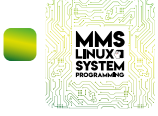
Системно програмиране за Линукс

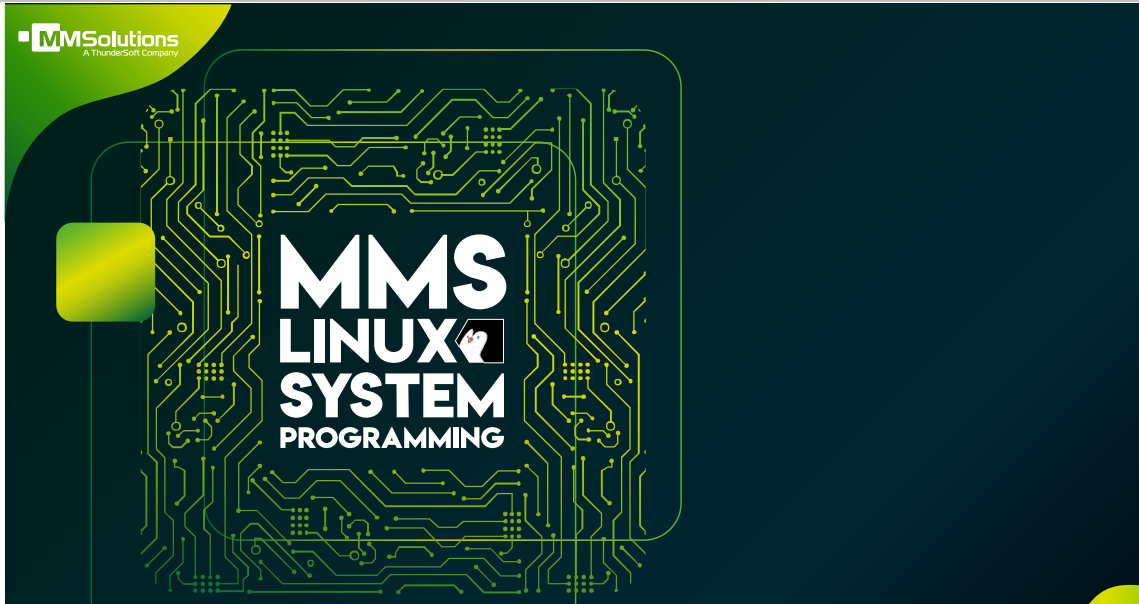
упражнения

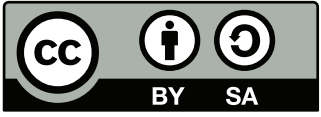
Димитър Димитров

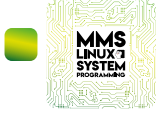
MMSolutions
A ThunderSoft Company

01.04.2021

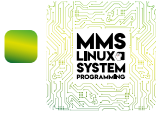



MMSolutions
A ThunderSoft Company
MMS
LINUX
SYSTEM
PROGRAMMING





Съдържание I

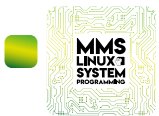


Съдържание II

- Pthreads. Разпаралеляване на примерна задача (quicksort). Демонстриране на механизми за синхронизация.
- Pipes. Обмяна на съобщения между процеси или нишки.
- Mmap. Оптимизирано боравене с файлове.
- Strace. Syscalls, ioctl.

6 Вграден Линукс

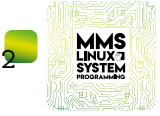
- GPIO, I2C - drivers in userspace.



Подготовка

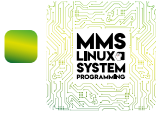
За курса ще са необходими:

- Добро владеене на езика C.
- Персонален компютър или лаптоп с инсталиран Линукс.
  - Възможно е стартиране на Линукс от USB флашка, без инсталация на основния диск.
  - `https://ubuntu.com/tutorials/try-ubuntu-before-you-install`
- Резервен вариант е да се ползва Линукс чрез локална или Web Browser виртуална машина.
  - `https://ubuntu.com/appliance/vm`
  - `https://bellard.org/jslinux/vm.html?url=alpine-x86.cfg&mem=192`
- Хъс.



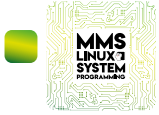
Относно този документ

Най-новата версия може да намерите на:
`https://github.com/MM-Solutions/lsp-course` .



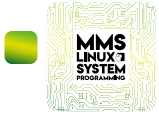
Терминал, shell

- Дистрибуция - що е то. Избор.
- Терминал.
- Команден ред.
  - `Ctrl+D`
  - Auto-complete
- Разходки из файловата система - `ls`, `cd`, `pwd`, `tree`.
- Текуща `.` и по-горна папка `..`
- Разделител в пътя: `/`
- `top`, `ps`.
- su, sudo.
- root и останалите потребители



Задача

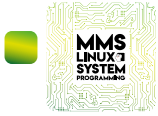
Отворете терминал и изпробвайте командите.



