esiea
INGÉNIEUR·E·S D'UN NUMÉRIQUE UTILE

Programmation système

LAB1
Prise en main de l'environnement

Table des matières

By Rexy

2 Objectifs

Ce LAB permet de mettre prendre en main l'environnement minimal pour aborder les notions de programmation système.

- Documents :
  - Supports : « Linux – Les commandes » et « Linux – L'éditeur Vi » ;
  - Fiches : « Compilation » et « Débogage ».
- Notions abordées : l'environnement de développement, le débogage.
- Commandes et fichiers exploités : gcc, commandes de gestion de fichiers, ddd.
- Travail à rendre : Vous devrez répondre directement à plusieurs questions au sein de ce document. Vous le copierez sur Moodle sous le nom : « LAB0_noms.odt ».

3 Rappels et mise en place de l'environnement

Les principes abordés dans ce document sont conformes aux préconisations « Standard Unix Specification V4 (SUSV4) ». Les distributions Linux, à travers la standardisation « Linux Standard Base (LSB) » sont conformes à « SUSV4 ».

Pour développer, vous devez utiliser les outils standards GNU (gcc + gdb/ddd) et un éditeur de texte type Vi/Emacs/Gedit, l'utilisation d'un IDE est interdite.

4 Travaux préparatoires

Pour vous familiariser à nouveau avec l'environnement :

- Lisez et effectuez les commandes décrites aux chapitres §1 à §3.7 du support « Linux - Les commandes.pdf » ;

5 Connaissance système

Objectif : Connaître le système sur lequel vous êtes en train de travailler : nom de la distribution, version du noyau, capacité de la machine (CPU, mémoire, carte mère et bios, cartes additionnelles), topologie du disque (marque, taille, partitionnement) ?

- Copiez ci-dessous les informations recherchées (les commandes utilisées et le résultat de celles-ci pour votre système).

CPU :

Taille mémoire vive :4go avec free -h

Carte mère et version du BIOS :lsb_release -a me permets de connaitre la version du bios manjaro linux 21.1.3

Cartes additionnelles :

Partitionnement du disque dur :30GB avec sfdisk -l -uM

- Comment lire le journal de démarrage du système (boot) ?
- Comment lire de manière continue le journal d'événement ?

Nous pouvons lire le journal de démarrage du system avec journalctl -list-boots

Pour le lire de manière continue nous pouvons utiliser la commande journalctl -f

- Ouvrez un terminal.

Dans quel répertoire vous trouvez-vous ?

Je me trouve dans le répertoire de mon utilisateur soit enzo

- Dans votre répertoire de connexion, créez un répertoire `tmp`
- Positionnez les droits d'accès à `rwx r-x ---` pour tmp
- Copiez les fichiers `passwd`, `group` et `hosts` du répertoire /`etc` dans `tmp`
- Changez le nom de `hosts` en `hotes`.
- Positionnez les droits d'accès à rw- r--- --- pour le fichier `hotes`. Lisez le contenu de `hotes`.

Remarque : la lecture du fichier `~/tmp/hotes` est permise. Le fichier peut néanmoins être vide.

- Retirez au propriétaire le droit en lecture sur le fichier `hotes` et essayez de le lire.

Quel est la signification du message d'erreur obtenu ?

Le message s'affichant est cat : hôtes Permission non accordée. Cela signifie que je n'ai pas la permission de lire ce fichier.

- Remettez pour le propriétaire le droit en lecture sur le fichier `hotes`.
- Retirez pour le propriétaire le droit en écriture sur le répertoire `tmp`.
- Essayez de détruire `hotes`.

Quel est la signification du message d'erreur obtenu ?

Le message d'erreur obtenu est rm : supprimer './hotes' qui est protégé en écriture et est du type « fichier »

Cela signifie que l'on ne peut pas supprimer le fichier si l'on ne possède pas le droit d'écriture.

- Retirez pour le propriétaire le droit en lecture sur le répertoire `tmp`.
- Essayez de lister le contenu de `tmp`.

