Smith College

Computer Science

CSC231-Assembly

Week #2

Dominique Thiébaut
dthiebaut@smith.edu

Outline

- Labs: emacs, assembly
- Assembly + Linking Process
- Object files
- DB directive: *Everything is a byte*!
- HexDump: we need to learn hexadecimal!
- Ascii Table

Emacs

Lab #1: Emacs

- ssh to aurora

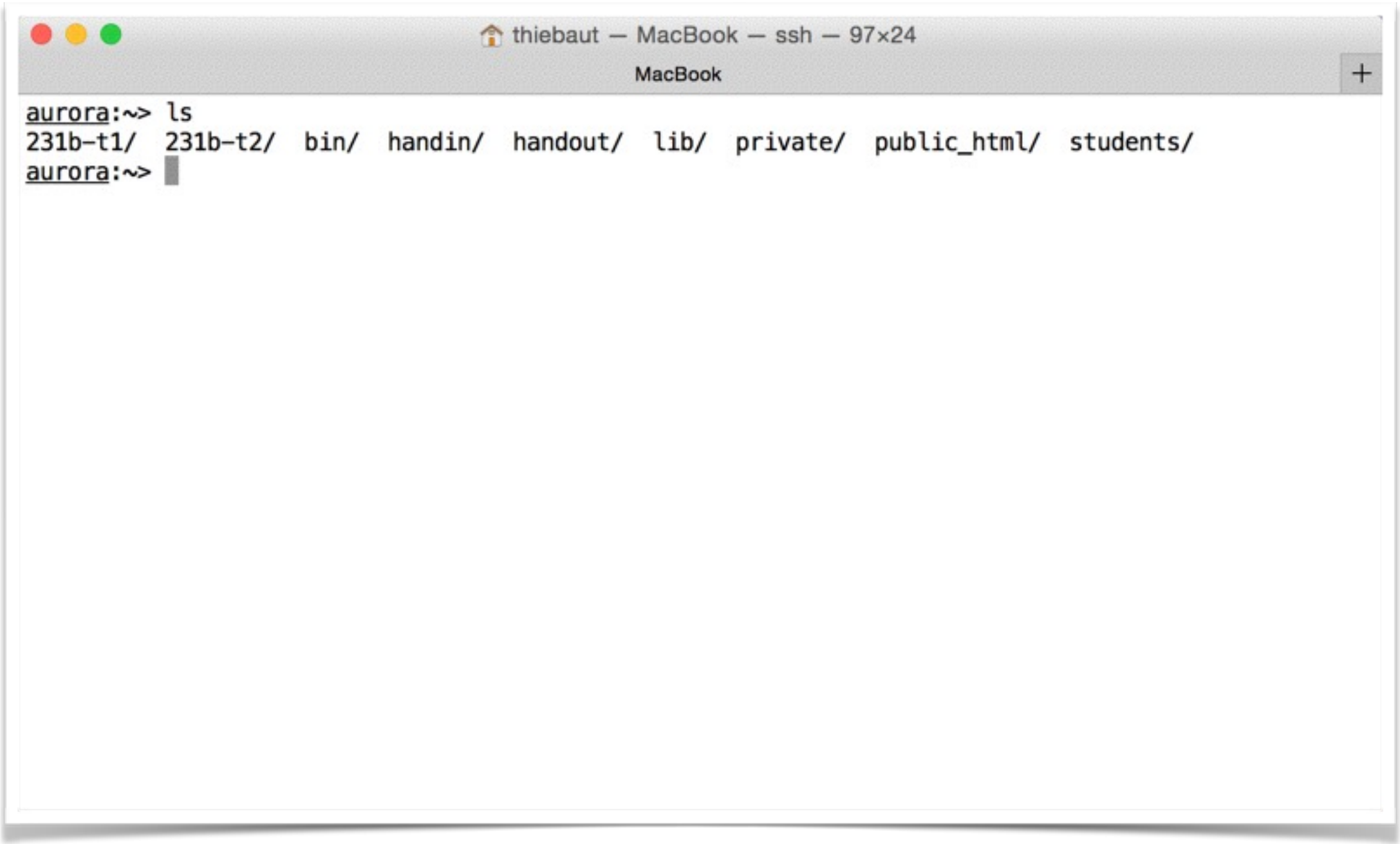
ssh -Y 231b-xx@aurora.smith.edu

- Emacs demo

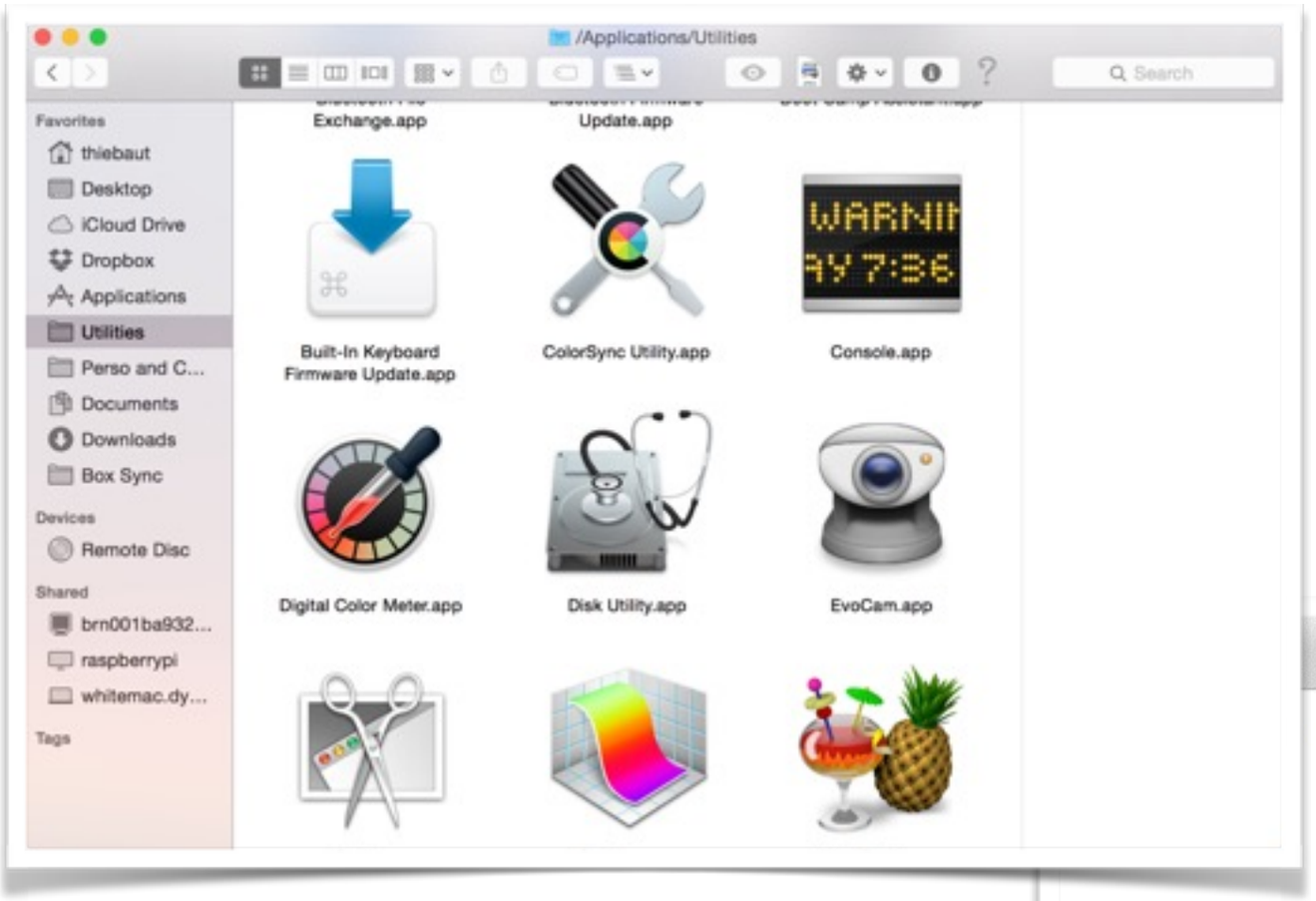
- Basic commands

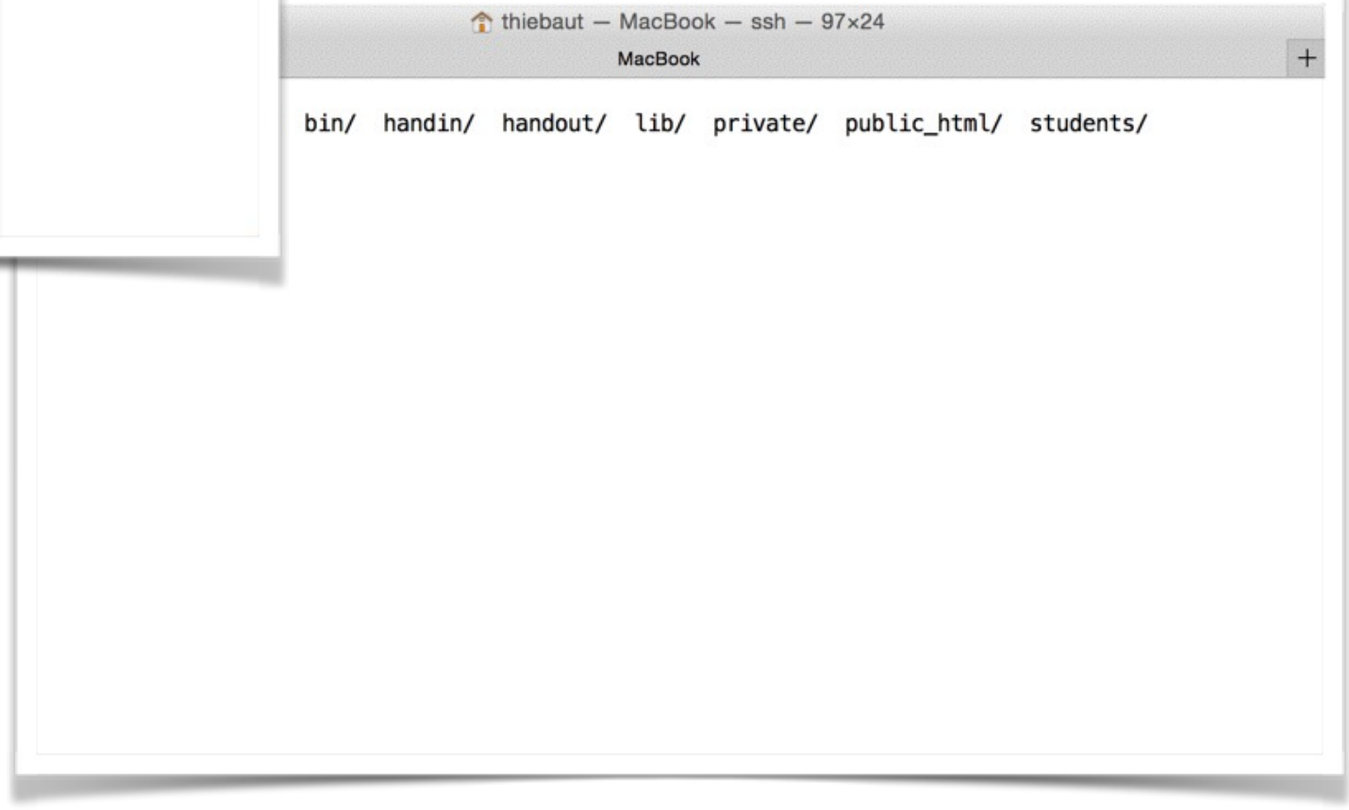
^X^C
^G
^D
^K
^Y
cursor keys

Shell


thiebaut — MacBook — ssh — 97×24
MacBook
aurora:~> ls
231b-t1/ 231b-t2/ bin/ handin/ handout/ lib/ private/ public_html/ students/
aurora:~>

Shell


/Applications/Utilities
Search
Favorites
thiebaut
Desktop
iCloud Drive
Dropbox
Applications
Utilities
Perso and C...
Documents
Downloads
Box Sync
Devices
Remote Disc
Shared
brn001ba932...
raspberrypi
whitemac.dy...
Tags
Exchange.app
Update.app
Built-In Keyboard Firmware Update.app
ColorSync Utility.app
Console.app
Digital Color Meter.app
Disk Utility.app
EvoCam.app


thiebaut — MacBook — ssh — 97×24
MacBook
bin/ handin/ handout/ lib/ private/ public_html/ students/

3 Useful Linux Commands

- **ls** (ell ess) list all files in a folder/**directory**
- **cp** copy file
- **cd** change directory

Linux Commands

command -switches file-name(s)

Linux Commands

command -switches file-name(s)

```
ls -l
ls *.asm
ls *.o
ls hello*
ls -l

man ls
```


LAB TIME!

