# Manuel Utilisateur

Le **manuel utilisateur** décrit comment utiliser les programmes développés illustrés avec des copies d'écran.

## Pré-requis

- Télécharger GNAT, un compilateur ADA
- Ajouter GNAT dans les variables d'environnement si ce n'est pas fait automatiquemennt
- Cloner ce dépôt Git

Maintenant que toutes ces étapes sont faites, vous avez de quoi compiler correctement le programme.

## Compilation

Ouvrir un **terminal de commandes** et se déplacer (cd) dans l'onglet src du projet git cloné puis exécuter la commande suivante :

> gnatmake .\menu.adb

Le programme a **compilé** correctement ! Pour **exécuter le programme** : il ne reste plus qu'à lancer la commande suivante :

> .\menu

## Utilisation programme

### Démarrage

Au démarrage du programme s'affiche le message suivant :

> Vous pouvez creer un SGF en ecrivant start

Cela signifie qu'il faut créer taper la commande 'start' afin de **créer un SGF** avec **un** dossier racine 'root'. Tant que vous n'avez pas taper la commande **start**, l'interface vous redemandera d'écrire start.

Il sera possible de **réinitialiser** le SGF par la suite grâce à l'instruction 'reset' qui vous permettra de remplacer le SGF actuel par un nouveau SGF avec seulement le dossier racine 'root'.

### Fonctionnement SGF

Il faut savoir que le SGF possède **un répertoire courant** des répertoires (fichier ou dossier) un dossier peut contenir d'autres fichiers et d'autres dossiers.

Une **destination (chemin)** est un chemin d'accès à un répertoire et est délimitée par des '/' entre chaque répertoire et il peut être représenté de 2 manières distinctes :

- **Chemin absolu** : Le chemin absolu débute par un '/' et part du répertoire racine (exemple : '*/home/enzo/ada/tp*').
- **Chemin relatif** : Le chemin relatif part du répertoire actuel du SGF (exemple : '*enzo/ada/tp*' avec comme répertoire actuel '*home*').

De plus dans un chemin, le caractère '.' représente le **répertoire actuel** et '..' représente le **répertoire père** d'un répertoire.

> Aide : Si une incompréhension subsiste liée au fonctionnement des SGF accéder au sujet de projet qui l'explicite de manière plus détaillée.

## Liste des commandes :

- **Accéder à son répertoire actuel (pwd)**

  Cette commande permet **d'accéder au répertoire courant** du sgf. Elle s'utilise avec le mot-clé **pwd** et sans aucun paramètre.

*Exemple :*

```
> pwd
/root

> pwd
/root/home/enzo
```

- **Déplacement répertoire actuel (cd)**

  Cette commande permet de **changer le répertoire courant** au SGF à l'aide d'une destination. Elle s'utilise avec le mot-clé **cd** et avec **1** paramètre : destination_dossier qui représente un chemin absolu ou relatif.

  *Exemple :*

```
> cd /home/enzo/ada/tp

> pwd
/root/home/enzo/ada/tp

> cd ../..

> pwd
/root/home/enzo

> cd ada/tp

> pwd
/root/home/enzo/ada/tp
```

**Cas d'erreur :** Si le nombre de paramètres n'est pas de 1, le programme affiche un message d'erreur Si le chemin n'existe pas, le programme affiche un message d'erreur

*Exemple :*

```
> cd param1 param2
Nombre de parametres incorrect, la commande correcte est 'cd [destination_repertoire]'

> cd param1 param2 param3
Nombre de parametres incorrect, la commande correcte est 'cd [destination_repertoire]'

> cd dossierQuiExistePas/tp
Erreur : Le chemin specifie n'existe pas !
```

- ## Création d'un fichier (touch)

Cette commande permet de **créer un fichier** dans une destination du SGF. Elle s'utilise avec le mot-clé **touch** et avec **1** paramètre : nom du fichier à créer (vous pouvez spécifier une destination précise où créer le fichier en mettant la destination où vous souhaiter le créer avant le nom du fichier délimité par un '/').

