPLES:

Commande	Effet
<code>cd c:\windows\system32</code>	Le répertoire courant devient c:\windows\system32.
<code>cd ..</code>	Le nouveau répertoire courant devient le parent de l'ancien répertoire courant (on remonte l'arborescence)
<code>cd .\rep1</code> ou <code>cd rep1</code>	Le nouveau répertoire courant devient le répertoire rep1, enfant de l'ancien répertoire courant (on descend l'arborescence).
<code>cd \</code>	Le nouveau répertoire courant devient la racine du lecteur en cours (si le lecteur est c:, le répertoire devient c:\).

II.2.3.D.LA COMMANDE COPY:**FONCTION:**

Permet de copier un fichier d'un répertoire dans le même répertoire ou dans un autre répertoire en changeant son nom et son extension. Attention: il faut que le répertoire cible EXISTE, sinon, il faut le créer au préalable.

FORMAT:

copy [lecteur:][chemin d'accès]<nom.ext> [lecteur:][chemin d'accès][<nom.ext>]

EXEMPLES:

Commande	Effet
copy fichier1.ext fichier2.ext	Fait une copie de fichier1.ext du répertoire courant dans fichier2.ext du répertoire courant.
copy c:\rep1\fic1.txt ..\copyFic1.txt	Fait une copie du fichier fic1.txt du répertoire c:\rep1\fic1.txt vers le fichier copyFic1.txt du répertoire parent du répertoire courant.
copy c:\windows\system32*.dll .\copyDll	Copie tous les fichiers d'extension .dll du répertoire c:\windows dans le sous-répertoire copyDll du répertoire courant. Les fichiers copiés ne changent pas de nom.
copy c:\windows\system32*.dll .\copyDll\copie_*.dll	Copie tous les fichiers d'extension .dll du répertoire c:\windows dans le sous-répertoire copyDll du répertoire courant. Les fichiers copiés prennent pour noms les noms des fichiers d'origine préfixés par "copie_".

II.2.3.E.LA COMMANDE ECHO:

FONCTION:

La commande echo permet:

- D'afficher des messages textuels;
- D'activer ou désactiver l'affichage des commandes.

FORMAT:

> echo [on | off]

> echo [message]

EXEMPLES:

Commande	Effet
echo "bonjour"	Affiche sur l'écran le message "bonjour" (utile dans les scripts)
echo %var%	Affiche sur l'écran le contenu de la variable d'environnement var. (exemple: echo %path% affiche tous les chemins d'accès déclarés)
echo "La date courante est:%date%"	Affiche: "La date courante est: 10/01/2019"
echo off	Stoppe l'affichage des infos de sortie des commandes (utile dans les scripts).
echo on	Rétablit l'affichage des infos de sortie des commandes (utile dans les scripts).

Voir en annexe la liste des variables systèmes.

II.2.3.F.LES COMMANDES DEL, ERASE, RM ET RMDIR:**FONCTION:**

DEL ou ERASE permettent de supprimer un ou des fichiers

RM et RMDIR permettent de supprimer un ou des répertoires

FORMAT:

del <Liste d'attributs> <liste de fichiers>

erase <Liste d'attributs> <liste de fichiers>

rmdir <Liste d'attributs> <liste de répertoires>

<liste de fichiers ou répertoires>: Liste de fichiers ou répertoires avec noms et chemins d'accès.

EXEMPLES:

Commande	Effet
del fichier.ext	Supprime le fichier fichier.ext du répertoire courant
del c:\User\berna\TestCMD\Fichier.txt	Supprime le fichier Fichier.txt du répertoire c:\User\berna\TestCMD\
del .\COPY*.txt	Supprime tous les fichier du sous-répertoire COPY du répertoire courant qui ont l'extension ".txt".
del .\Dir1*.exe .\Dir2*.bat	Supprime tous les fichier du sous-répertoire Dir1 qui ont l'extension .exe et tous les fichier du sous-répertoire Dir2 qui ont l'extension .bat.
rmdir dir1	Supprime le sous-répertoire dir 1 du répertoire courant.

II.2.3.G.LA COMMANDE REN OU RENAME:**FONCTION:**

Permet de changer le nom d'un fichier ou d'un répertoire

FORMAT:

> ren <nom fichier A+chemin d'accès> <Nouveau nom du fichier A>

EXEMPLES:

Commande	Effet
ren ..\liste.txt maliste.text	Change le nom du fichier liste.txt du répertoire parent du répertoire local en "maliste.text"
ren Affiche*.bat Affo*.bat	Les noms de tous les fichiers .bat contenant la chaîne "Affiche" en début sont changés: "Affiche" devient "Affoche".

II.2.3.H.LA COMMANDE D'ÉDITION D'UN FICHIER:

FONCTION:

Permet d'éditer un fichier de texte avec l'éditeur bloc note de windows.

FORMAT:

> notepad <nom fichier A+chemin d'accès>

EXEMPLES:

Commande	Effet
notepad c:\Users\berna\TestCMD\Fichier.txt	Permet d'éditer le fichier Fichier.txt du répertoire c:\Users\berna\TestCMD
notepad Fichier.txt	Permet d'éditer le fichier Fichier.txt du répertoire courant

II.2.3.I.LA COMMANDE MD OU MKDIR:

FONCTION:

Permet de créer un répertoire

FORMAT:

md <nom et chemin d'accès du répertoire à créer>

mkdir <nom et chemin d'accès du répertoire à créer>

EXEMPLES:

Commande	Effet
md c:\Users\UserXX	Permet de créer un sous-répertoire UserXX dans le répertoire c:\Users

II.2.3.J.LA COMMANDE TIME:**FONCTION:**

Permet d'afficher et de modifier l'heure courante

FORMAT:

> time [/T] [<nouvelle heure>]

EXEMPLES:

Commande	Effet
time (sans paramètre)	Permet d'afficher l'heure courante. La réponse est: L'heure actuelle est: 18:21:53,53 Entrez la nouvelle heure: Entrer la nouvelle heure ou faire ctrl/c
time /T	N'affiche que les heures et minutes
time 18:22:54	Change directement l'heure courante en la nouvelle heure.

II.2.3.K.LA COMMANDE DATE:**FONCTION:**

Permet d'afficher et de modifier la date courante

FORMAT:

> date [<nouvelle date>]

EXEMPLES:

Commande	Effet
date (sans paramètre)	Permet d'afficher la date courante. La réponse est: La date du jour est: 10/01/2019 Entrez la nouvelle date: (jj-mm-aa) Entrer la nouvelle date ou faire ctrl/c
date 15-04-20	Change directement la date courante en la nouvelle date (ici, 15 avril 2020).