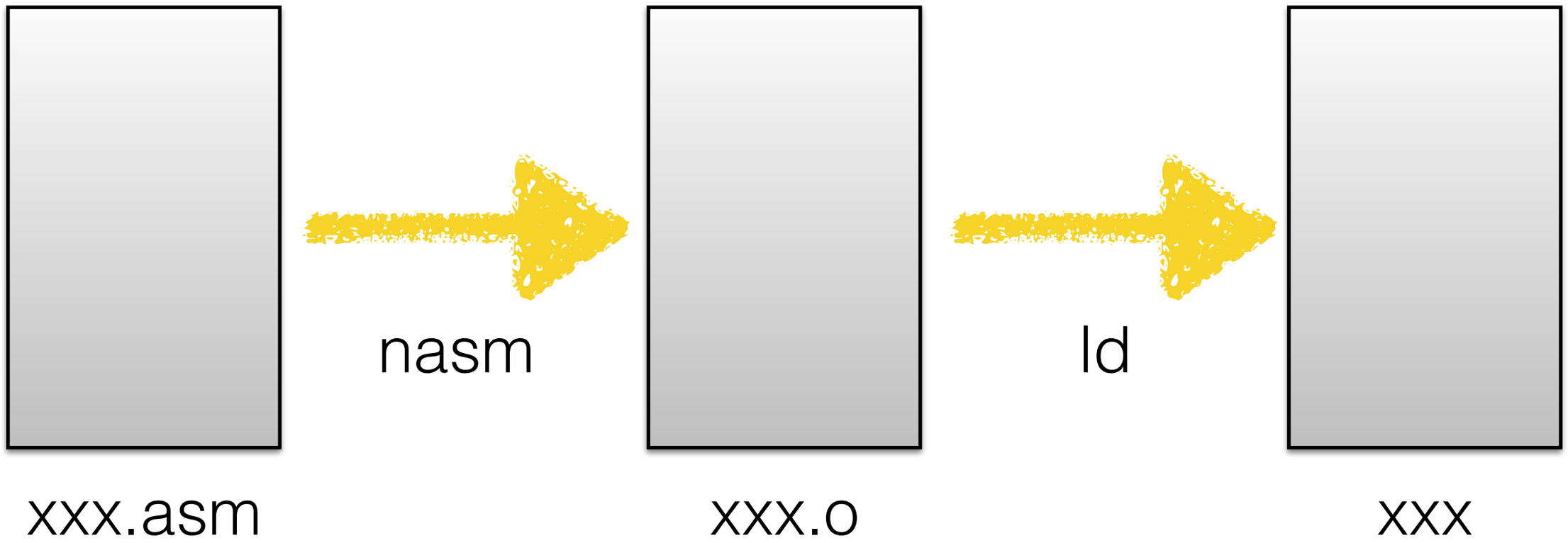
Skeleton

```
;;; program_name.asm
;;; your name
;;;
;;; a description of the program
;;;
;;; to assemble and run:
;;;
;;;     nasm -f elf -F  stabs program.asm
;;;     ld -melf_i386 -o program program.o
;;;     ./program
;;; ------------------------------------------------------------------

;%include files here...



        ;; -----------------------------------------------------------
        ;; data areas
        ;; -----------------------------------------------------------

        section .data


        ;; -----------------------------------------------------------
        ;; code area
        ;; -----------------------------------------------------------

        section .text
        global  _start

_start:


        ;; (add your code here!!!!)



        ;; exit()

        mov     eax,1
        mov     ebx,0
        int     0x80            ; final system call
```

