**Portfolio**

(Entreprise numérique)

Noah Liurno TI-G

Sam Lenaerts TI-G

Tiago Lupascu TI-G

**Table des matières**

[1 Programmation (Langage en C) (Noah Liurno) 3](#_Toc155515454)

[1.1 Déclaration et affectation d’une variable 3](#_Toc155515455)

[1.2 Entrées et sorties 4](#_Toc155515456)

[1.3 Les conditions 5](#_Toc155515457)

[1.4 Les boucles / répétitive 6](#_Toc155515458)

[1.5 Les Tableaux 6](#_Toc155515459)

[1.6 Les Fonctions 7](#_Toc155515460)

[1.7 Source utilisée 7](#_Toc155515461)

[2 Commandes OS sous Linux et Windows (Sam Lenearts) 8](#_Toc155515462)

[2.1 Ligne de commandes sous Linux 8](#_Toc155515463)

[2.1.1 Les commandes entrées sur l’Invite Shell comportent **trois parties principales** 8](#_Toc155515464)

[2.1.2 Liste de commandes avec l’environnements GNOME 8](#_Toc155515465)

[2.1.3 Liste de commandes pour le traitement de texte (Commandes Nano) 8](#_Toc155515466)

[2.2 Ligne de commandes sous Windows 9](#_Toc155515467)

[2.2.1 Les variables d’environnement: 9](#_Toc155515468)

[2.2.2 Commandes de base: 9](#_Toc155515469)

[2.2.3 Commandes de gestion de fichiers: 9](#_Toc155515470)

[2.2.4 Les jokers correspondent à différents types de caractères qui sont utilisé pour chercher un mot d’une expression 9](#_Toc155515471)

[2.2.5 Commandes de gestion des utilisateurs: 9](#_Toc155515472)

[3 Commandes Réseau pour Windows / Linux (Tiago Lupascu) 10](#_Toc155515473)

[3.1 Série des commandes LINUX 10](#_Toc155515474)

[3.1.1 Commandes utiles 10](#_Toc155515475)

[3.1.2 Les commandes ip addr 10](#_Toc155515476)

[3.1.3 Les commandes de route 10](#_Toc155515477)

[3.2 Commandes réseau windows 11](#_Toc155515478)

[3.2.1 Commandes utiles 11](#_Toc155515479)

[3.2.2 Les commandes NETSH 11](#_Toc155515480)

[3.2.3 Les commandes NETSH (DNS) 12](#_Toc155515481)

[3.2.4 es commandes de route 12](#_Toc155515482)

# Programmation (Langage en C) (Noah Liurno)

#include <…>

Permet d’inclure des bibliothèques qui rajoute des fonctions au programme par exemple la bibliothèque stdio.h permet d’utiliser la fonction printf.

// Commentaires

Les // permettent la création d’un commentaire cela peut être utile pour faciliter la lecture d’un programme et pouvoir ainsi s’y retrouver plus facilement.

(pour réaliser un commentaire sur plusieurs lignes on peut utiliser /\* au début et \*/ à la fin du commentaire).

## Déclaration et affectation d’une variable

<var\_type> <var\_name> ;

Déclare une variable, lui donne un nom et permet d’y stocker une valeur en fonction du type de cette variable.

Quelques types de variable fréquent:

* Pour stocker un **entiers** on utilise **int**,
* Pour stocker un **réel** on utilise **double**,
* Pour stocker un **booléens** on utilise **bool**,
* Pour stocker un **caractères** on utilise **char**.

<var\_name> = <value> ;

Affecte une valeur à la variable cette valeur est stocker dans une zone mémoire (la valeur peut être le résultat d’un calcule ou une constante littérale mais attention car la valeur est influencée par le type de la variable.

## Entrées et sorties

printf("le nombre est <write\_format>", <var\_name>) ;

Cette fonction affiche des informations à l’écran (le plus souvent des chaines de caractères et la/les valeur(s) d’une ou plusieurs variables donnée(s)).

Le write format est le format d’écriture de la valeur demandée il change en fonction du type de sa variable (les différent write format dans le tableau ci-dessous).

Scanf\_s("<read\_format>", &<var\_name>);

Scanf va lire le clavier de l’utilisateur ce qui va lui permettre d’enregistrer ce qu’il écrit dans une variable que ce soient des lettres, des chiffres ou une chaine de caractères.