II.2.3.L.LA COMMANDE IPCONFIG

FONCTION:

La commande ipconfig permet d'afficher toutes les valeurs de la configuration réseau TCP/IP en cours. Elle permet également d'actualiser les paramètres DHCP (Dynamic Host Configuration Protocol) et le nom de domaine (DNS).

FORMAT:

ipconfig [/allcompartments] [/? | <Attributs>

REMARQUE: les interfaces réseaux d'une machine peuvent être isolés les uns des autres dans des "compartiments". Cette répartition en compartiments a pour but d'empêcher le routage d'un compartiment vers un autre.

EXEMPLES:

Commande	Effet
ipconfig (sans autre paramètre)	Afficher les adresses IP (V4 et V6), le masque de sous-réseau et la passerelle par défaut de chaque carte réseau (réelle ou virtuelle) installée.
ipconfig /all	Affiche l'ensemble des informations disponibles pour l'hôte et les cartes réseau installées: adresses IP V4 et V6, masques de sous-réseaux, adresses physiques (mac), infos DHCP et DNS, types de cartes réseau, etc.
ipconfig /allcompartments	Même chose que /all, mais pour tous les compartiments définis.

REMARQUE: la commande permet également de:

- Relacher ou renouveler une adresse IP (v4 ou V6) d'une carte (/release, /release6, /renew, /renew6);
- Afficher ou modifier les id de classes DHSC (/showclassid, /setclassid, /showclassid6, /setclassid6);
- Visualiser le cache de résolution DNS (/displaydns);
- etc.

II.2.3.M.LA COMMANDE NETSTAT**FONCTION:**

Affiche des informations sur les protocoles et les connexions réseau TCP/IP en cours.

FORMAT:

netstat <attributs-paramètres d'attributs> [période d'affichage en secondes]

EXEMPLES:

Commande	Effet
netstat	Affiche les connexions actives (sockets) et leurs états courants.
netstat -a	Affiche toutes les connexions et tous les ports d'écoute.
netstat -a 5	Affiche toutes les connexions et tous les ports d'écoute toutes les 5 secondes.
netstat -b	Affiche toutes les connexions, tous les ports d'écoute et les applications utilisatrices (exige mode administrateur).

II.2.3.N.LA COMMANDE PING**FONCTION:**

Expédie vers un hôte cible du réseau local une requête ICMP permettant de tester l'existence et l'activité de cet hôte. L'hôte répond en retournant certains renseignements sur son activité.

FORMAT:

ping <attributs-paramètres> <nom ou adresse IP de la cible>

EXEMPLES:

Commande	Effet
ping 192.168.1.21	Envoi vers l'hôte 192.168.1.21 de 4 requêtes ICMP de ping.
ping -n 10 192.168.1.21	Envoi vers l'hôte 192.168.1.21 de 10 requêtes ICMP de ping.
ping -i 23 192.168.1.21	

II.2.3.O.LA COMMANDE ARP:**FONCTION:**

Affiche le "cache ARP" associé aux interfaces réseau (physiques ou virtuels) existant dans la machine.

RAPPEL: Le "cache ARP" d'une machine est la table de correspondance entre les adresses IP du réseau local et les adresses MAC correspondantes. Cette table est mise à jour automatiquement au démarrage de la machine et en cours d'utilisation de celle-ci.

Exemple:

Adresse IP	Adresse MAC (adresse physique)
192.168.1.1	08-3e-5d-da-03-9e
192.168.1.10	c0-d0-44-c9-8a-90
192.168.1.255	ff-ff-ff-ff-ff-ff

FORMATS:

arp -s inet_addr eth_addr [if_addr] (ajout dans un cache)

arp -d inet_addr [if_addr] (

arp -a [inet_addr] [-N if_addr] [-v]

EXEMPLES:

Commande	Effet
arp -a	Affiche l'état courant du cache ARP de la machine.
arp -a 192.168.1.1	Affiche les lignes du cache correspondant à l'interface 192.168.1.1
arp -s 192.168.1.21 08-00-27-aa-ba-4e 192.168.1.12	Ajoute dans le cache la ligne de correspondance IP – Mac: 192.168.1.21 08-00-27-aa-ba-4e
arp -d 192.168.1.21	Supprime dans le cache les lignes correspondant à l'hôte 192.168.1.1

III.FICHIERS BATCHES SOUS WINDOWS:

III.1.PROGRAMMATION DE FICHIERS BATCHES SOUS WINDOWS:

III.1.1.REMARQUE PRÉLIMINAIRE:

Le vocable BATCH peut être traduit par "lot". En informatique, un TRAITEMENT PAR LOTS (batch processing) consiste en l'enchaînement automatique d'une suite de commandes sans intervention d'un opérateur. Dans le cas qui nous occupe, un fichier batch est une liste de commandes systèmes formant un ALGORITHME que le système d'exploitation est capable d'exécuter comme les instruction d'un programme. Les expressions "script batch" (windows) ou "script shell" (unix-linux) sont souvent utilisées.

III.1.2.LES FICHIER BATCH SOUS WINDOWS:

Les fichiers batch sont des fichiers de textes munis de l'extension ".bat" (l'extension ".cmd" est aussi reconnue pour les windows récents). Ces fichiers contiennent des listes de commandes systèmes "en ligne".

Ces fichiers peuvent être exécutés automatiquement:

- Soit en effectuant un "double clic" sur leur nom dans une fenêtre d'affichage graphique, exactement comme pour un fichier ".exe";
- Soit par l'intermédiaire de l'invite de commande CMD. C'est ce dernier cas qui nous intéresse:

Lors de leur exécution, les commandes qu'ils renferment sont exécutées lignes après ligne (lot après lot), sauf si la commande en cours d'exécution déclenche une rupture de séquence (commandes goto, if, for, etc.). Ces commandes sont donc semblables aux instructions "de test" des langages de programmation. Elles permettent l'écriture de véritables algorithmes.

III.1.3.EXEMPLE DE CRÉATION ET D'UTILISATION D'UN FICHER BATCH:**III.1.3.A.CRÉATION D'UN FICHER DE TEST:**

Nous allons d'abord créer un fichier de texte "semaine.txt" contenant les noms des 7 jours de la semaine, écrits sur sept lignes consécutives (avec une majuscule au début de chaque mot):

Fichier semaine.txt :

Lundi

Mardi

Mercredi

Jeudi

Vendredi

Samedi

Dimanche

Nous utiliserons pour cela l'éditeur "bloc note" intégré dans les systèmes WINDOWS et nous créerons ce fichier dans un sous-répertoire TestCMD du répertoire courant;

```
C:\Users\berna\TestCMD> notepad semaine.txt
```

Puis, sauvegardons et fermons le bloc note.