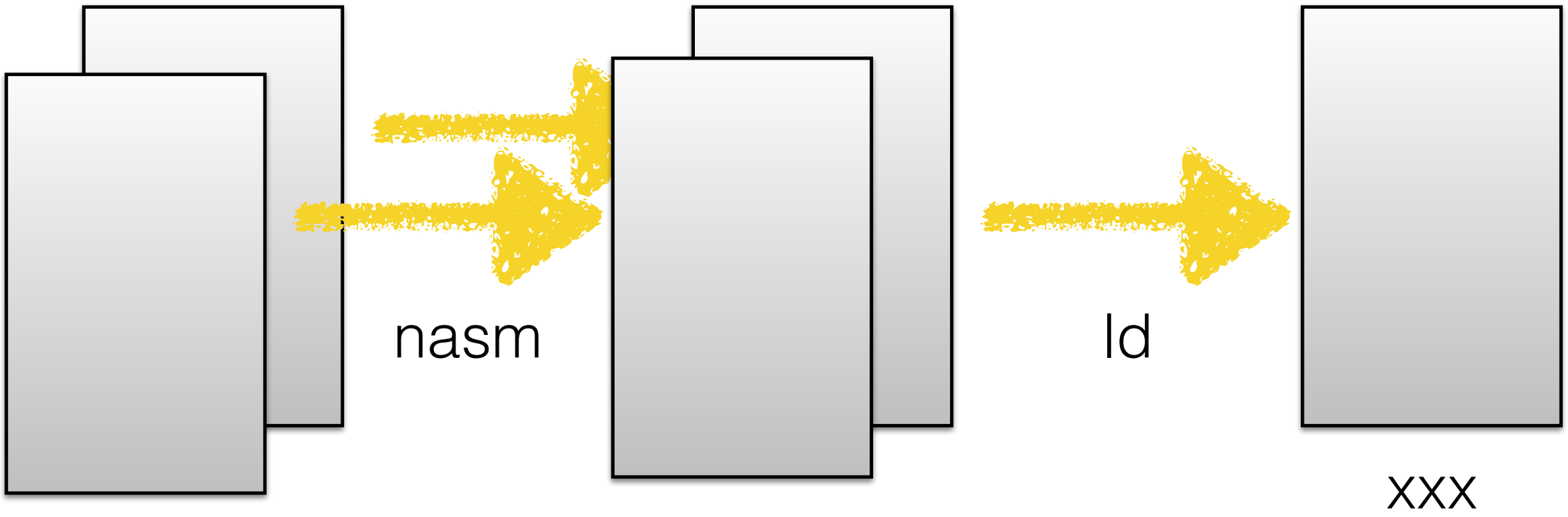
Hello World!

```
;;; helloWorld.asm
;;; D. Thiebaut
;;;
;;; Display "Hello there!" on the screen
;;;
;;; To assemble, link, and run:
;;;     nasm -f elf  helloWorld.asm
;;;     ld -melf_i386 -o helloWorld helloWorld.o
;;;     ./helloWorld
;;;

                section .data
Hello           db      "Hello there!"
HelloLen        equ     $-Hello

                section .text
                global  _start
_start:

;;; print message

                mov     eax, 4          ; write
                mov     ebx, 1          ; stdout
                mov     ecx, Hello      ; address of message to print
                mov     edx, HelloLen   ; # of chars to print
                int     0x80

;;; exit

                mov     ebx, 0
                mov     eax, 1
                int     0x80
```

Assembly+Linking



Assembly+Linking



DB: byte storage

```
                    section .data
Hello               db      "Hello there!"
HelloLen            equ     $-Hello
```

- A byte is…

- DB: "**D**efine **B**yte of storage"
  stores information in a byte format
  (could be more than 1 byte)

- **The X86 memory is a memory of bytes**

Data Section vs Memory

```
message     db      "hello",10
```

We stopped here last time…

Data Section vs Memory

```
message     db      "hello"
            db      10
```

Data Section vs Memory

```
message     db      "hel"
            db      "lo"
            db      10
```

Data Section vs Memory

```
message     db      104,"el"
            db      "lo"
            db      10
```