Quel est la signification du message d'erreur obtenu ?

Le message d'erreur obtenu est ls : impossible d'ouvrir le répertoire « ./tmp» : permission non accordée

Cela signifie que l'on ne possède pas la permission de lire le contenu du répertoire tmp et de ce fait nous ne pouvons pas le lister.

- Lisez le contenu de `hôtes`.

Pourquoi pouvez-vous le lire ?

Nous pouvons le lire car nous nous trouvons déjà dans le répertoire.

- Retirez pour le propriétaire le droit en exécution sur le répertoire `tmp`.
- Essayez de vous positionner sur ce répertoire.
- Quel est la signification du message d'erreur obtenu ?

Le message d'erreur obtenu est cd : permission non accordée : ./tmp cela signifie que nous ne pouvons pas nous déplacer sur ce répertoire.

- Essayez de lire le contenu de `hotes`.

Quel est la signification du message d'erreur obtenu ?

Le message d'erreur obtenue est cat : ./tmp/hotes : permission non accordée cela signifie que nous ne pouvons pas lire le contenu de hotes.

6 Source, code assembleur, binaire et débogage.

- Lisez la fiche « rappel compilation » ;
- Réalisez le premier programme de la fiche `welcome.c` ;
- Lancez et analysez la compilation étape par étape comme décrite dans la fiche « rappel compilation ».

Copier ici, le **code assembleur** de ce programme.

Les commandes utilisées sont :

gcc -o welcome welcome.c

chmod u+x welcome

gcc -E welcome.c>welcome.i

gcc -S welcome.i

vi welcome.s

Voici le code assembleur :

```
.file   "welcome.c"
        .text
        .section        .rodata
.LC0:
        .string  "Rexy is welcome you"
        .text
        .globl   main
        .type    main, @function
main:
.LFB0:
        .cfi_startproc
        pushq  %rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
        movq   %rsp, %rbp
```

```
        .cfi_def_cfa_register 6
        leaq    .LC0(%rip), %rax
        movq    %rax, %rdi
        call    puts@PLT
        movl    $0, %eax
        popq    %rbp
        .cfi_def_cfa 7, 8
        ret
        .cfi_endproc
.LFE0:
        .size   main, .-main
        .ident  "GCC: (GNU) 11.1.0"
        .section        .note.GNU-stack,"",@progbits
```

Pour connaître le type d'un fichier (binaire, son, image, etc.), on peut se fier à son extension (.mp3, .jpeg, etc.). Cela est limité surtout quand le nom du fichier ne possède pas d'extension. La commande `file` sous Linux permet de connaître le type d'un fichier en faisant abstraction de son nom.

Quel type de fichier votre compilateur a-t-il généré ? Récupérez un fichier binaire exécutable pour Windows (Notepad++ par exemple) et testez la commande `file` avec celui-ci ?