*Exemple :*

```
> touch fichier1
Le fichier fichier1 a ete cree avec succes

> touch /home/enzo/ada/tp/tp1.adb
Le fichier tp1.adb a ete cree avec succes
```

**Cas d'erreur :** Si le nombre de paramètres n'est pas de 1, le programme affiche un message d'erreur Si un fichier du même nom existe déjà dans le répertoire où nous voulons le créer Si le chemin n'existe pas, le programme affiche un message d'erreur

*Exemple :*

```
> touch home/pasEnzo/fichierAcreer
Erreur : Le chemin specifie n'existe pas

> touch home/enzo/ada/tp/tp1.adb
Erreur : Le fichier tp1.adb existe deja dans le repertoire courant

> touch param1 param2
Nombre de parametres incorrect, la commande correcte est 'touch |destination|[nom_fichier]'
```

- ## Création d'un dossier (mkdir)

Cette commande permet de **créer un dossier** dans une destination du SGF. Elle s'utilise avec le mot-clé **mkdir** et avec **1** paramètre : nom du dossier à créer (vous pouvez spécifier une destination précise où créer le fichier en mettant la destination où vous souhaiter le créer avant le nom du fichier délimité par un '/').

*Exemple :*

```
> mkdir /home/enzo/ada/projet
Le dossier projet a ete cree avec succes

> mkdir home/enzo/reseau
Le dossier reseau a ete cree avec succes
```

**Cas d'erreur :** Si le nombre de paramètres n'est pas de 1, le programme affiche un message d'erreur Si un dossier du même nom existe déjà dans le répertoire où nous voulons le créer Si le chemin n'existe pas, le programme affiche un message d'erreur

*Exemple :*

```
> mkdir home/enzo/reseau
Erreur : Le dossier reseau existe deja dans le repertoire courant

> mkdir
Nombre de parametres incorrect, la commande correcte est 'mkdir |destination|[nom_dossier]'

> mkdir home/enzo/informatique/dossier
Erreur : Le chemin specifie n'existe pas
```

- ## Afficher les répertoires contenus dans un répertoire (ls)

Cette commande permet de **lister les répertoires** contenus dans un répertoire du SGF. Elle s'utilise avec le mot-clé **ls** et avec **0** ou **1** paramètre facultatif : nom du répertoire où l'on cherche à lister les répertoires, sinon affiche les répertoires contenus dans le répertoire courant.

*Exemple :*

```
> ls
Name                    Length          Rights          File
=============================================================
dossier                   505             777          Dossier

> ls /dossier
Name                    Length          Rights          File
=============================================================
fichier2                  5             777           Fichier
fichier1                  500             777           Fichier
```

**Cas d'erreur :** Si le nombre de paramètres est supérieur à 1, le programme affiche un message d'erreur Si le chemin n'existe pas, le programme affiche un message d'erreur

*Exemple :*

```
> ls dossier4
Erreur : Le chemin specifie n'existe pas !

> ls param1 param2
Commande incorrecte, la commande correcte est 'ls |destination|' ou 'ls -r |destination|'
```

- ## Afficher les répertoires et sous-répertoires dans un répertoire (ls -r)

  Cette commande permet de **lister les répertoires et sous répertoires** contenus dans un SGF. Elle s'utilise avec le mot-clé **ls -r** et avec **0** ou **1** paramètre facultatif : nom du répertoire où l'on cherche à lister les répertoires et sous répertoires, sinon affiche les répertoires et sous répertoires contenus dans le répertoire courant.

