Høyskolen Kristiania

Denne forelesningsøkten vil bli tatt opp og lagt ut i emnet i etterkant.

<u>Hvis</u> du ikke vil være med på opptaket:

	La være å delta med webkameraet ditt.
	La være å delta med mikrofonen din.
	Still spørsmål i Chat i stedet for som lyd. Hvis du ønsker kan spørsmålet også sendes privat til foreleser.

Høyskolen
Kristiania

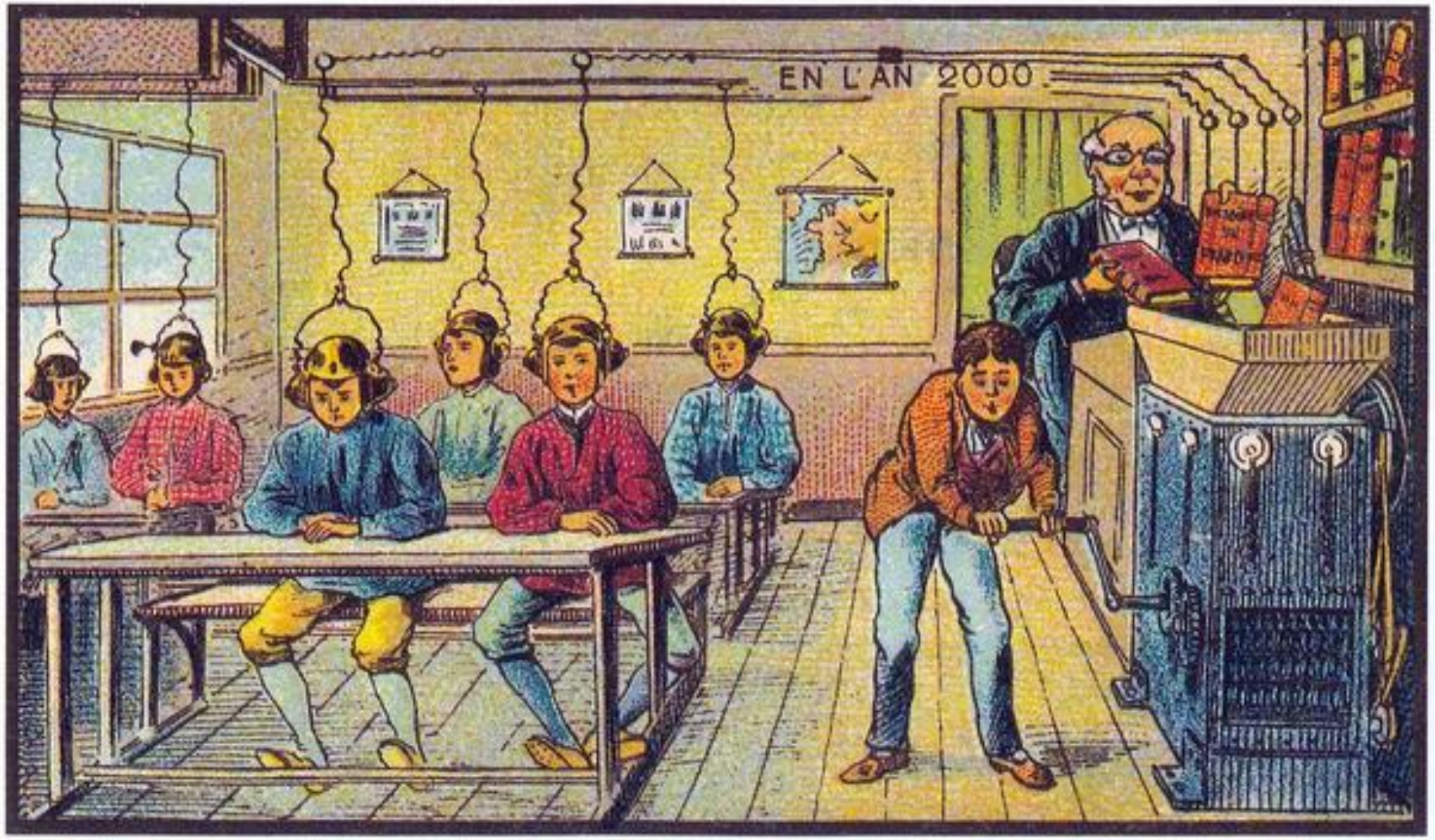
Høyskolen Kristiania

PG3400 Programmering i C for Linux

Lecture #2: Intro to Tools, OS tools, Compiling tools and Debugging tools

Bengt Østby

I apologize for the repetition and info-dump.