Exercise

```
                section .data
msg1        db          "lo "
msg2        db          "hel"
msg3        db          "world!", 10
msgLen      equ$-msg3
```

```
_start:
    mov  eax, 4    ; print
    mov  ebx, 1    ; to stdout
    mov  ecx, msg2   ; string
    mov  edx, 3
    int  0x80

    mov  eax, 4    ; print
    mov  ebx, 1    ; to stdout
    mov  ecx, msg1   ; string
    mov  edx, 3      ; # of chars
    int  0x80

    mov  eax, 4    ; print
    mov  ebx, 1    ; to stdout
    mov  ecx, msg3   ; string
    mov  edx, msgLen; # of chars
    int  0x80   ; ask Linux to print
```

TIME FOR THE LAB ON PRINTING STRINGS

Hexdump

Hex dump

Hexadecimal

Brief Overview of Hexadecimal

- Number system
- 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- 0 —--> 00
  10 —--> 0A
  255 —--> FF

Why Hex?

00000000
00000001
00000010
00000011

…

…
11111110
11111111

0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

Why Hex?

00000000
00000001
00000010
00000011

…

…
11111110
11111111

0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Demo

thiebaut — bash — 69×20

bash

- Demo of **hexdump** command on aurora

```
nasm -f elf hello.asm -l hello.lst
ld -melf_i386 hello.o  -o  hello
hexdump -v -C hello
```

- See link to **hexdump** page in CSC231 Schedule page

Mystery Program

```
00000000  7f 45 4c 46 01 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  02 00 03 00 01 00 00 00  80 80 04 08 34 00 00 00  |............4...|
00000020  f0 00 00 00 00 00 00 00  34 00 20 00 02 00 28 00  |........4. ...(.|
00000030  06 00 03 00 01 00 00 00  00 00 00 00 00 80 04 08  |................|
00000040  00 80 04 08 b8 00 00 00  b8 00 00 00 05 00 00 00  |................|
00000050  00 10 00 00 01 00 00 00  b8 00 00 00 b8 90 04 08  |................|
00000060  b8 90 04 08 0e 00 00 00  0e 00 00 00 06 00 00 00  |................|
00000070  00 10 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000080  b8 04 00 00 00 bb 01 00  00 00 b9 b8 90 04 08 ba  |................|
00000090  06 00 00 00 cd 80 b8 04  00 00 00 bb 01 00 00 00  |................|
000000a0  b9 c3 90 04 08 ba 03 00  00 00 cd 80 bb 00 00 00  |................|
000000b0  00 b8 01 00 00 00 cd 80  48 65 6c 6c 6f 20 74 68  |........Hello th|
000000c0  65 72 65 21 0a 0a 00 2e  73 79 6d 74 61 62 00 2e  |ere!....symtab..|
000000d0  73 74 72 74 61 62 00 2e  73 68 73 74 72 74 61 62  |strtab..shstrtab|
000000e0  00 2e 74 65 78 74 00 2e  64 61 74 61 00 00 00 00  |..text..data....|
000000f0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000100  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000110  00 00 00 00 00 00 00 00  1b 00 00 00 01 00 00 00  |................|
00000120  06 00 00 00 80 80 04 08  80 00 00 00 38 00 00 00  |............8...|
00000130  00 00 00 00 00 00 00 00  10 00 00 00 00 00 00 00  |................|
00000140  21 00 00 00 01 00 00 00  03 00 00 00 b8 90 04 08  |!...............|


00000250  1c 00 00 00 80 80 04 08  00 00 00 00 10 00 01 00  |................|
00000260  23 00 00 00 c6 90 04 08  00 00 00 00 10 00 02 00  |#...............|
00000270  2f 00 00 00 c6 90 04 08  00 00 00 00 10 00 02 00  |/...............|
00000280  36 00 00 00 c8 90 04 08  00 00 00 00 10 00 02 00  |6...............|
00000290  00 6d 79 73 74 65 72 79  2e 61 73 6d 00 48 65 6c  |.mystery.asm.Hel|
000002a0  6c 6f 00 48 65 6c 6c 6f  4c 65 6e 00 5f 73 74 61  |lo.HelloLen._sta|
000002b0  72 74 00 5f 5f 62 73 73  5f 73 74 61 72 74 00 5f  |rt.__bss_start._|
000002c0  65 64 61 74 61 00 5f 65  6e 64 00                 |edata._end.|
```