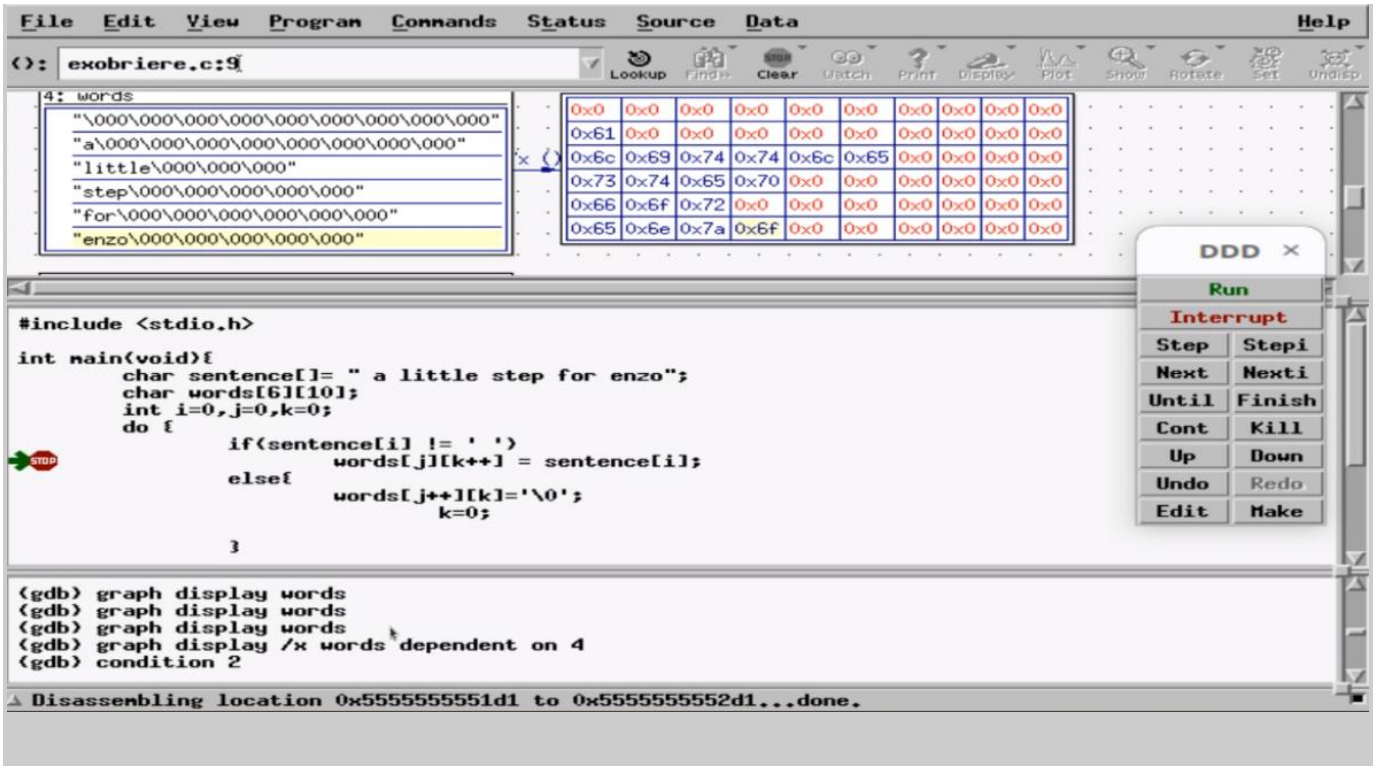
Type du fichier binaire Linux: welcome: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=ec40dc8b529f9a586b9172ff758ade5e2850a517, for GNU/Linux 4.4.0, not stripped

Type du fichier binaire Windows : commande :file notepad++.exe

Type : notepad++.exe: PE32+ executable (GUI) x86-64, for MS Windows

- Analysez et écrivez le deuxième programme `sentence2words.c` ;
- Lancez-le pas à pas dans le débogueur `ddd` pour suivre l'évolution du chargement du tableau de mots.

Réalisez une copie d'écran du débogueur affichant la zone mémoire du tableau rempli. Dans cette copie d'écran, on doit aussi voir la fenêtre de résultat du programme affichant le tableau de mots.