At School

Operating System

- Linux, Windows, OSx, iOs, SintranIII, MS-DOS…
    - File Management
    - Process Management
    - Processing
    - Self organization
    - Handling hardware

To be good at Linux you need to be good with terminal emulators.

Høyskolen Kristiania

“Shell” is way to interact with OS.

Most common:

- BASH [Bourne Again SHell]
- CSH [C SHell]
- GUI available but we live in denial!

Terminal navigation

Høyskolen Kristiania

“TAB” – smart completion

“UP Arrow/Down Arrow” – previous commands

“Enter/Return” – Submit the text [commands] into the shell

“Ctrl – R” – search history of commands

“Ctrl – R” – Keep searching!

Terminal Interaction

Høyskolen Kristiania

End command with & to start as separate process.

gedit griseenkelt.c &

Ctrl-C - to kill a program

Ctrl-Z – suspend a program

- Can be restarted with “fg” or “bg”

Highlight text with mouse

Auto clipboard

Middle click to paste

Everything is a file!

Høyskolen Kristiania

All resources are files

- I/O devices
- disks

Even directories

Unix File Structure

Høyskolen Kristiania

- Everything starts with ‘/’ – root folder
- Typically a user apart from root cannot do much in that folder
- The entire system is well-organized in folders in ‘/’
- (Virtual links)

Folders

/bin – Command line binaries
/boot – files needed for boot
/dev - devices, hardware
/etc – configuration files
/home – home directories for users (~/ == /home/hka/)
/lib – shared libraries for the entire system
/media – removable disks – could be in mnt
/mnt – mounted devices
/opt – optional stuff, used rarely
/proc – virtual files for processes and kernel info
/root – root user’s home
/sbin – system binaries
/tmp – cleared on every shutdown
/usr – has pretty much everything
/var – variable files printed and changed frequently

(links…)

man

Manual program

Provides documentation

“man PROGRAM” provides documentation about PROGRAM

- man cd
- man ls
- man gcc

Read man man ☺

cd

Change directory – “cd”

With full path

- cd /home/hka/westerdals

With user home

- cd ~/westerdals

“..” is parent directory

“.” is current directory

“cd –” switch to previous directory (like “back” in browser)

ls

List

- list all files in the directory

ls /home/ -> list files in /home

ls –l list all files in long format[a lot of info](ll in some systems)

ls –a list all files[including the hidden ones]

Hide a file with ‘.’ as prefix

(**dir** synonym for **ls**)

cat

concatenate files and print to standard output

Usage :

- “cat file1 file2” will print file1 and file2 next to each other
- “cat file1” will print the contents of file1

Usually good to join files by simply placing them together

Good to use with “split” for network

less/more

Both are based on vi

It just displays text

Navigation

- space – next page
- enter – next line
- page up – prev page
- Arrow keys up/down == one line up/down
- q - quit

File commands

Høyskolen Kristiania

mkdir – create directory

cp - copy

mv - move

rm - remove

rm

Høyskolen Kristiania

rm – remove

Usage :

- **rm file** – deletes file
- **rm –r dir** – deletes directory with a confirmation prompt
- **rm –f file** – skips the confirmation prompt
- **rm –rf** – Oops! Is not the correct reaction after asking for it

(You can go "**sudo rm –rf /**" - but you shouldn't unless you want to install your system again…)

tar

Høyskolen Kristiania

An archiver/extractor tool

Usage :

- **tar cf archive.tar files/directory**
- **tar xf archive.tar** untars
- **tar tvf archive.tar** lists files to terminal

c – create

v – verbose

x – extract

t – list files

f – indicates the next argument is a compressed file [usually the case]

z – compression and extraction using gzip

ps

Høyskolen Kristiania

ps – processes

- Prints a list of all processes
- Without any arguments prints only the ones in the current terminal

ps ax – prints other processes with their paths

ps ex – prints all parameters

pid is the process id – displayed in the first column

kill/ killall

kill pid – exit process with id pid

kill -9 pid – kill a process with id pid

killall name – kill all with matching name

(Make sure you kill the right processes, else you might disturb functionality of the system)