Текстов редактор

Пробвай и избери!

- vi ← Препоръчвам!
  - Има си самоучител: `vimtutor`
- gedit
- nano.
- emacs ← Любим редактор на множество хакери.
- Много, много други.



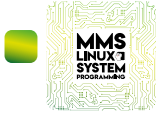
Изпълними файлове

- UNIX permissions: `chmod ugo+rwx MYFILE`

```
drwxr-xr-x
 \./\./\./
  |  |  '----␣Other
␣␣|␣␣'-------  Group
  '----------␣User␣owner
```

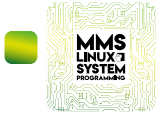
- ELF.
- Shell/Python/... scripts.
- Shebang.

```
#!/bin/sh
```



Помощ!

- `man`
- `man 1 kill` или `man 2 kill`
  - `man man`
- `apropos`
- `whatis`
- `help` - за вградени команди на shell

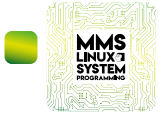


Задача

Създайте скрипт, който да:

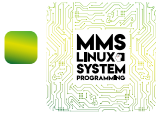
1. Разпечатва един ред с текущата директория.
2. След това разпечатва списък с файловете.
3. Списъкът да съдържа освен имената на файловете, и техните размери, собственици и права.

Съвет: Ползвайте `pwd`, `man ls`



Unix shell

- Променливи на обвивката (environment variables). `printenv`
- Променливи на средата (shell variables).
- `echo`
- `PATH`
- `cat`, `echo`, `less`
- `cp`, `mv`, `mkdir`
- `ln` - soft and hard.

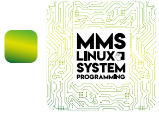


Задача

Направете мека и твърда връзка към скрипта от предишната задача.

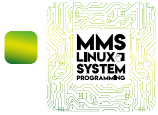
1. Какъв е резултата от `ls -l`
2. Разпечатайте (`cat`) съдържанието на трите файла.
3. Изтрийте оригиналния файл. Има ли разлика в `ls -l` ?
4. Има ли валидно съдържание в двете връзки (`cat`) ?

Съвет: Ползвайте `pwd`, `man ls`



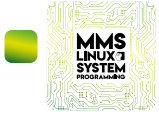
Многозадачност с Unix Shell

- Задачи в обвивката (shell) и процеси в OS.
- `bg`, `fg`, `&`, `wait`
- `Ctrl+Z` vs `Ctrl+C`
- `jobs`
- `kill`



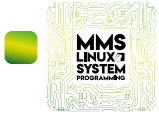
Задачи

- Изпробвайте приспиването на задачи, и поставянето им във фонов режим.
  - Съвет: Ако не се сещате за други опитни зайчета, ползвайте `cat` и `top`.
- Кой е номерът на задачата, и кой е номерът на процеса (PID)?
- Изпробвайте `wait` с помощта на няколко `sleep 10 &` задачи.
- Убийте един от вашите процеси. Как ще проверите дали е ”мъртъв”?



Тръби

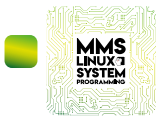
- Pipes.
- stderr, stdout, stdin.
- пренасочване чрез `>`, `>>`, `1>`, `2>`
- `tee`
- Филтри: `grep`, `sort`, `wc`, `cut`, `xxd`



Тръби - примери

Изпробвайте ги при вас!

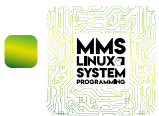
```
$ echo "Hello, world"
$ echo "Hello, world 1" > test.log
$ echo "Hello, world 2" > test.log
$ echo "Hello, world 3" >> test.log
$ echo "Aloha!" >> test.log
$ cat test.log
$ cat test.log | sort
$ cat test.log | xxd
$ cat /bin/ls | xxd | head -10 | tee A.log
$ cat A.log | cut -f2- -d":"
```



Тръби - примери

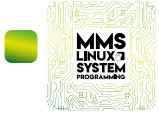
Изпробвайте ги при вас!

```
$ ls /bin | grep sh
$ ls /bin | grep sh | sort
$ ls /bin | sort > listing
$ cat listing
$ ls /epa-nema-takava-direktoria | sort > listing
$ cat listing
$ ls /epa-nema-takava-direktoria 2> listing
$ cat listing
```



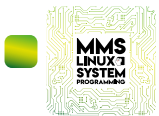
Всичко е файл!

- `/dev` : Device filesystem.
- `/proc` : Process filesystem.
- `/sys` : System filesystem.
- Даже и тръбите: `mkfifo`



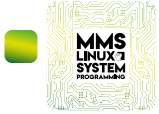
Задача

- Разузнайте `dev`, `/proc` и `/sys`.
  - Съвет: Припомнете си командите от предното упражнение: `ls`, `ls -l`, `cd`, `pwd`
  - Съвет: За списък на всички файлове: `find . -type f`
  - Съвет: За търсене по част от име на файл: `find /sys -name "*temp*"`
  - Съвет: За търсене по име на файл: `find /sys -name "temp"`
- Дали `cat` работи с тези файлове?
- Какви са странните директории наименувани с числа в `/proc` ?