CRÉATION D'UN FICHER BATCH:

Créons alors un fichier batch "AfficheJoursSemaine.bat" vide. Nous pouvons le faire à partir de l'éditeur "bloc note" ou bien utiliser la commande en ligne:

```
>echo "" > AfficheJoursSemaine.bat
```

NOTA: La commande `echo ""` engendre en sortie une chaîne de caractère nulle qui est redirigée dans le fichier `AfficheJoursSemaine.bat`, après création de ce fichier.

III.1.3.B.CRÉATION DU CONTENU DU FICHER BATCH:

Nous utiliserons pour cela l'éditeur Bloc Note. Pour éditer le fichier commande.bat, saisissons la commande:

```
>notepad AfficheJoursSemaine.bat
```

et plaçons dans ce fichier les lignes:

```
echo off
```

```
type semaine.txt.
```

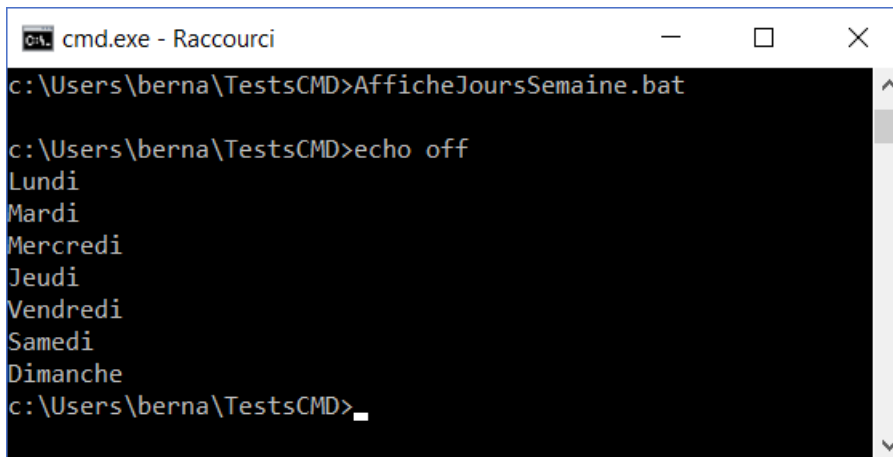
Cette commande a pour effet d'afficher ligne par ligne le contenu du fichier semaine.txt que nous avons créé précédemment. La première ligne a pour effet d'empêcher l'affichage des commandes au fur et à mesure de leur exécution. Sauvegardons ce contenu et fermons le bloc note.

III.1.3.C.EXÉCUTION DU FICHER BATCH:

Pour cela, saisissons dans l'invite de commande le nom du fichier batch:

> AfficheJoursSemaine.bat

Ceci va déclencher l'exécution de AfficheJoursSemaine.bat. Celui-ci va afficher tel quel le contenu du fichier semaine.txt:



```
cmd.exe - Raccourci
c:\Users\berna\TestsCMD>AfficheJoursSemaine.bat
c:\Users\berna\TestsCMD>echo off
Lundi
Mardi
Mercredi
Jeudi
Vendredi
Samedi
Dimanche
c:\Users\berna\TestsCMD>
```

L'exécution du fichier batch a donc le même résultat que l'exécution de la commande qu'il renferme.

Modifions alors la deuxième ligne du fichier AfficheJoursSemaine.bat:

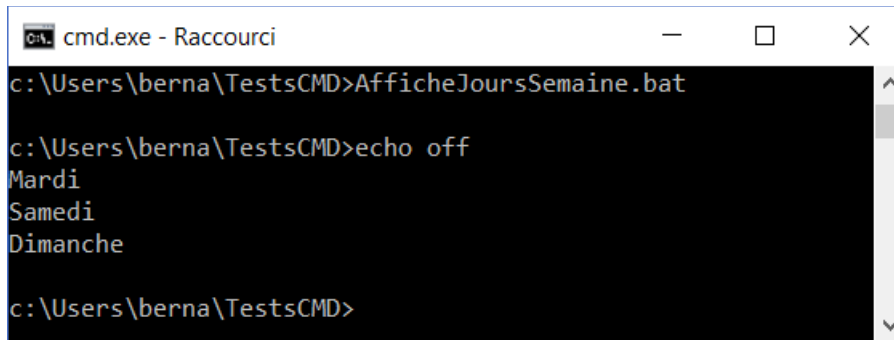
type semaine.txt | find "a"

INDICATIONS:

- La commande type permet d'afficher ligne par ligne le fichier texte qui lui est spécifié en argument (ici: "semaine.txt"). En sortie de type nous aurons donc les noms des sept jours de la semaine;
- Le signe "|" (commande "pipeline") permet d'enchaîner deux commandes, la sortie de la première devenant l'entrée de la seconde. De ce fait, le pipe line va présenter en entrée de la commande find les sorties de la commande type (les sept jours de la semaine);
- La directive "find" agit comme un filtre pour les chaînes de caractères qu'on lui présente en entrée: elle ne laisse passer que les chaînes qui contiennent la sous-chaîne qu'on lui a spécifiée en argument. Ici, elle ne laissera passer que les chaînes contenant la sous-chaine "a".

De ce fait, le résultat de cette commande composée doit être l'affichage des jours de la semaine comprenant un "a".

Lançons alors l'exécution de ce fichier batch: >AfficheJoursSemaine.bat. On obtient:



```
cmd.exe - Raccourci
c:\Users\berna\TestsCMD>AfficheJoursSemaine.bat
c:\Users\berna\TestsCMD>echo off
Mardi
Samedi
Dimanche
c:\Users\berna\TestsCMD>
```

CONCLUSION:

- La saisie de la commande `type semaine.txt | find "a"` est donc remplacée par la saisie du nom du fichier batch renfermant cette commande `>AfficheJoursSemaine.bat`: ceci est déjà avantageux lorsque la commande remplacée est complexe. Cela l'est encore plus lorsque le fichier batch contient une série de commandes beaucoup plus longue.
- De plus, le fichier batch une fois créé peut être réutilisé un nombre de fois illimité, permettant ainsi de créer de véritables "tâches d'administration" utilisables à chaque besoin.

