

TME1 : Outils de développement de programmes C

1. Objectifs
2. Etape 1 : programme principal
3. Etape 2 : Analyse de la ligne de commande
4. Etape 3 : Analyse du fichier de commandes
5. Etape 4 :

Objectifs

L'objectif des deux premiers TMEs est de vous permettre de vérifier que vous maîtrisez les principaux outils de développement utilisés pour programmer efficacement en langage C sous UNIX. Attention : Cette U.E. suppose que vous connaissez le langage C. Si vous ne le connaissez pas, ou si vous avez des lacunes, vous devez vous former par vous même. Les bibliothèques sont faites pour cela, et vous pouvez utiliser les machines des salles de TP en libre service pour faire des exercices.

On utilisera les outils suivants:

- vim ou emacs : éditeur de texte
- indent : outil d'indentation automatique
- gcc : préprocesseur, compilateur, linker
- ar : constructeur d'archives
- make : génération automatique avec gestion des dépendances
- gdb : débogueur

L'application logicielle proposée est une application de manipulation d'images au format Pgm. Les algorithmes réalisent fondamentalement des parcours de tableaux à 2 dimensions.

Commencez par créer un répertoire "tme1", qui contiendra tous les fichiers utilisés dans ce TME.

Etape 1 : programme principal

Commencez par créer un sous-répertoire tme1/etape1 dans le répertoire tme1, et placez-vous dans ce répertoire. Dans cette première étape, vous allez devoir écrire deux fichiers : un fichier 'main.c' contenant la fonction main(), et un fichier 'Makefile' permettant de compiler le programme et de générer un fichier exécutable 'pgmg'.

Le fichier Makefile doit contenir deux règles :

```
pgmg : main.c
```

qui effectue la compilation ET l'édition de liens en une seule passe.

```
clean
```

qui efface les fichiers temporaires et un éventuel 'core', pour ne conserver que les fichiers sources.

Le programme 'main()' va lire le fichier 'input_file_name', en utilisant la fonction readpgm(), et recopier chaque octet dans un tampon intermédiaire que vous devez allouer. Il va ensuite recopier le contenu de ce tampon dans un fichier 'output_file_name', en utilisant la fonction writepgm(). Le prototype de la fonction main() est défini comme suit:

```
int main(int argc, char *argv[])
```

Vous trouverez les prototypes des fonctions de lecture et d'écriture du format pgm dans la page [PgmInputOutput](#).

L'édition de liens doit se faire avec la bibliothèque qui se trouve dans

```
/users/enseig/encadr/cao/lib/libpgmio.a
```

Un fichier contenant une image au format 'pgm' est disponible dans

```
/users/enseig/encadr/cao/tme1/Untitled.pgm
```

Ce fichier peut être visualisé en utilisant la commande

```
display Untitled.pgm
```

Etape 2 : Analyse de la ligne de commande

Commencez par créer un second sous-répertoire `tme1/etape2` dans votre répertoire `tme1`. Recopiez les fichiers `main.c` et `Makefile` de dans ce sous-répertoire. La seconde étape consiste à écrire un fichier `'getarg.c'`, qui contient la fonctions `getarg()`. La fonction `main()` ne fait plus l'analyse de la ligne de commande, cette tâche étant réalisée par la fonction `getarg()`. Le fichier `'getarg.c'` doit contenir aussi la déclaration de la variable globale `mainarg`, de type `'mainarg_t'`, qui contient les valeurs de tous les paramètres lus par la fonction `getarg()`.

Le fichier `'getarg.h'` doit contenir la définition du type `mainarg_t`, et la déclaration du prototype de la fonction `getarg()`:

```
typedef struct mainarg_t {
char    *InputFileName;
char    *OutputFileName;
char    *CommandFileName
char    verbose;
} mainarg_t;

extern mainarg_t  mainarg;

extern void getarg(int argc, char argv[]);
```

On souhaite que le prototype du programme `'pgmg'` soit le suivant:

```
pgmg [-c command_file_name] [-v] input_file_name output_file_name
```

Les deux arguments optionnels doivent pouvoir être placés n'importe où sur la ligne de commande.

- `-v` est un drapeau qui demande au programme d'afficher un message à chaque étape du traitement.
- `-c command_file_name` permet de définir une série de traitement à appliquer à l'image, dans un fichier de commandes.

On se contente dans cette étape d'analyser les arguments de la ligne de commande. L'analyse du des commandes contenues dans le fichier de commande sera réalisée à l'étape suivante.

Etape 3 : Analyse du fichier de commandes

Etape 4 :