su/sudo

Super user – root or admin

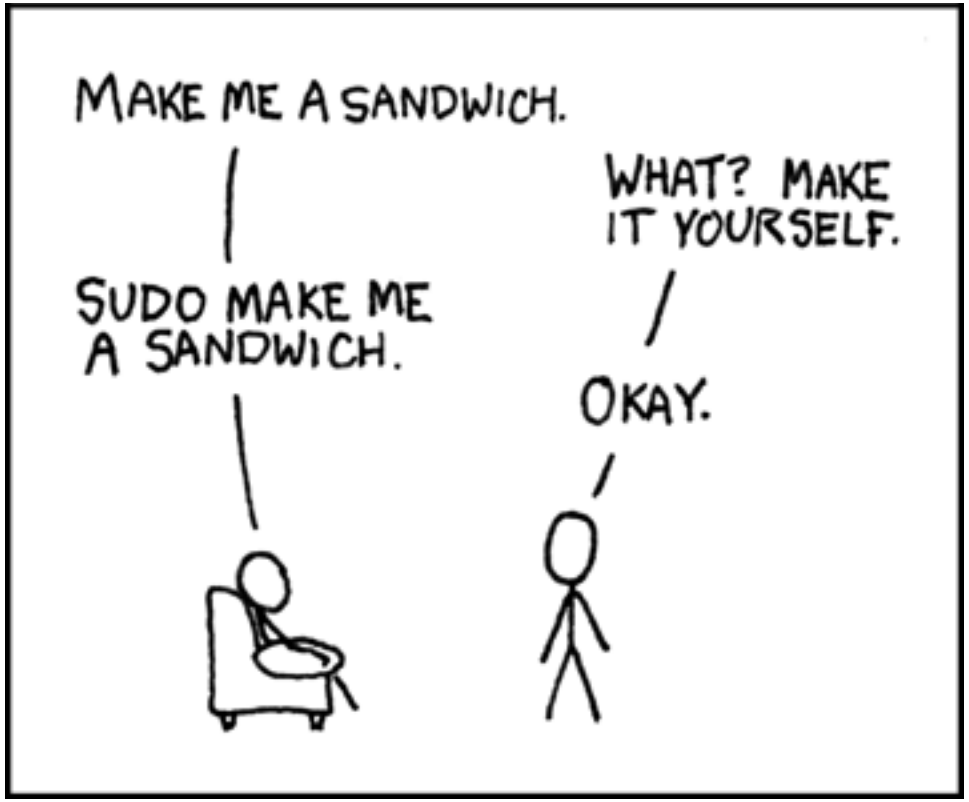
Type “su –” makes you login as root

sudo – “super user” + do

sudo – for just one command

“su” sometimes needs “sudo”.

(On our Ubuntu, su doesn’t work. Use **sudo –i**)



http://xkcd.com/149/

grep

grep – globally search a regular expression and print

Prints all matching lines

Regular expression – [aA]nt – searches for ant, Ant

Usage :

- grep regex file/files
- grep “hello” *.txt
- grep “[hH]ello” *.txt
- grep –C 10 “hello” *.txt
- grep –i “hello” *.txt

Permissions

Permissions are important in Unix based systems

Three sets:

- User
- Group
- Other

Three permissions per set

- Read- r
- Write - w
- Execute – x

Directory flag

chmod

Change the permissions

User: **u**(ser) **g**(roup) **o**(others) **a**(all)

Permissions:

'+' '-' '='

r (ead) w (rite) (e) x (e_ecute)

chmod uo +r *file*

chmod +x *nonexecutable*

Can also set permissions in octal.

chown

Change owner

Usage :

chown *owner file*

wget

Wget – web Get

It is useful to grab files across HTTP, FTP

Usage :

wget http://www.google.com
wget http://www.aspenberg.info/instruksjoner.txt
wget http://www.aspenberg.info/linux.txt

pipe

User symbol '|'

Sending output from one program as input to another:

ls - lR | less

- Sending file list for less, to see a page at a time

ls –l | grep “.txt”

- Prints lines containing the text “.txt“ (**ls –l *.txt**)

Can be combined in series:

ls - lR | grep “.txt” | less

nano / gedit

Høyskolen Kristiania

The most available editor on Unix platforms

It is simple, intuitive and not-so-powerful

Usage :