III.1.4.UTILISATION DE VARIABLES DANS UN FICHER BATCH:

III.1.4.A.GÉNÉRALITÉS:

Il est possible de définir des VARIABLES à l'intérieur d'un fichier batch. Comme dans un langage de programmation, ces variables sont des CONTENEURS permettant de stocker des VALEURS (numériques, alphanumériques, etc.).

Ces variables ne doivent pas être confondues avec les VARIABLES D'ENVIRONNEMENT du système d'exploitation telles que `%PATH%`, `%USERNAME%` ou `%TIME%` (bien que les valeurs de celles-ci soient accessibles dans un fichier batch).

III.1.4.B.DÉCLARATION DE VARIABLES:

On utilise la commande SET:

```
> set <nom de variable>=<valeur initiale>
```

Exemple: `> set PSEUDO=Batman` crée la variable PSEUDO et l'initialise à la valeur Batman.

REMARQUES:

- Ne pas insérer d'espace autour du signe =;

- Les caractères saisis à droite du signe = constituent la valeur de la variable.

III.1.4.C.UTILISATION DES VARIABLES:

PRINCIPE:

A l'intérieur d'un script batch, la valeur d'une variable est représentée par son nom entouré de caractères "%": par exemple, la valeur de PSEUDO est représentée par %PSEUDO%.

Exemple: le script:

```
echo off
set a=Hello!
echo %a%
```

a pour résultat l'affichage de: Hello!

Autre exemple: le script:

```
echo off
set a=PI
set b=3.14159
echo %a%=%B%
```

a pour résultat l'affichage de: >PI=3.14159

III.1.4.D.OPÉRATIONS ARITHMÉTIQUES AVEC DES VARIABLES:

Il est possible d'initialiser une variable par une opération arithmétique, avec l'option "/a":

Par exemple:

```
echo off
set a=40
set b=10
set c=2
set /a R=(%a%+%b%)/%c%
echo %R%
```

Donne pour résultat: > 25 soit: $(40+10)/2 = 25$

Ou encore:

```
echo off;
set /a R=(3+5)*7
echo R
```

qui donne: >56

Les possibilités de traitements mathématiques sont assez étendues (voir l'aide: >help set).

III.1.4.E.INITIALISATION AVEC DES DONNÉES SAISIES EN LIGNE:

Il est possible d'initialiser une variable avec des données saisies en ligne en utilisant le commutateur /p:

```
set /p <nom de variable>=<texte d'invite>
```

Exemple: le script:

```
echo off
set /p R=Saisissez un nombre:
echo Vous avez saisi le nombre %R%.
```

S'arrête à la deuxième ligne après avoir affiché >Saisissez un nombre:

Saisir alors, par exemple, le nombre 128. Le script redémarrera pour afficher:

```
>Vous avez saisi le nombre 128.
```

III.1.4.F.UTILISATION DES VARIABLES D'ENVIRONNEMENT:

Celles-ci peuvent être utilisées exactement comme des variables locales aux scripts:

Exemple: affichage de l'heure courante:

```
echo off
set a=L'heure courante est:
echo %a% %TIME%
```

Exploite la valeur de la variable d'environnement %TIME% pour afficher l'heure courante (le résultat sera par exemple : > L'heure courante est: 16:06:30,36).

REMARQUE: Les variables peuvent également être utilisées dans des commandes en ligne.

EXEMPLE:

```
C:\Users\berna\set PSEUDO=Batman
C:\Users\berna\set MDP=batmanforever
C:\Users\berna\echo Connexion: %PSEUDO%- %MDP%
Affiche: C:\Users\berna\Connexion: Batman-batmanforever
```

III.1.4.G.DÉFINITION DE VARIABLES LOCALES À UN SCRIPT BATCH:

Les mots-clefs SETLOCAL et ENDLOCAL permettent de définir des sections à l'intérieur d'un script où les variables définies sont LOCALES à la section.

Sinon, par défaut, les variables sont globales au script (remarque: ENDLOCAL se produit automatiquement en fin de script).

III.1.4.H.DÉSACTIVATION D'UNE VARIABLE:

La commande:

```
set "<variable>=" (ex: set "VAR=")
```

permet de désactiver la variable dont le nom est libellé dans la commande.

III.1.4.I.SUBSTITUTION DE CARACTÈRES DANS LE CONTENU D'UNE VARIABLE:

Nous avons vu que lorsqu'on veut utiliser le contenu d'une variable dans fichier batch, on utilise la notation %<nom de variable>%. Lorsque le contenu de cette variable est textuel, il est possible à cette occasion de substituer facilement une chaîne de caractères par une autre dans la **valeur récupérée** de cette variable en utilisant la notation:

```
%<nom de variable>:<chaîne à substituer>=<chaîne substituée>%
```

NOTA: cette substitution est faite uniquement dans le texte récupéré: le contenu de la variable ne change pas.

EXEMPLE: le script:

```
set VAR=illusion  
echo%VAR:ill=diff%
```

affiche: > **diffusion** (la sous-chaîne "ill" de illusion est remplacée par "diff").

REMARQUE: ce mécanisme permet d'effectuer facilement des substitutions de sous-chaînes d'un chaîne de caractères dans un fichier batch. Par exemple, pour substituer à la sous-chaîne "ill" la sous-chaîne "diff" dans le **contenu** de VAR, il suffit de faire:

```
set VAR=%VAR:ill=diff%
```

III.1.5.UTILISATION D'ARGUMENTS D'ENTRÉE DANS UN FICHER BATCH:

III.1.5.A.PASSAGE D'ARGUMENTS A UN FICHER BATCH:

Jusqu'à maintenant, l'exécution du fichier batch aboutit au même résultat que si on avait saisi directement les commandes dans l'invite MS-DOS. Cependant, les fichiers batch présentent également l'avantage de permettre la saisie d'ARGUMENTS D'ENTRÉE.

Les valeurs de ces arguments d'entrée doivent être saisies à la suite du nom du fichier, dans la commande d'exécution.:

```
> <Nom de commande.bat> arg1 arg2 ... arg n
```

Ces valeurs peuvent être représentées par des nombres, des chaînes de caractères ou des variables.

EXEMPLE:

```
> Calcule.bat 5 + 6
```

Déclenche l'exécution du script calcule.bat avec 3 arguments d'entrée: les valeurs "5", "+" et "6".

III.1.5.B.UTILISATION DES ARGUMENTS DANS UN FICHER BATCH:

Dans le fichier batch, les valeurs de ces arguments sont identifiées par le numéro de l'argument précédé du caractère % (%1, %2, %3, etc.). Rappelons que la valeur d'une **variable** s'obtient en **entourant** son nom de **deux** caractères % (valeur de var =%var%).