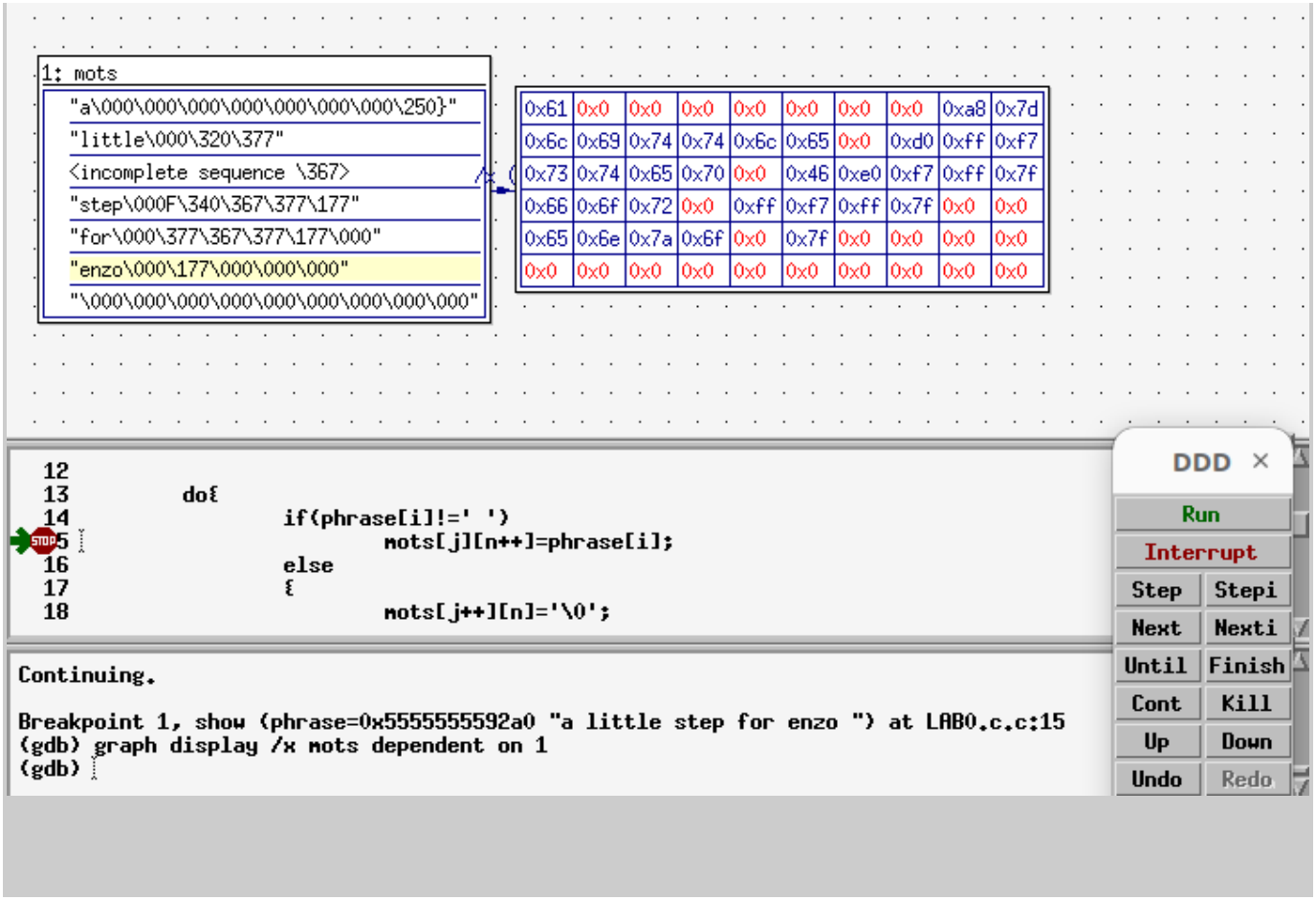


7 Évolution de code

Proposez l'évolution suivante du code du deuxième programme :

- Au lancement du programme, la phrase initiale est demandée à l'utilisateur ;
- Le tableau est dimensionné dynamiquement en fonction de cette phrase ;
- L'affichage du tableau est réalisé par l'appel d'une fonction.

Copier votre code sur Moodle sous le nom de fichier « LAB0.c ».
Rappel : un fichier source qui ne compile pas n'est pas corrigé !
Copiez ci-dessous une copie d'écran du résultat de son exécution



8 Amélioration de la fiche « débogage »

L'objectif est d'améliorer cette fiche, exploitez `ddd` sur un programme utilisant des fonctions et expliquez le rôle des fichiers `core`.

Copiez votre fiche « débogage » sur Moodle en même temps que ce LAB

Vérifiez que vous avez bien copié 3 fichiers sur Moodle : ce LAB, le code de l'évolution du programme `sentence2words.c` et votre fiche débogage.