- **nano** *filename*

Ctrl o to save the file

Ctrl x to exit

(gedit is also quite usable…)

mount

"Mount" - sets up a directory to represent the contents of a device. Requires root, unless you want to mount is already in the "/etc/fstab"

mount <options> *device folder*

Mounts "device" in the "folder"

Common option is "-t" for filesystem type

Most mount is automatic in modern Linux

Filesystems

Høyskolen Kristiania

"Ext3" or "ext4": default linux

"Ntfs" newer Windows

"VFAT" older windows / dos. Often used on USB-sticks

"Smbfs" windows shares

"Iso9660" cd-roms

"Vboxsf" virtual box

Redirect < >

Uses symbols '<' and '>'

Sending data to or from file

ls –lR > file.txt

output of ls to file.txt

rm - rf / 2>&1>/dev/null

Sending both standard and error messages to the black hole (useful in scripts).

patch -p0 <diff.txt

Patch files with diff from diff.txt

Symbolic link

Windows has shortcuts

Links are much more *)

Link to a real file from a "virtual"

Most programs will read the file it points to

etc., cp, rm will only operate on the link

Usage :

ln –s *target link-name*

More commands

Høyskolen Kristiania

du – Disk usage

who – who is logged in

free – displays memory usage

top – displays Linux processes and the resources consumed

file – identify file.

Summary

echo “Linux is awesome!”

General usage of OS for development

Familiarity with using terminal

“man” is your friend

Scripting…

Summary

wget http://www.eastwillsecurity.com/pg3401/linux.txt

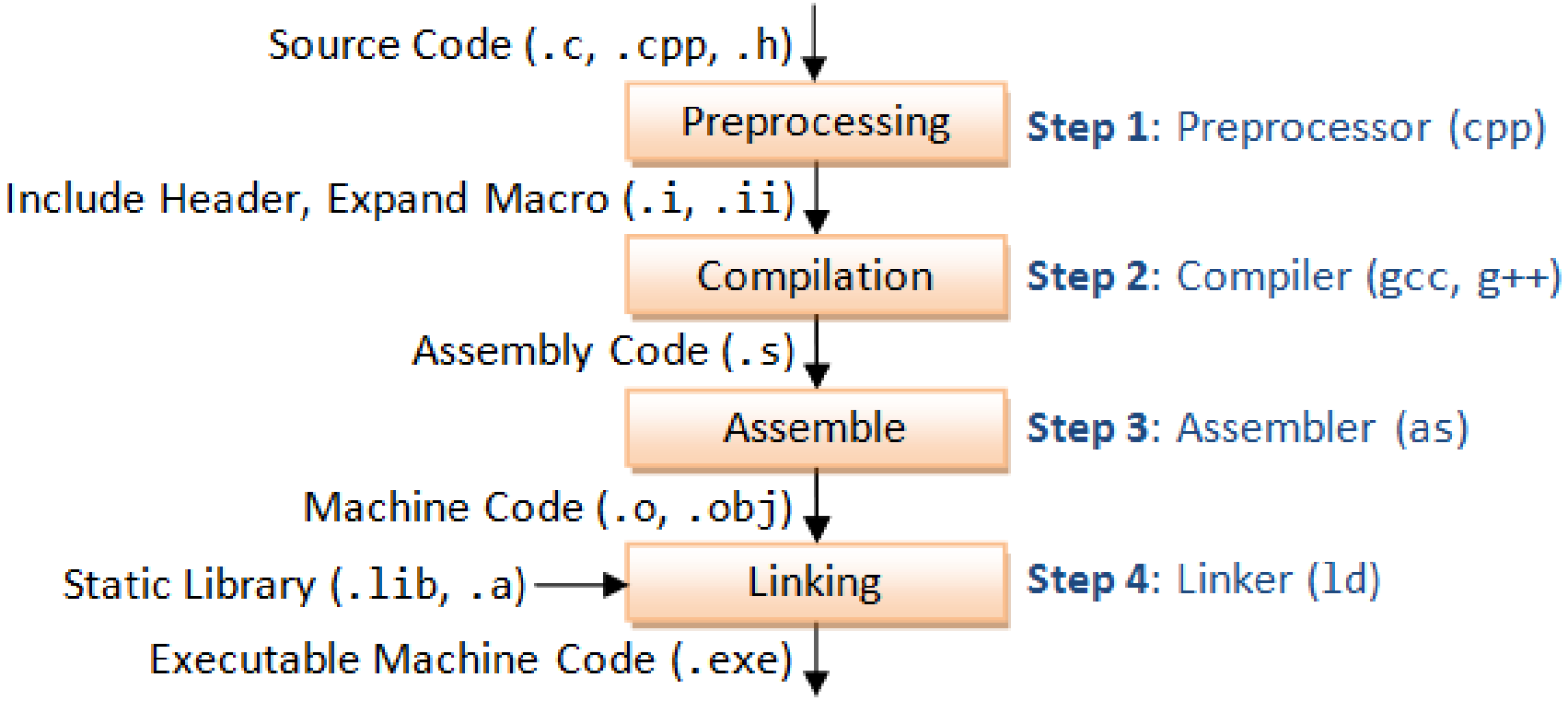
Work through it?