  *Exemple :*

```
> ls -r
/
 \- usr
    \- local
       \- share
 \- home
    \- user1
       \- methprog
          \- tp
             \- tp1
                |- enzoF.adb
                |- puissance.adb
                |- newton.adb
                |- min_max_serie.py
                |- min_max_serie.adb
          \- projet
             |- exemple.adb

> ls -r home/user1/methprog/tp
          \- tp
             \- tp1
                |- enzoF.adb
                |- puissance.adb
                |- newton.adb
                |- min_max_serie.py
                |- min_max_serie.adb
          \- projet
             |- exemple.adb
```

**Cas d'erreur :** Si le nombre de paramètres est supérieur à 1, le programme affiche un message d'erreur Si le chemin n'existe pas, le programme affiche un message d'erreur

*Exemple :*

```
> ls -r param1 param2
Commande incorrecte, la commande correcte est 'ls |destination|' ou 'ls -r |destination|'

> ls -r usr/enzo
Erreur : Le chemin specifie n'existe pas !
```

- ## Supprimer un fichier (rm)

  Cette commande permet de **supprimer un fichier** dans une destination du SGF. Elle s'utilise avec le mot-clé **rm** et avec **1** paramètre : nom du fichier à supprimer (vous pouvez spécifier une destination précise où supprimer le fichier en mettant la destination où vous souhaiter le supprimer avant le nom du fichier délimité par un '/').

  *Exemple :*

```
> rm ../fichier
Le fichier fichier a ete supprime avec succes

> rm enzo/ada/tp/tp1.adb
Le fichier tp1.adb a ete supprime avec succes
```

**Cas d'erreur :** Si le nombre de paramètres n'est pas de 1, le programme affiche un message d'erreur Si le fichier n'existe pas dans le répertoire où nous voulons le supprimer Si le chemin n'existe pas, le programme affiche un message d'erreur

*Exemple :*

```
> rm enzo/ada/tp/tp8.adb
Le fichier tp8.adb a supprimer n'existe pas dans le dossier tp

> rm enzo/ada/tp/projet/tp/projet/tp/tp1.adb
Erreur : Le chemin specifie n'existe pas !

> rm
Commande incorrecte, la commande correcte est 'rm |destination|[nom_fichier]' ou 'rm -r |destination|[nom_dossier]'
```

- ## Supprimer un dossier (rm -r)

Cette commande permet de **supprimer un dossier** dans une destination du SGF. Elle s'utilise avec le mot-clé **rm -r** et avec **1** paramètre : nom du dossier à supprimer (vous pouvez spécifier une destination précise où supprimer le dossier en mettant la destination où vous souhaiter le supprimer avant le nom du dossier délimité par un '/').

*Exemple :*

```
> rm -r /home/enzo
Le dossier enzo a ete supprime avec succes

> rm -r damien
Le dossier damien a ete supprime avec succes
```

**Cas d'erreur :** Si le nombre de paramètres n'est pas de 1, le programme affiche un message d'erreur Si le dossier n'existe pas dans le répertoire où nous voulons le supprimer Si on cherche à supprimer un dossier dans lequel le répertoire courant du SGF est contenu dans le répertoire à supprimer Si le chemin n'existe pas, le programme affiche un message d'erreur

*Exemple :*

```
> rm -r ../home
Erreur : Impossible de supprimer un dossier dans lequel vous ├¬tes actuellement

> rm -r
Commande incorrecte, la commande correcte est 'rm |destination|[nom_fichier]' ou 'rm -r |destination|[nom_dossier]'

> rm -r dossier
Le dossier dossier a supprimer n'existe pas dans le dossier root

> rm -r dossier/dossier1/dossier2
Erreur : Le chemin specifie n'existe pas !
```

- ## Changer la taille d'un fichier (size)

  Cette commande permet de **changer la taille d'un fichier** Elle s'utilise avec le mot-clé **size** et avec **2** paramètre : **nom du fichier** à changer de taille (vous pouvez spécifier une destination précise où créer le fichier en mettant la destination où vous souhaiter changer la taille d'un fichier avant le nom du fichier délimité par un '/ ') et la **nouvelle taille** avec un entier.