EXEMPLE:

Plaçons dans le fichier batch AfficheJoursSemaine.bat la ligne suivante:

```
type semaine.txt | find "%1"
```

Il suffit alors lors de l'activation de la procédure de faire suivre le nom du fichier batch de la chaîne de caractère que l'on veut sélectionner pour obtenir la sélection correspondante:

- > **AfficheJoursSemaine di** permet de n'afficher que les jours qui ont la chaîne "di" dans leur nom (tous sauf Dimanche, dont le D est majuscule);
- > **AfficheJoursSemaine a** permet de n'afficher que les jours qui ont la lettre "a" dans leur nom (Mardi, Samedi, Dimanche);
- **AfficheJoursSemaine L** Permet de n'afficher que les jours qui ont la lettre "L" dans leur nom (Lundi);

III.1.6.COMMANDES BATCH RÉALISANT DES RUPTURES DE SÉQUENCE:**III.1.6.A.RUPTURES CONDITIONNELLE (IF):****FORME GÉNÉRALE:**

if [NOT] <condition> (<commandes>) ELSE (<commandes>)

Il est possible d'adopter une présentation plus lisible:

```
if [NOT] <condition> (
    Commande 1
    -----
    Commande n

) ELSE (
    Commande a
    -----
    Commande x

)
```

EXEMPLE:

La procédure suivante permet d'additionner ou soustraire deux nombres passés en arguments.

Paramètres d'entrée: %1 = premier nombre,%2 = deuxième nombre,%3 = + ou -.

```
echo off
```

```
if %3% EQU + (
    set /a R=%1+%2
) else (
    set /a R=%1-%2
)
echo Resultat=%R%
```

- L'appel de la procédure avec les arguments 3, 4 et + affichera 7 (3+4)
- L'appel de la procédure avec les arguments 3, 4 et - affichera -1 (3-4)

REMARQUE: Il est possible d'imbriquer une structure if dans une autre structure if. Par exemple, le script Compare.bat utilise deux if imbriqués pour comparer deux nombres passés en argument:

```
echo off
if %1 GTR %2 (
    set R=plus grand que
) else (
    if %1 LSS %2 (
        set R=plus petit que
    ) else (
        set R=egal a
    )
)
echo %1 est %R% %2
```

- >Compare.bat 54 57 donne la sortie "54 est plus petit que 57"
- >Compare.bat 109 43 donne la sortie "109 est plus grand que 43"
- >Compare.bat 37 37 donne la sortie "37 est égal à 37"

Les opérateurs de comparaison peuvent être:

Opérateur	Signification	Exemple
EQU	Egalité	If A EQU B: si A et B ont la même valeur
NEQ	Inégalité	If A NEQ B: si A et B n'ont pas la même valeur
LSS	Inférieur à ...	If A LSS B: si A est inférieur à B
LEQ	Inférieur ou égal à ...	If A LEQ B: si A est inférieur ou égal à B
GTR	Plus grand que ...	If A GTR B: si A est plus grand que B
GEQ	Plus grand ou égal à ...	If A GEQ B: si A est supérieur ou égal à B
"=="	Test d'égalité entre deux chaînes	If %A == OK: si la chaîne %A est égale à "OK"
ERRORLEVEL n	Si le code retourné par la dernière exécution est supérieur ou égal à n.	If ERRORLEVEL 3: si le compte-rendu d'exécution est supérieur ou égal à 3.
EXIST <nom fichier>	Si le fichier existe	If EXIST control.bat ...
DEFINED <nom variable>	Si la variable est définie...	If DEFINED PSEUDO ...

Ces opérateurs peuvent être précédés de l'opérateur NOT qui inverse la condition:

- **If NOT EXIST control.bat ...** signifie: Si le fichier control.bat n'existe pas.
- **IF NOT %S==OK** signifie: si la variable %S n'est pas égale à OK

III.1.6.B.RUPTURES INCONDITIONNELLES:

Ce sont les commandes de type

`>goto:<nom d'étiquette>`

ces commandes déroutent l'exécution du script courant vers l'étiquette libellée en argument. Ces étiquettes doivent être positionnées dans le script à l'endroit où l'on veut dérouter l'exécution:

EXEMPLE: cet algorithme affiche un décompte de 10 à 1. Tant que le compteur N est supérieur à 0, goto renvoie l'exécution après l'étiquette :Etiq

Fichier Decompte.bat:

```
echo off
set N=10
:Etiq
if %N% GTR 0 (
    echo %N%
    set /a N=%N%-1
    goto :Etiq
)
```

REMARQUE: ce script définit une structure de boucle de traitement :

REM initialisation de la variable de parcours

```
set N=10
```

REM Etiquette de début de boucle

```
:Etiq
```

REM Condition d'arrêt de la boucle

```
if <condition sur la variable de boucle N> (
```

```
    < Commande 1>
```

```
    -----
```

```
    < Commande n>
```

```
    goto :Etiq
```

```
)
```

Cette structure peut être utilisée en alternative de la boucle FOR que nous allons étudier par la suite.

III.1.6.C.LA BOUCLE FOR:

La commande FOR est la seule commande spécifique permettant de programmer une boucle dans un shell (à moins d'utiliser la commande GOTO et la structure vue au paragraphe précédent). Sa forme générale est:

```
FOR [commutateur] ["delims=<chaîne>"] %%<nom de variable> IN (<ensemble>) DO (
    commande 1 [paramètres]
    -----
    commande n [paramètres]
)
```

Commutateurs	<p>/d: Spécifie que <ensemble> est un ensemble de répertoires. /l: Spécifie que <ensemble> contient 3 nombres: (début,pas et fin) qui définissent les valeurs prises par la variable de parcours. /r: Spécifie que le parcours doit être exécuté récursivement. /f: Le parcours se fait:</p> <ul style="list-style-type: none"> - sur la liste des fichiers spécifiés dans <ensemble> , - sur le flux de sortie de la commande spécifiée par <ensemble> (ex: 'dir') - sur le texte spécifié par <ensemble>(Ex: 'c'est mon texte')
"delims=<chaîne>"	Indique que les fichiers de textes doivent être traités par tronçons délimités par la chaîne spécifiée. Si celle-ci est omise, le texte est traité ligne par ligne.
%%<nom de variable>	<p>Variable de parcours qui va contenir la valeur chaque élément de la recherche (nom de fichier, ligne de texte, etc.). Remarquer le doublage du caractère %. A l'intérieur de la boucle, la valeur de la variable est %%<nom de variable>.</p> <p>Exemple: dans la boucle for %%i in, la valeur de parcours i est accessible par %%i</p>
<ensemble>	<p>Désigne l'ensemble dans lequel la variable prend ses valeurs:</p> <ul style="list-style-type: none"> - Ensemble de fichiers (répertoire ou liste de fichiers. Les caractères génériques (*, *.* , etc.) sont autorisés. Ex: <ensemble> = *.txt ou <ensemble> = fichier1 fichier2, ... - Texte (<ensemble> = 'C'est mon texte' - Flux de sortie d'une commande Ex: <ensemble> = 'dir' - 3 valeurs entières représentant la valeur initiale, la valeur finale et le pas de progression de la variable de boucle (exemple: <ensemble> = 0, 2, 10 (itération de 0 à 10 par pas de 2).
commande(s)	Désigne une commande ou un ensemble de commandes qui seront exécutées pour chaque fichier trouvé dans l'ensemble.
paramètres	Liste des paramètres ou des options pour la commande spécifiée.