wget http://www.eastwillsecurity.com/pg3401/ linux_with_hints.txt

Høyskolen Kristiania

Compiling tools and Debugging tools

Life of C


Source Code (.c, .cpp, .h)
Preprocessing
Step 1: Preprocessor (cpp)
Include Header, Expand Macro (.i, .ii)
Compilation
Step 2: Compiler (gcc, g++)
Assembly Code (.s)
Assemble
Step 3: Assembler (as)
Machine Code (.o, .obj)
Static Library (.lib, .a)
Linking
Step 4: Linker (ld)
Executable Machine Code (.exe)

Editor

Høyskolen Kristiania

The first human input

Several editors:

- Vim
- Emacs
- Gedit
- Nano
- Notepad
- ...

Simple text files with extension .c/.h

Source code

Høyskolen Kristiania

```
#define MEMSIZE 10
int main(int argc, char *argv[])
{
        //My first buggy program
        printf("Entering main\n");
        int *array;
        int alpha, beta;
        array = (int*)malloc(MEMSIZE*sizeof(int));
        array[5] = array[2]+10;
        alpha = MEMSIZE + 20;
        beta = alpha + MEMSIZE + 30;
}
```

Preprocessor

Handles all preprocessing

Preprocessing:

- All includes – header, sources
- Macros

Outputs text file

Ex : gcc –E in.c –o out.c

After Preprocessing

Høyskolen Kristiania

```
# 1 "test.c"
# 1 "<command-line>"
# 1 "/usr/include/stdc-predef.h" 1 3 4
# 30 "/usr/include/stdc-predef.h" 3 4
# 1 "/usr/include/x86_64-linux-gnu/bits/predefs.h" 1 3 4
# 31 "/usr/include/stdc-predef.h" 2 3 4
# 1 "<command-line>" 2
# 1 "test.c"


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

  printf("Entering main\n");
  int *array;
  int alpha, beta;
  array = (int *)malloc(10*sizeof(int));
  array[5] = array[2]+10;
  alpha = 10 + 20;
  beta = alpha + 10 + 30;
}
```

Compiler

Conversion from Source to assembly code

Compiler optimizations

Ex : **gcc –O0 –S in.c –o out.s**

With optimization : **gcc –O2 –S in.c –o out.s**

After Compiling- no opt

Høyskolen Kristiania

```
        .file       "pre.c"
        .section    .rodata
.LC0:
        .string     "Entering main"
        .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
        subq        $32, %rsp
        movl        %edi, -20(%rbp)
        movq        %rsi, -32(%rbp)
        movl        $.LC0, %edi
        call        puts
        movl        $40, %edi
        call        malloc
        call        malloc
        movq        %rax, -8(%rbp)
        movq        -8(%rbp), %rax
        addq        $20, %rax
        movq        -8(%rbp), %rdx
        addq        $8, %rdx
        movl        (%rdx), %edx
        addl        $10, %edx
        movl        %edx, (%rax)
        movl        $30, -16(%rbp)
        movl        -16(%rbp), %eax
        addl        $40, %eax
        movl        %eax, -12(%rbp)
        leave
        .cfi_def_cfa 7, 8
        ret
        .cfi_endproc
.LFE0:
        .size       main, .-main
        .ident      "GCC: (Ubuntu/Linaro 4.8.1-10ubuntu9) 4.8.1"
        .section    .note.GNU-stack,"",@progbits
```