  *Exemple :*

```
> ls
Name                    Length          Rights          File
============================================================
dossier                   10             777           Dossier

> ls dossier
Name                    Length          Rights          File
============================================================
fichier2                  5              777           Fichier
fichier1                  5              777           Fichier

> size dossier/fichier1 500

> ls
Name                    Length          Rights          File
============================================================
dossier                   505             777           Dossier

> ls dossier
Name                    Length          Rights          File
============================================================
fichier2                  5              777           Fichier
fichier1                  500             777           Fichier
```

  **Cas d'erreur :** Si le nombre de paramètres n'est pas de 2, le programme affiche un message d'erreur Si le fichier n'existe pas dans le répertoire où nous voulons changer sa taille Si le chemin n'existe pas, le programme affiche un message d'erreur Si la taille est un entier négatif

  *Exemple :*

```
> size fichier1 -5
Erreur : La taille ne peut pas etre negative

> size fichier2 4
Erreur : Le fichier 'fichier2' n'existe pas dans le dossier root

> size dossier/dossier/fichier1 5
Erreur : Le chemin specifie n'existe pas !

> size fichier1
Nombre de parametres incorrect, la commande correcte est 'size |destination|[nom_fichier] [taille_fichier]'
```

- ## Copier un fichier dans un autre répertoire (cp)

Cette commande permet de **copier un fichier** dans un autre répertoire. Elle s'utilise avec le mot-clé **cp** et avec **2** paramètre : **nom du fichier** à copier (vous pouvez spécifier une destination précise où copier le dossier en mettant la destination où vous souhaiter le copier avant le nom du dossier délimité par un '/') ainsi que la **destination** où coller le fichier. Il est éventuellement possible de **renommer le fichier copié** quand nous allons le coller en spécifiant un nouveau nom de fichier lors de la destination où coller le fichier en ajoutant un '/' et en écrivant le nom du nouveau fichier à la suite de celui-ci.

*Exemple :*

```
> ls dossier
Name                    Length          Rights          File
============================================================
f2                       5              777           Fichier
test                     5              777           Fichier

> ls hello
Le dossier hello ne possede pas de repertoires

> cp dossier/test hello/f1
Le fichier f1 a ete cree via copie avec succes dans le dossier hello

> cp dossier/test hello
Le fichier test a ete cree via copie avec succes dans le dossier hello

> ls hello
Name                    Length          Rights          File
============================================================
test                     5              777           Fichier
f1                       5              777           Fichier
```

**Cas d'erreur :** Si le nombre de paramètres n'est pas de 2, le programme affiche un message d'erreur- Si le fichier à copier n'existe pas Si l'un des chemin n'existe pas, le programme affiche un message d'erreur

*Exemple :*

```
> cp param1
Commande incorrecte, la commande correcte est 'cp |destination_source|[nom_fichier] [destination_cible]'

> cp param1/param2/param3 param1/param2/param3
Erreur : Le chemin 'param1/param2/param3' n'existe pas !

> cp usr/fichier home
Le fichier fichier n'existe pas dans le dossier usr
```

- ## Déplacer un fichier dans un autre répertoire (mv)

Cette commande permet de **déplacer un fichier** dans un autre répertoire. Elle s'utilise avec le mot-clé **mv** et avec **2** paramètre : **nom du fichier** à déplacer (vous pouvez spécifier une destination précise où déplacer le dossier en mettant la destination où vous souhaiter le déplacer avant le nom du dossier délimité par un '/') ainsi que la **destination** où coller le fichier. Il est éventuellement possible de **renommer le fichier** quand nous allons le déplacer en spécifiant un nouveau nom de fichier lors de la destination où déplacer le fichier en ajoutant un '/' et en écrivant le nom du nouveau fichier à la suite de celui-ci (il est possible de renommer un fichier sans le déplacer en spécifiant la même destination).

*Exemple :*

```
> mv hello/f1 dossier/test
Le fichier test a ete deplace avec succes dans le dossier dossier

> mv hello/f2 dossier
Le fichier f2 a ete deplace avec succes dans le dossier dossier

> ls dossier
Name                    Length          Rights          File
============================================================
f2                       5              777           Fichier
test                     5              777           Fichier
```

**Cas d'erreur :** Si le nombre de paramètres n'est pas de 2, le programme affiche un message d'erreur- Si le fichier à copier n'existe pas Si l'un des chemin n'existe pas, le programme affiche un message d'erreur