EXEMPLE 1:

Le fichier batch suivant permet de lister toutes les lignes de tous les fichiers .bat du répertoire courant qui contiennent la chaîne "set ":

Fichier BoucleFor_1.bat:

```
echo off
for %%V in ( *.bat ) do (
    echo ----- Fichier %%V -----
    type %%V | find "set"
)
```

EXEMPLE 2:

Le fichier batch suivant permet d'afficher les nombres entiers de 1 à 40 par pas de 4:

Fichier BoucleFor_2.bat:

```
echo off
for /l %%l in (1,4,40) do (
    echo - %%l
)
```

EXEMPLE 3:

Le fichier batch suivant permet d'afficher les lignes du fichier semaine.txt en habillant la présentation:

Fichier BoucleFor_3.bat:

```
echo off
for /f "delims=" %%l in (semaine.txt) do (
    echo *****
    echo * %%l
)
echo *****
```


IV.DÉCLENCHEMENT AUTOMATIQUE DE TÂCHES :

IV.1.INTRODUCTION :

Pour administrer les systèmes et réseaux informatiques, il est très utile de pouvoir planifier le déclenchement de certaines tâches à des instants précis, périodiquement ou ponctuellement, sans intervention humaine.

C'est le cas, par exemple, des tâches de sauvegarde qu'il est utile de déclencher systématiquement à des dates précises, de préférence en dehors des heures d'utilisation intensive des systèmes (par exemple, le vendredi soir à 23h30).

IV.2.COMMANDE DE CRÉATION D'UNE TÂCHE:

La commande windows SCHEDULETASKS permet de créer et de gérer des tâches planifiées. Ces tâches peuvent lancer des exécutable (.exe ou .bat) à des instants déterminés.

IV.2.1.FORMAT:

`schtasck <Action> <Attribut-Paramètre, Attribut-Paramètre, ... , Attribut-Paramètre>`

Actions: < create / delete / query / change / run / end / ShowSid >

Attributs-paramètres:

- /tn "<nom de la tâche>"
- /tr "<nom complet fichier exécutable> // *.exe, *.bat, etc.
- /sc <unité de durée> // minute/hourly/daily/weekly/monthly
- /sc <événement déclencheur> // once/onstart/onlogon/ onidle/onevent
- /mo <nombre unités spécifiées par /sc> // nombre entier
- /d <jour(s)> // mon / tue / wed /thu /fri /sat /sun /1-31(si // monthly) / * (tous les jours)
- /sd < date de début > // dd/mm/yyyy
- /ed < date de fin > // dd/mm/yyyy
- /st < heure de début> // hh:mm
- /et < heure de fin> // hh:mm>
- /du <durée de la tâche> // hh:mm>
- /u "<compte utilisateur>" // %computername%\administrateur"
- /p "<mot de passe>" // Si mot de passe absent, il est demandé // en ligne
- /ru <compte utilisateur d'emprunt> // "%computername%\administrateur"
- /rp "< mot de passe d'emprunt >" // Si mot de passe absent, demandé en // ligne
- /rl < highest / limited (défaut) > // Niveau d'exécution

REMARQUE: La syntaxe décrite ci-dessous est incomplète. Pour plus d'informations, saisir la commande: >schtasks /create /?

IV.2.2.EXEMPLES:

PREMIER EXEMPLE:

```
schtasks /create /tn "mytask" /tr "c:\monscript.bat" /sc weekly /mo 2 /d tue /sd 09/06/2018
/st 01:00 /ru "%computername%\administrateur" /rp "password" /rl highest
```

Crée la tâche **mytask** qui exécute la commande **monscript.bat** les mardis (**tue**), toutes les 2 semaines (**/mo 2**) à partir du **09/06/2018** à une heure du matin (**01:00**), sous l'utilisateur **administrateur** au niveau le plus élevé (**highest**).

DEUXIÈME EXEMPLE:

```
schtasks /create /tn "reboot" /tr "shutdown.exe -r -f -t 1 -d p:2:4" /sc weekly /d mon
/sd 15/12/2018 st 06:00 /ru "%computername%\administrateur" /rp "mot de passe" /rl
highest
```

Crée la tâche **reboot** qui exécute la commande **shutdown.exe** les lundis (**mon**) de toutes les semaines à partir du **15/11/2018** à 6 heures du matin (**06:00**), sous l'utilisateur **administrateur** au niveau le plus élevé (**highest**).

IV.3.AFFICHAGE DES TÂCHES PLANIFIÉES:

```
> schtasks /query (toutes)
```

```
> schtasks /query | find "<chaîne à trouver>" ( tâches dont le nom contient la chaîne)
```

IV.4.COMMANDE DE SUPPRESSION D'UNE TACHE:

```
>schtasks /delete /tn "<nom de tâche>"
```

IV.5.DÉMONSTRATIONS:

Pour afficher toutes les minutes le fichier batch **AfficheJoursSemaine.bat**, taper la ligne suivante (après avoir inséré une commande "pause" dans le fichier **AfficheJoursSemaine.bat**, sinon, la fenêtre d'affichage se referme trop vite).

```
> schtasks /create /tn Aff /tr c:\Users\bernal\TestsCMD\AfficheJoursSemaine.bat /sc minute
```

Pour supprimer cette tâche, faire: > schtasks /delete /tn "Aff"

Pour lister toutes les tâches planifiées existantes, taper: > schtasks

V.ANNEXES:

V.1.EXEMPLES PLUS COMPLEXES DE FICHIERS BATCHES:

V.1.1.Calculatrice effectuant les 4 opérations de base (+, -, *, /):

Fichier Calcule.bat

Appel: > Calcule <1er nombre> <Opérateur (+,-,*,/)><Deuxième nombre>

```
echo off
REM *****
REM Cette procedure effectue une des 4 operations de base sur deux nombres
REM entiers (resultat entier).
REM Parametres d'entree:
REM   %1 = premier nombre, %2 = operateur (+,-, *, /), %3 = deuxieme nombre,
REM *****
if %2 == + (
    set /a R=%1+%3
) else (
    if %2 == - (
        set /a R=%1-%3
    ) else (
        if %2 == * (
            set /a R=%1*%3
        ) else (
            if %2 == / (
                set /a R=%1/%3
            ) else (
                set R="Erreur!"
            )
        )
    )
)
echo %1 %2 %3 = %R%
```

V.1.2.Sauvegarde périodique d'un répertoire:

Supposons un poste windows dans lequel existe l'utilisateur local "dev". Le répertoire racine de cet utilisateur sera alors c:\Users\dev.

Supposons également que cet utilisateur regroupe tous ses travaux en cours dans le sous-répertoire c:\Users\dev\TravauxEnCours (par exemple, des fichiers sources en cours de développement). Cet utilisateur veut, à la fin de chaque semaine (par exemple, le vendredi à 22h), sauvegarder automatiquement le contenu de son répertoire de travail dans un sous répertoire de c:\Users\dev\ dont le nom sera de la forme STRA_<Date courante> (par exemple STRA_22_02_2019 pour la sauvegarde du vendredi 22 février 2019).

Créons d'abord la procédure de sauvegarde en elle-même:

```
REM Fichier SauveTravaux.bat:
mkdir c:\Users\dev\STRA_%date:/_=%
xcopy /S c:\Users\dev\TravauxEnCours c:\Users\berna\STRA_%date:/_=%
```

La deuxième ligne "mkdir c:\Users\dev\STRA_%date:/_=%" permet de créer sous c:\Users\dev un sous-répertoire dont le nom est composé du préfixe "STRA_" et dont le suffixe est la valeur de l'expression "%date:/_=%", c'est à dire la date du jour donnée par la variable%date% dans laquelle les caractères "/" ont été remplacés par des caractères "_".

Exemple: le 22 février 2019, la variable%date% aura pour contenu 22/02/2019. Le pattern: "/=_%" va remplacer les "/" par des "_" pour donner 22_02_2019 qui se rajouteront au préfixe STRA_ pour donner "STRA_22_02_2018".

La troisième ligne va lancer la copie du contenu de c:\Users\dev\TravauxEnCours dans c:\Users\dev\STRA_22_02_2019.

Créons maintenant une tâche SVG qui lancera la procédure SauveTravaux.bat tous les vendredi à 22 heures:

```
REM Fichier CreerTachePeriodique.bat
schtasks /create /tn "SVG" /tr "c:\Users\dev\SauveTravaux.bat" /sc weekly/d
wed /st 22:00
```

Cette procédure crée une tâche périodique appelée SVG qui exécute le fichier batch SauveTravaux.bat toutes les semaines (/sc weekly), le vendredi (/d wed) à 22 heures (/st 22:00).

NOTA: Pour arrêter l'exécution, il suffira d'exécuter la commande:

```
> schtasks /delete /tn "SVG"
```

V.1.3.Procédure permettant d'afficher des informations réseau:

Cette procédure utilise les informations fournies par la commande en ligne ifconfig pour afficher diverses informations sur les interfaces réseaux d'une machine windows:

```
echo off
REM Affichage d'informations sur les connexions réseau de la machine.
REM Paramètre d'entrée: aucun
REM Affichage des différents types d'informations sélectionnables
echo "Pour afficher la description des cartes reseau installees, saisir 1"
echo "Pour afficher l'adresse ou les adresses IPV4 de la machine, saisir 2"
echo "Pour afficher le masque de sous-reseau, saisir 3"
echo "Pour afficher la passerelle par defaut, saisir 4"

REM Saisie de l'information à afficher
set /p input=Saisir l'option choisie:

REM Définition de la sous-chaîne permettant de sélectionner dans le flux
REM de sortie de la commande ipconfig /all les informations désirées
if %input% == 1 (
    set string=Description
) else if %input% == 2 (
    set string=IPv4
) else if %input% == 3 (
    set string=Masque
) else if %input% == 4 (
    set string=Passerelle
) else (
    set string=""
)

REM Boucle parcourant ligne par ligne les sorties de la commande ipconfig /all
for /f "delims=" %%i in ('ipconfig /all') do (
    echo %%i | find "Carte"
    echo %%i | find "%string%"
)
```

V.2.COMMANDES DE L'INTERPRÉTEUR CMD:

V.2.1.LISTE DES COMMANDES:

COMMANDE	FONCTION
ASSOC	Affiche ou modifie les applications associées aux extensions de fichiers
ATTRIB	Affiche ou modifie les attributs d'un fichier.
BREAK	Active ou désactive le contrôle étendu de CTRL+C.
BCDEDIT	Définit les propriétés dans la base de données de démarrage pour le contrôle du chargement d'amorçage.
CACLS	Affiche ou modifie les listes de contrôles d'accès aux fichiers.
CALL	Appelle un fichier de commandes à partir d'un autre fichier de commandes
CD	Modifie le répertoire ou affiche le répertoire actif.
CHCP	Modifie ou affiche le numéro de la page de code active.
CHDIR	Modifie le répertoire ou affiche le nom du répertoire actif.
CHKDSK	Vérifie un disque et affiche un rapport d'état.
CHKNTFS	Affiche ou modifie la vérification du disque au démarrage.
CLS	Efface l'écran.
CMD	Exécute une nouvelle instance de l'interpréteur de commandes de WINDOWS
COLOR	Modifie les couleurs du premier plan et de l'arrière-plan de la console
COMP	Compare les contenus de deux fichiers ou groupes de fichiers.
COMPACT	Modifie ou affiche la compression des fichiers sur une partition NTFS.
CONVERT	Convertit des volumes FAT en volumes NTFS. Impossible en cours d'utilisation.
COPY	Copie un ou plusieurs fichiers.
DATE	Affiche ou définit la date.
DEL	Supprime un ou plusieurs fichiers.
DIR	Affiche la liste des fichiers et des sous-répertoires d'un répertoire.
DISKCOMP	Compare les contenus de deux disquettes.
DISKCOPY	Copie le contenu d'une disquette sur une autre.
DISKPART	Affiche ou configure les propriétés d'une partition de disque.