After Compiling - Opt

Høyskolen Kristiania

```
        .file   "pre.c"
        .section        .rodata.str1.1,"aMS",@progbits,1
.LC0:
        .string "Entering main"
        .section        .text.startup,"ax",@progbits
        .p2align 4,,15
        .globl  main
        .type   main, @function
main:
.LFB0:
        .cfi_startproc
        movl    $.LC0, %edi
        jmp     puts
        .cfi_endproc
.LFE0:
        .size   main, .-main
        .ident  "GCC: (Ubuntu/Linaro 4.8.1-10ubuntu9) 4.8.1"
        .section        .note.GNU-stack,"",@progbits
```

Assembler

Høyskolen Kristiania

Convert Assembly code to binary code

Usually called object file

Non-executable

Contains references to external libraries

Output not quite human friendly

Ex: as in.s –o out.o

Needing to code in assembler is very rare today; x86 assembler can be used for some special protected calls in kernel drivers, and proprietary assembler might be needed for some (rare) embedded programming.

But this is 2020, so we can safely learn a programming language from 1989.

Linker

Links the references to external libraries

External functions: printf, malloc

Ex: ld, collect2

Ex: ld –o executable objectfile –all_libraries

In summary

Høyskolen Kristiania

```
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/4.8/lto-wrapper
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu/Linaro 4.8.1-10ubuntu9' --with-bugurl=file:///usr/share/doc/gcc-4.8/README.Bugs --enable-languages=c,c++,java,go,d,fortran,objc,
obj-c++ --prefix=/usr --program-suffix=-4.8 --enable-shared --enable-linker-build-id --libexecdir=/usr/lib --without-included-gettext --enable-threads=posix --with-gxx-include-dir=/usr/includ
e/c++/4.8 --libdir=/usr/lib --enable-nls --with-sysroot=/ --enable-clocale=gnu --enable-libstdcxx-debug --enable-libstdcxx-time=yes --enable-gnu-unique-object --enable-plugin --with-system-zl
ib --disable-browser-plugin --enable-java-awt=gtk --enable-gtk-cairo --with-java-home=/usr/lib/jvm/java-1.5.0-gcj-4.8-amd64/jre --enable-java-home --with-jvm-root-dir=/usr/lib/jvm/java-1.5.0-
gcj-4.8-amd64 --with-jvm-jar-dir=/usr/lib/jvm-exports/java-1.5.0-gcj-4.8-amd64 --with-arch-directory=amd64 --with-ecj-jar=/usr/share/java/eclipse-ecj.jar --enable-objc-gc --enable-multiarch -
-disable-werror --with-arch-32=i686 --with-abi=m64 --with-multilib-list=m32,m64,mx32 --with-tune=generic --enable-checking=release --build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=x8
6_64-linux-gnu
Thread model: posix
gcc version 4.8.1 (Ubuntu/Linaro 4.8.1-10ubuntu9)
COLLECT_GCC_OPTIONS='-v' '-O2' '-o' 'test' '-mtune=generic' '-march=x86-64'
 /usr/lib/gcc/x86_64-linux-gnu/4.8/cc1 -quiet -v -imultiarch x86_64-linux-gnu test.c -quiet -dumpbase test.c -mtune=generic -march=x86-64 -auxbase test -O2 -version -fstack-protector -Wformat
 -Wformat-security -o /tmp/ccLJWfWn.s
GNU C (Ubuntu/Linaro 4.8.1-10ubuntu9) version 4.8.1 (x86_64-linux-gnu)
        compiled by GNU C version 4.8.1, GMP version 5.1.2, MPFR version 3.1.1-p2, MPC version 1.0.1