*Exemple :*

```
> mv param1
Commande incorrecte, la commande correcte est 'mv |destination_source|[nom_fichier] [destination_cible]'

> mv param1/param2/param3 param1/param2/param3
Erreur : Le chemin 'param1/param2/param3' n'existe pas !

> mv usr/fichier home
Le fichier fichier n'existe pas dans le dossier usr
```

- ## Archiver un dossier en un fichier (tar)

Cette commande permet **d'archiver un dossier** (créer un fichier de même taille dans le répertoire père. Elle s'utilise avec le mot-clé **tar** et avec **1** paramètre : nom du dossier à archiver (vous pouvez spécifier une destination précise où archiver le dossier en mettant la destination où vous souhaiter l'archiver avant le nom du dossier délimité par un '/').

*Exemple :*

```
> mkdir hello
Le dossier hello a ete cree avec succes

> touch hello/f1
Le fichier f1 a ete cree avec succes

> tar /hello

> ls
Name                    Length          Rights          File
============================================================
hello.zip                 5             777           Fichier
hello                     5             777           Dossier
```

**Cas d'erreur :** Si le nombre de paramètres n'est pas de 1, le programme affiche un message d'erreur Si le dossier n'existe déjà dans le répertoire où nous voulons l'archiver Si le chemin n'existe pas, le programme affiche un message d'erreur

*Exemple :*

```
> tar hella
Le dossier hella n'existe pas dans le dossier

> tar dossier/dossier/dossier
Erreur : Le chemin '' n'existe pas !

> tar tar tar
Commande incorrecte, la commande correcte est 'tar [destination_dossier]'
```

- ## Réinitialiser le SGF (reset)

Cette commande permet de **réinitialiser le SGF** (remplacer le SGF actuel par un nouveau contenant qu'un répertoire racine). Elle s'utilise avec le mot-clé **reset** et sans aucun paramètre.

*Exemple :*

```
> mkdir hello
Le dossier hello a ete cree avec succes

> reset
Reinitialisation du sgf avec succes

> cd hello
Erreur : Le chemin specifie n'existe pas !
```

- ## Accéder à une page d'aide (help)

  Cette commande permet d'avoir **la liste des commandes existantes** du SGF. Elle s'utilise avec le mot-clé **help** et sans aucun paramètre.

  *Exemple :*

```
> help
Voici les differentes commandes que vous pouvez ecrire :

=> cd : |destination| : Changer de direction de repertoire

=> cp |destination_source|[nom_fichier] [destination_cible] : Copier un fichier

=> ls |destination| : Afficher la liste des fichiers et dossiers

=> ls -r |destination| : Afficher tous les repertoires et tous les sous-repertoires

=> mkdir |destination|[nom_dossier] : Creer un dossier

=> mv |destination_source|[nom_fichier] [destination_cible] : Deplacer un fichier

=> pwd : Afficher le repertoire courant

=> quit : Quitter l'interface

=> reset : Creer un SGF avec que le repertoire racine

=> rm |destination|[nom_fichier] : Supprimer un fichier

=> rm -r |destination|[nom_dossier] : Supprimer un dossier

=> size |destination|[nom_fichier] [taille_fichier] : Changer la taille d'un fichier

=> tar |destination|[nom_dossier] : Archiver un dossier

=> touch |destination|[nom_fichier] : Creer un fichier
================================================================================
Legende : |param| => parametre facultatif [param] => parametre obligatoire
```

- ## Quitter le SGF (quit)

Cette commande permet de **quitter le SGF**. Elle s'utilise avec le mot-clé **quit** et sans aucun paramètre.

*Exemple :*

```
> quit
Aurevoir :( ...
```

## Informations importantes

- A la création d'un fichier, sa taille est de **5 Octects**
- La taille du disque est de **1 Méga-Octects**, à partir de cette limite atteinte il vous sera **impossible** de créer un nouveau répertoire, copier un nouveau fichier ou d'augmenter la taille d'un fichier (des **exceptions** peuvent se produire dans ces cas).
- Si une commande du SGF est mal tapée, il vous demandera de la réécrire et il conseillera la commande help pour avoir davantage d'informations sur les commandes existantes.