DOC: Administration Systèmes et Réseaux - Langage de script WINDOWS

DIR	Affiche la liste des fichiers et des sous-répertoires d'un répertoire.
DOSKEY	Modifie les lignes de commande, rappelle des commandes WINDOWS, et crée des macros.
DRIVERQUERY	Affiche l'état et les propriétés du pilote de périphérique en cours d'utilisation.
ECHO	Affiche des messages ou active/désactive l'affichage des commandes.
ENDLOCAL	Stoppe la localisation des modifications d'environnement dans un fichier de commandes.
ERASE	Supprime un ou plusieurs fichiers.
EXIT	Quitte l'interpréteur de commandes (CMD.EXE).
FC	Compare deux fichiers ou groupes de fichiers et affiche les différences.
FIND	Recherche une chaîne de caractères dans un ou plusieurs fichiers.
FINDSTR	Cherche des chaînes dans les fichiers.
FOR	Exécute une commande sur chaque fichier d'un ensemble de fichiers.
FORMAT	Formate un disque devant être utilisé avec WINDOWS.
FSUTIL	Affiche ou configure les propriétés du système de fichiers.
FTYPE	Affiche ou modifie les types de fichiers utilisés dans les associations d'extensions.
GOTO	Indique l'exécution d'un fichier de commandes pour une ligne identifiée par une étiquette.
GPRESULT	Affiche les informations de stratégie de groupe pour un ordinateur ou un utilisateur.
GRAFTABL	Permet à WINDOWS d'afficher un jeu de caractères en mode graphique.
HELP	Affiche des informations sur les commandes de WINDOWS.
ICACLS	Afficher, modifier, sauvegarder ou restaurer les listes de contrôle d'accès pour les fichiers et les répertoires.
IF	Effectue un traitement conditionnel dans un fichier de commandes
LABEL	Crée, modifie ou supprime le nom de volume d'un disque.
MD	Crée un répertoire.
MKDIR	Crée un répertoire.

DOC: Administration Systèmes et Réseaux - Langage de script WINDOWS

MKLINK	Créer des liens symboliques et des liens réels
MODE	Configure un périphérique du système.
MORE	Affiche la sortie écran par écran.
MOVE	Déplace un ou plusieurs fichiers d'un répertoire à un autre.
OPENFILES	Affiche les fichiers partagés ouverts à distance par les utilisateurs.
PATH	Affiche ou définit le chemin de recherche des fichiers exécutables.
PAUSE	Interrompt l'exécution d'un fichier de commandes et affiche un message.
POPD	Restaure la valeur précédente du répertoire actif enregistrée par PUSHD.
PRINT	Imprime un fichier texte.
PROMPT	Modifie l'invite de commande de WINDOWS.
PUSHD	Enregistre le répertoire actif puis le modifie.
RD	Supprime un répertoire.
RECOVER	Récupère l'information lisible d'un disque défectueux.
REM	Insère un commentaire dans un fichier de commandes ou CONFIG.SYS.
REN	Renomme un ou plusieurs fichiers.
RENAME	Renomme un ou plusieurs fichiers.
REPLACE	Remplace des fichiers.
RMDIR	Supprime un répertoire.
ROBOCOPY	Utilitaire avancé pour copier les fichiers et les arborescences de répertoires
SET	Affiche, définit ou supprime des variables d'environnement WINDOWS.
SETLOCAL	Commence la localisation des modifications d'environnement dans un fichier de commandes.
SC	Affiche ou configure les services (processus en arrière-plan).
SCHTASKS	Planifie les commandes et les programmes à exécuter sur l'ordinateur.
SHIFT	Modifie la position des paramètres remplaçables dans un fichier de commandes.
SHUTDOWN	Permet un arrêt local ou distant correct de l'ordinateur.
SORT	Trie les entrées.

SHIFT	Modifie la position des paramètres remplaçables dans un fichier de commandes.
START	Ouvre une fenêtre séparée pour l'exécution d'un programme ou d'une commande spécifique.
SUBST	Associe un chemin d'accès à une lettre de lecteur.
SYSTEMINFO	Affiche les propriétés et la configuration spécifiques de l'ordinateur.
TASKLIST	Affiche toutes les tâches en cours d'exécution, y compris les services.
TASKKILL	Termine ou interrompt un processus ou une application en cours d'exécution.
TIME	Affiche ou définit l'heure du système.
TITLE	Définit le titre de la fenêtre pour une session CMD.EXE.
TREE	Affiche le graphisme de la structure de répertoire d'un lecteur ou d'un chemin d'accès.
TYPE	Affiche le contenu d'un fichier texte.
VER	Affiche la version de WINDOWS.
VERIFY	Demande à WINDOWS de vérifier si vos fichiers sont correctement écrits sur le disque.
VOL	Affiche le nom et le numéro de série d'un volume de disque.
XCOPY	Copie les fichiers et les arborescences de répertoires.
WMIC	Affiche les informations WMI dans l'interface de commande interactive.

V.2.2.AIDE SUR LES COMMANDE:

Aide sur une commande: `C:\user\my>help <nom de la commande>`

Aide sur toutes les commandes: `C:\user\my>help`

V.3.PRINCIPALES VARIABLES D'ENVIRONNEMENT:

VARIABLE	VALEUR
%COMPUTERNAME%N	Nom de l'ordinateur
%COMSPEC%	Emplacement de la console
%HOMEDRIVE%	Emplacement du lecteur principal
%HOMEPATH%	Indique où se trouve le dossier personnel de l'utilisateur en cours
%LOGONSERVER%	Adresse réseau de l'ordinateur
%NUMBER_OF_PROCESSORS%	Nombre de processeurs installés sur l'ordinateur
%OS%	Système d'exploitation installé
%PATH%	Chemins d'accès déclarés
%PROCESSOR_ARCHITECTURE%	Architecture du processeur
%PROCESSOR_IDENTIFIER%	Identification du processeur
%PROGRAMFILES%	Adresse du répertoire des programmes
%SYSTEMDRIVE%	Lecteur principal
%SYSTEMROOT%	Répertoire où se situe le système
%TEMP%	Adresse du répertoire temporaire
%USERNAME%	Nom de l'utilisateur
%USERPROFILE%	Répertoire personnel de l'utilisateur en cours
%WINDIR%	Répertoire où est installé WINDOWS
%DATE%	Date
%TIME%	Heure
%CD%	Emplacement actuel de la console
%RANDOM%	Nombre aléatoire
%CMDCMDLINE%	Emplacement de la console.