GGC heuristics: --param ggc-min-expand=100 --param ggc-min-heapsize=131072
ignoring nonexistent directory "/usr/local/include/x86_64-linux-gnu"
ignoring nonexistent directory "/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../x86_64-linux-gnu/include"
#include "..." search starts here:
#include <...> search starts here:
 /usr/lib/gcc/x86_64-linux-gnu/4.8/include
 /usr/local/include
 /usr/lib/gcc/x86_64-linux-gnu/4.8/include-fixed
 /usr/include/x86_64-linux-gnu
 /usr/include
End of search list.
GNU C (Ubuntu/Linaro 4.8.1-10ubuntu9) version 4.8.1 (x86_64-linux-gnu)
        compiled by GNU C version 4.8.1, GMP version 5.1.2, MPFR version 3.1.1-p2, MPC version 1.0.1
GGC heuristics: --param ggc-min-expand=100 --param ggc-min-heapsize=131072
Compiler executable checksum: b946adc090b5605f865521be8b22959c
test.c: In function ‘main’:
test.c:5:3: warning: incompatible implicit declaration of built-in function ‘printf’ [enabled by default]
   printf("Entering main\n");
   ^
test.c:8:18: warning: incompatible implicit declaration of built-in function ‘malloc’ [enabled by default]
   array = (int *)malloc(MEMSIZE*sizeof(int));
                  ^
COLLECT_GCC_OPTIONS='-v' '-O2' '-o' 'test' '-mtune=generic' '-march=x86-64'
 as -v --64 -o /tmp/cc3o1HBq.o /tmp/ccLJWfWn.s
GNU assembler version 2.23.52 (x86_64-linux-gnu) using BFD version (GNU Binutils for Ubuntu) 2.23.52.20130913
COMPILER_PATH=/usr/lib/gcc/x86_64-linux-gnu/4.8/:/usr/lib/gcc/x86_64-linux-gnu/4.8/:/usr/lib/gcc/x86_64-linux-gnu/:/usr/lib/gcc/x86_64-linux-gnu/4.8/:/usr/lib/gcc/x86_64-linux-gnu/
LIBRARY_PATH=/usr/local/cuda-5.5/lib64/../lib/:/usr/lib/gcc/x86_64-linux-gnu/4.8/:/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../x86_64-linux-gnu/:/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../li
b/:/lib/x86_64-linux-gnu/:/lib/../lib/:/usr/lib/x86_64-linux-gnu/:/usr/lib/../lib/:/usr/local/cuda-5.5/lib64/:./:/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../:/lib/:/usr/lib/
COLLECT_GCC_OPTIONS='-v' '-O2' '-o' 'test' '-mtune=generic' '-march=x86-64'
 /usr/lib/gcc/x86_64-linux-gnu/4.8/collect2 --sysroot=/ --build-id --eh-frame-hdr -m elf_x86_64 --hash-style=gnu --as-needed -dynamic-linker /lib64/ld-linux-x86-64.so.2 -z relro -o test /usr/
lib/gcc/x86_64-linux-gnu/4.8/../../../x86_64-linux-gnu/crt1.o /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../x86_64-linux-gnu/crti.o /usr/lib/gcc/x86_64-linux-gnu/4.8/crtbegin.o -L/usr/local/cuda
-5.5/lib64/../lib -L/usr/lib/gcc/x86_64-linux-gnu/4.8 -L/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../x86_64-linux-gnu -L/usr/lib/gcc/x86_64-linux-gnu/4.8/../../../../lib -L/lib/x86_64-linux-gnu
 -L/lib/../lib -L/usr/lib/x86_64-linux-gnu -L/usr/lib/../lib -L/usr/local/cuda-5.5/lib64 -L. -L/usr/lib/gcc/x86_64-linux-gnu/4.8/../../.. /tmp/cc3o1HBq.o -lgcc --as-needed -lgcc_s --no-as-nee
ded -lc -lgcc --as-needed -lgcc_s --no-as-needed /usr/lib/gcc/x86_64-linux-gnu/4.8/crtend.o /usr/lib/gcc/x86_64-linux-gnu/4.8/../../../x86_64-linux-gnu/crtn.o
```