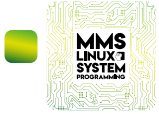
Задача

Напишете скрипт, който да разпечатва на екрана първите 160 байта от даден файл в HEX формат.
Съвет: Ползвайте `man xxd`, `head`



Shell scripts

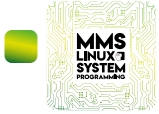
- Входни параметри - `$#`, `$@`, `$0`, `$1`, ...
- Unix SH е сложен език с много възможности ..... които няма да разглеждаме на тези упражнения.
- За допълнително четене: https://linuxcommand.org/tlcl.php



CRON

Изпълнение на наши задачи в указан час.

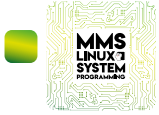
- `man crontab`



Задача

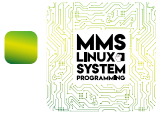
Направете си система за мониторинг на потребителите, които ползват вашата система. Нека всяка минута да се добавя текстова информация в `$HOME/users.log` с информация за датата, и кои потребители ползват системата.

- Съвет: Ползвайте `w` за списък на потребителите.
- Съвет: Ползвайте `date` за датата.



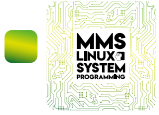
Компилатори

- GCC vs clang
- Крос компилатори - host vs target.
- `man gcc`



Фази

- Препроцесор.
  - Изход: C код
- C компилатор.
  - Изход: асемблерен код
- Асемблер.
  - Изход: обектен файл (ELF object).
- Свързващ редактор (linker).
  - Изход: ELF executable.

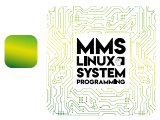


GCC - опции

- `-o OUTFILE` : в кой файл да се запише изхода.
- `-Os`, `-O0`, ... `-O3` : ниво на оптимизация.
- `-c` : да се компилира и асемблира, без свързване.
- `-g` : да се добави debug информация.
- `-Wall -Wextra` : да се предупреждава при съмнителен код.
- `-Werror` : да се излиза с грешка при съмнителен код.

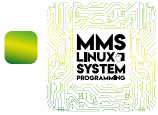
Пример:

```
$ gcc -O2 -Wall -Wextra test.c -o test
$ file test
```



Задача

- Напишете C програма за събиране на две числа, подадени от командния ред.
- Програмата да е разделена в два C файла - за `main()` и `calc_sum()` функциите.
- Да има header с декларация на `calc_sum()`
- Напишете скрипт, който компилира програмата.
- За по-напредналите: `objdump -d calc.elf | less`

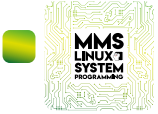


GNU Make

- GNU Make има множество алтернативи, но въпреки това е все още популярна.
- Позволява огромен контрол върху процеса на построяване (build).
- Правилно описани правила за построяване позволяват:
  - Автоматично разпаралелване.
  - Бързи и сигурни надстроявания (incremental builds).

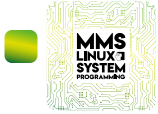
Запомнете:

```
# This is a comment.
target: dependencies
        commands
```



Задача

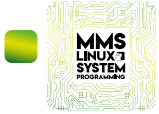
- Напишете `Makefile` за C програмата от предишното упражнение.
- За по-напредналите: Опитайте да направите своя `Makefile` по-универсален посредством променливи.
- За по-напредналите: Проследете exit code при успешно и при неуспешно построяване. Ползвайте специалната променлива на `sh` - `$?` .



GIT

- TODO.

```
$ git init
$ git clone https://github.com/MMS-Solutions/lsp-course
$ git add *
$ git commit
$ git push
$ git log
```

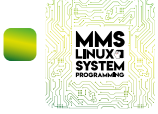


Задача

- Регистрирайте се в `github.com`, или `gitlab.com`, или `bitbucket.org`, или подобен.
- Направете си публично ГИТ хранилище (GIT project, GIT repo) с име ”c-hello-world”.
- Добавете С задачата от предишното упражнение към новото ГИТ хранилище, и я публикувайте.
- Добавете README.md с кратко описание, и го публикувайте като отделен GIT commit.
- За по-напредналите: свалете голям проект (например `https://github.com/torvalds/linux`), и изследвайте историята. Изпробвайте командата `gitk`.

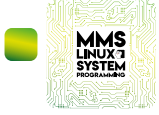
PThreads

- TODO.



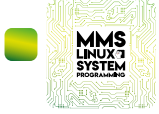
Pipes

- TODO.



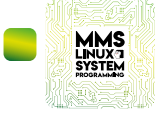
mmap

- TODO.



strace

- TODO.



Стандартни интерфейси към периферията

- TODO.