Le read format est le format de lecture de la valeur écrite il change en fonction du type de sa variable (les différent read format dans le tableau ci-dessous).

|  |  |  |
| --- | --- | --- |
| Type les plus courants | <write\_format> | <read\_format> |
| Int | %d | %d |
| Long | %ld | %ld |
| Float | %f | %f |
| Double | %f | %lf |
| Char | %c | %c |

Getchar();

Le getchar permet de lire un seul caractère entrer au clavier et peut le stocker dans une variable.

Putchar(...);

Putchar va afficher un caractère stocké

## Les conditions

If (condition [opérateur logique condition]) {

Si la condition est respectée alors …

} else {

Si la condition n’est pas respectée alors …

}

|  |  |
| --- | --- |
| Opérateurs de comparaison | Traduction |
| == | Égal |
| != | Différent |
| < | Plus petit |
| > | Plus grand |
| <= | Plus petit ou égal |
| >= | Plus grand ou égal |
| Opérateurs  logique | Traduction |
| && | Et |
| || | Ou |
| ! | Non |

## Les boucles / répétitive

While (condition) {…}

While permet de répéter ce qui est entre les trémas jusqu’à ce que la condition soit respectée.

On utilise while quand on ne connait pas le nombre d’itération

For (initialization ; condition ; incrementation) {…}

For permet aussi de répéter ce qui est entre les trémas jusqu’à ce que la condition soit respectée.

Mais à la différence du while on l’utilise quand on connait le nombre d’itération.

Do {…} while (condition)

Do…While est utilisée pour pouvoir exécuter la condition si celle-ci est fausse dès le départ puisse que la condition se fait à la fin.

## Les Tableaux

Int nomDuTableau[nombreDeCases];

Permet la création d’un tableau contenant le nombre de cases souhaitée.

nomDuTableau[caseSouhaitée] = valeur ;

Permet d’inscrire une valeur dans n’importe qu’elle case souhaitée.

ATTENTION !

Si on crée un tableau de 3 cases celui-ci contiendras les cases de 0 à 2 et non pas de 1 à 3, la première case d’un tableau commence par 0.

## Les Fonctions

Type nomFonction(paramètres) {…}

Permet la création d’une fonction pour découper un morceau de code qui plus tard pourras être rappeler dans le programme.

Les paramètres sont les valeurs que la fonction va utiliser.

## Source utilisée

<https://openclassrooms.com/fr/courses/19980-apprenez-a-programmer-en-c>

Cours de langage en C hénallux

# Commandes OS sous Linux et Windows (Sam Lenearts)

## Ligne de commandes sous Linux

Les principales commandes permettant de découvrir le **CLI** et se familiariser avec certains raccourcis sous Linux.

Concept du Shell Bash.

### Les commandes entrées sur l’Invite Shell comportent **trois parties principales**

* La **commande** à exécuter,
* Les **options** qui ajustent le comportement de la commande,
* Les **arguments** qui sont le plus souvent les cibles de la commande.

### Liste de commandes avec l’environnements GNOME

* **MKDIR** (make directory) permet de créer un répertoire
* **TOUCH** permet de créer un fichier
* **DATE** permet d’afficher l’aide d’une commande (mkdir date pour exemple)
* **HEAD** et **TAIL** affichent le début et la fin d’un fichier + **l’option -n** afin de spécifier un nombre différent de lignes comparé à celui de base
* **WC** compte les lignes, mots et caractères d’un fichier + **-l** (ligne) **-w** (mots) et également **-c** (caractère) uniquement
* **HISTORY** affiche une liste des commandes précédemment exécutées
* **SU – ROOT** permet de se connecter en root

### Liste de commandes pour le traitement de texte (Commandes Nano)

* **NANO** nom fichier permet d’ouvrir ou créer un fichier, Nano s’ouvrira,
* **CTRL + O** permet de sauvegarder l’édition de ce fichier,
* **CTRL + X** ferme Nano,
* **CTRL + W** permet de chercher et remplacer du texte,
* **CTRL + A** lance une ligne**,**
* **CTRL + E** termine une ligne,
* **CTRL + T** exécute une commande.

Nano serait considéré comme bien pour les novices de Linux grâce à sa simplicité à le prendre en main.

## Ligne de commandes sous Windows

Une série de commandes qui vont pouvoir nous faire découvrir le **CMD** et de se familiariser avec les principales commandes de Windows.

On distingue deux catégories de commandes :

* Les commandes intégrées dans l’invite de commandes
* Les commandes externes pouvant être exécutées

### Les variables d’environnement:

* **PATH** est la + importante car elle comprend **la liste des répertoires** contenant les programmes que le système peut exécuter,
* **SET** affiche la liste des variables d’environnements.