Or

Høyskolen Kristiania

gcc –O2 griseenkelt.c –o griseenkelt

“make” your life easier

Building tools…

make

cmake – Not strictly building tool - Powerful

Make

Høyskolen Kristiania

Classic and still widely used for Unix based platforms (and Windows!)

gcc –O2 griseenkelt.c –o griseenkelt

```
#
# Simple makefile for compiling and linking one single source file
#

griseenkelt : griseenkelt.c
        gcc -O2 griseenkelt.c -o griseenkelt
```

Make

Høyskolen Kristiania

```
#
# Simple makefile for compiling and linking one single source file
#

TARGET = griseenkelt

$(TARGET) : $(TARGET).c
        gcc -O2 $^ -o $@
```

Make

Høyskolen Kristiania

A bit more complex …

```
#
# Generic makefile for compiling and linking more than one source file
#

OBJS = hello.o number.o # List object files here
# DEPS = number.h
CFLAGS = -O2

%.o: %.c $(DEPS)
        gcc -c -o $@ $< $(CFLAGS)

hello: $(OBJS)
        gcc -o $@ $^ $(CFLAGS)

.phony: clean
clean:
        rm -f *.o
```

Make

```
INCLDIR = ./include
CC = gcc
CFLAGS = -O2
CFLAGS += -I$(INCLDIR)

OBJDIR = obj

_DEPS = number.h
DEPS = $(patsubst %,$(INCLDIR)/%,$(_DEPS))

_OBJS = hello.o number.o
OBJS = $(patsubst %,$(OBJDIR)/%,$(_OBJS))


$(OBJDIR)/%.o: %.c $(DEPS)
        $(CC) -c -o $@ $< $(CFLAGS)

hello: $(OBJS)
        gcc -o $@ $^ $(CFLAGS)

.PHONY: clean
clean:
        rm -f $(OBJDIR)/*.o *~ core $(INCLDIR)/*~
```

Other useful tools

Gdb – A powerful debugging tool

Valgrind – Powerful memory tool

Make sure you compile with –O0[No optimization] and –g option for better results from these tools

GDB

Høyskolen Kristiania

GNU Debugger

Useful to track the errors

Controlled run

Display memories

GDB

Høyskolen Kristiania

To run the debugger

- $ gdb *filename* (or simply) gdb

Inside gdb environment

- run (or) r to run the program
- start / step / stepi
- bt – back trace the stack
- print/info/display – to see useful information

GDB output

Høyskolen Kristiania

```
Starting program: /home/vamsidhar/pg3400/./a.out

Program received signal SIGSEGV, Segmentation fault.
0x00000000004005d6 in main (argc=1, argv=0x7fffffffde08) at testdb.c:11
11          a[10] = 'a';
```

Valgrind

Høyskolen Kristiania

Memory debugging

Memory leaks

Uninitialized memory

Reading/writing to invalid memory

- Freed memory
- Out of allocated blocks

Valgrind

Høyskolen Kristiania

To use valgrind:

valgrind --leak-check=yes --track-origins=yes *filename*

Overwhelming information!!

Valgrind output

Høyskolen Kristiania

```
==6461== Memcheck, a memory error detector
==6461== Copyright (C) 2002-2012, and GNU GPL'd, by Julian Seward et al.
==6461== Using Valgrind-3.8.1 and LibVEX; rerun with -h for copyright info
==6461== Command: ./a.out
==6461==
==6461== Invalid write of size 4
==6461==    at 0x4005B7: main (testdb.c:8)
==6461==  Address 0x51fc1d0 is not stack'd, malloc'd or (recently) free'd
==6461==
==6461== Invalid read of size 4
==6461==    at 0x4005BD: main (testdb.c:9)
==6461==  Address 0x51fc1d0 is not stack'd, malloc'd or (recently) free'd
==6461==
alpha is 0
==6461==
==6461== HEAP SUMMARY:
==6461==     in use at exit: 40 bytes in 1 blocks
==6461==   total heap usage: 1 allocs, 0 frees, 40 bytes allocated
==6461==
==6461== LEAK SUMMARY:
==6461==    definitely lost: 40 bytes in 1 blocks
==6461==    indirectly lost: 0 bytes in 0 blocks
==6461==      possibly lost: 0 bytes in 0 blocks
==6461==    still reachable: 0 bytes in 0 blocks
==6461==         suppressed: 0 bytes in 0 blocks
==6461== Rerun with --leak-check=full to see details of leaked memory
==6461==
==6461== For counts of detected and suppressed errors, rerun with: -v
==6461== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 2 from 2)
```

Summary

Compilation tools : gcc

- Makefile, Cmake

Debugging tools

- gdb – execution bugs
- valgrind – memory bugs

Practical

Practical – Tools :

wget http://www.eastwillsecurity.com/pg3401/tools.txt

Follow the instructions

Lab exercises

Høyskolen Kristiania

Help each other becoming used to Linux. Some of you might have used it before, some are new to this OS. Make sure you all help each other – this is one of the most difficult courses you will take at this school, everyone of you will need help from the class! :-) You must attend Campus trainings, trying to learn it all your self will most often fail, hard…

After THIS week everyone in the class MUST have a running Linux environment and be able to edit and compile code.

Test the two previous files, also download PG3401-H21-Exercises-Lection2.zip from Canvas.

Final exercise: Create your own program that when run writes “HELLO WORLD” on the screen. (Hint: Use the function “printf”.)