### Commandes de base:

* **CD** permet de se déplacer dans l’arborescence de fichiers,
* L’extension **/A** après **CD** va afficher les **fichiers cachés,**
* **/P** interrompt l’affichage lorsque l’écran est plein,
* **/S** permet de voir les sous-répertoires.

### Commandes de gestion de fichiers:

* **MKDIR** exactement comme Linux,
* **RMDIR** quant à lui vas supprimer le répertoire,
* **COPY** peut copier un fichier source vers une autre destination et possible **sous un autre nom.**

### Les jokers correspondent à différents types de caractères qui sont utilisé pour chercher un mot d’une expression

* **?** remplace **un seul** caractère du nom ou de l’extension d’un fichier par exemple,
* **\*** remplace un **ensemble** de caractère indéfini.

### Commandes de gestion des utilisateurs:

* **NET USER** permet de créer et de gérer les utilisateurs,
* **NET LOCALGROUP** également mais pour les groupes.

# Commandes Réseau pour Windows / Linux (Tiago Lupascu)

## Série des commandes LINUX

### Commandes utiles

**ping <Adresse>**

permet de ping (envoyer un paquet) et voir si un pc/appareil peut répondre et s'il répond ça veut dire qu'il est Joignable.

### Les commandes ip addr

**ip addr show ou ip a**

permet de voir les différentes interfaces réseau avec les adresses ip attribués à celles-ci ainsi que leurs adresses MAC.

**ip addr add** **<Adresse IP></Mask dev <nomInterfaceRéseau>**

Changer adresse ip et ajouter des adresses IP.

**ip addr flush dev** **"Nom interface"**

Pour retirer toute la configuration d'une interface (une carte réseau).

**ip addr delete <adresse IP></Mask> dev <InterfaceReseau>**

Effacer adresse ip commande.

**Pour ces 3 commandes du haut si c'est de l'ipv6 il suffit de rajouter -6 entre addr et ip et de mettre le masque en / collée à l'adresse en héxadécimale**

### Les commandes de route

Pour activer le routage en ligne de commande il faut taper : **nano /etc/sysctl.conf** et puis aller là : **net.ipv4.ip\_forward=1** et enlever le hashtag qui se trouve en début de ligne

**ip route add <adresse IP destination></Mask> via <adresse IP par laquelle ça passe>**

Ajouter une route

## Commandes réseau windows

### Commandes utiles

**ping <Adresse>**

permet de voir si quelque chose est joignable ou pas donc envoi un paquet et si l'appareil répond alors cela veut dire qu'il est joignable.

**IpConfig/all**

permet de voir les différentes interfaces avec les différentes adresses IP et les Adresses MAC.

accéder interface graphique : Windows + R **« ncpa.cpl » ou aller dans recherche et taper ncap.cpl.**

### Les commandes NETSH

**netsh interface show interface**

permet de voir le nom des différentes interfaces et voir si elles sont connectées ou pas.

**netsh interface set interface "Nom interface" disable**

Désactiver une carte réseau : pour désactiver il faut taper.

**netsh interface set interface "Nom interface" enable.**

Activer la carte réseau.

**netsh interface ip set address "NOM DE L'INTERFACE" static "IP\_SOUHAITEE" "MASQUE\_DE\_SOUS\_RESEAU" "PASSERELLE"**

Changer Adresse ip Windows commande.

**Pour mettre en IPV6 c'est la même chose à part qu'il faut changer le ipv4 en ipv6 et biensûr mettre les adresses en hexadécimal**

**netsh interface ip set adress "Nom Interface" dhcp**

Pour mettre en dynamique/automatique il suffit de taper cela.

**Après pour que ça le prenne en compte il faut faire ipconfig /renew pour demander une nouvelle adresse**

**netsh interface ip add address "Nom Interface" "Adresse IP" "Masque"**

Pour ajouter plusieurs adresses IP sur une même interface il faut taper.

**Même chose pour le Ipv6 à part qu'il faut mettre ipv6 à la place de ip et mettre le masque après l'adresse ip collée en slash après celle-ci**

### Les commandes NETSH (DNS)

**netsh interface ip set dns "Nom Interface" static "Adresse DNS"**

Pour changer le dns il faut le configurer à part (Pour une adresse statique).

**netsh interface ip set dns "Nom interface" dhcp**

(Pour du dynamique/automatique).

**netsh interface ip add dns "Nom interface" "Adresse DNS"**

Pour ajouter un DNS.

### es commandes de route

**route add -p "Adresse défaut=en général 0.0.0.0" mask "Masque" "Passerelle"**

ajouter une route.

**route delete "Adresse ip de la route"**

Pour supprimer il faut taper.