VE280 Recitation Class (1)

Prepared by,

Zhang Yuhang

Yu Jinze

Cheng Songzhe

Data Structure Summer
May 21st, 2012
E-Building, R2-103

Outline

- Intro to Linux
- Linux command
- Process of Compiling
- Coding Style

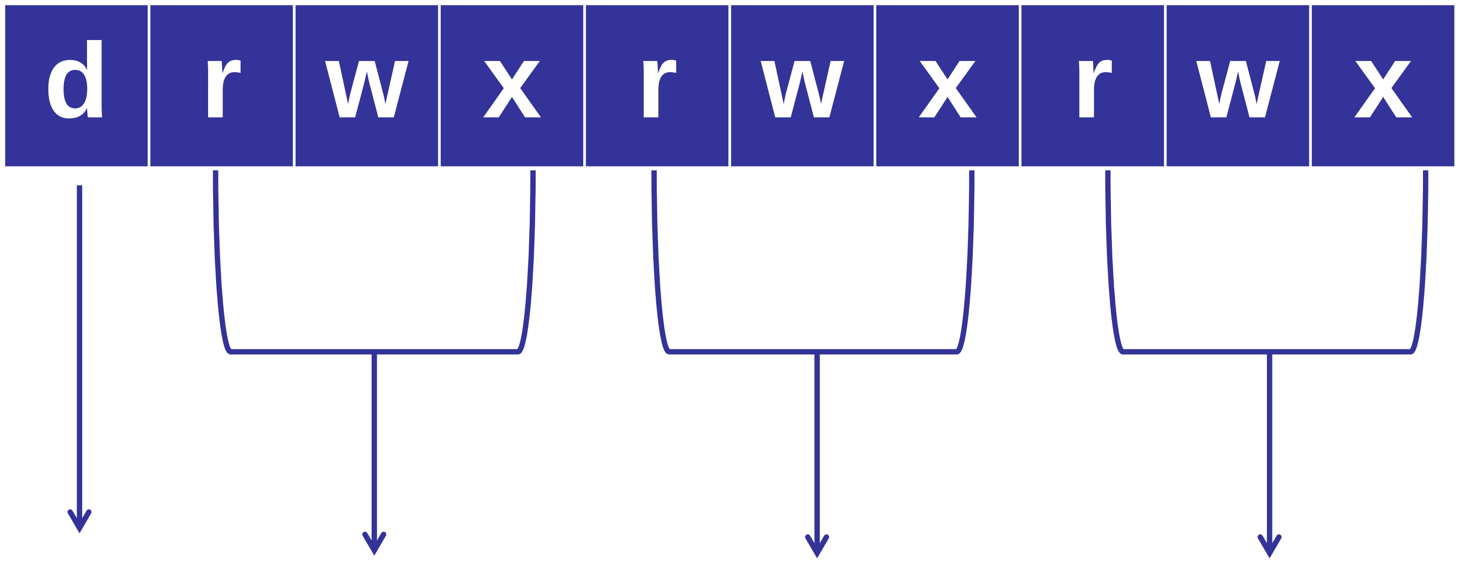
Windows VS Linux

	Windows	Linux
Editor (编辑器)	Visual Studio	vim / emacs / gedit
Compiler (编译器)		gcc
Debugger (调试器)		gdb / sdb

Linux Command

- man
- ls (-l)
- cd
- mv
- cp
- rm
- mkdir
- rmdir
- echo (echo $?)
- cat
- find
- grep
- tar
- chmod
- >
- <
- >>
- |
- *
- ?

Permission



user group other

-	general file
d	directory
l	link file
c	character file
b	block file
p	pipe file

r: permission to read
w: permission to write
x: permission to execute

chmod 777 file

Directory System

/	root directory
.	current directory ./a.out
..	parent directory
~/	home directory

Others:

/usr/include	standard lib (.h)
/usr/lib && /lib	library files (.a && .so)
/usr/bin && /bin	the path of commands
/etc	information files

/var /mnt /media

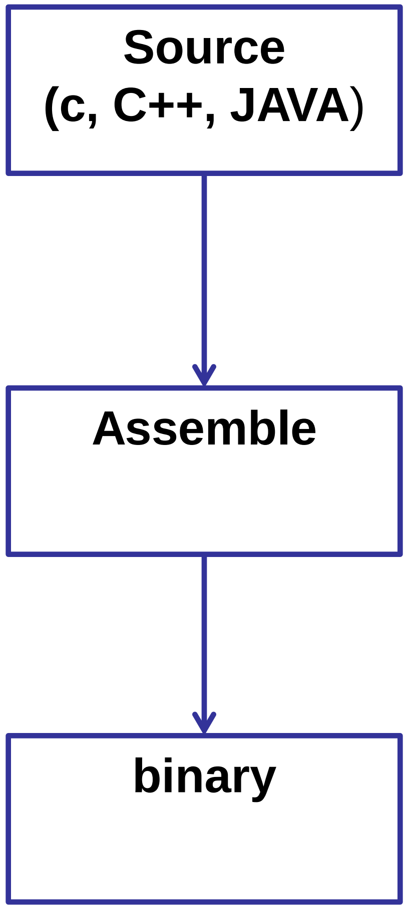
Compiling

- Compile C program
- -> Assembly Language

- Assemble Assembly Language
- -> Machine Language

- Program Run

Compiling



C = A + B

Add $1, $2
Mov $1, $3

```
10000 00001 000010 00000 00000 000000
10001 00001 000011 00000 00000 000001
```

Operation with g++

- g++ -Wall –Werror –O2 –o hello hello.c hello2.c
- g++ hello.c hello2.c
- g++ -c hello.c hello2.c
- g++ -g hello.c hello2.c

Coding Style

- **COMMENTS** (Effect, Modifies, Return Value, any other info for user or client)
- **SHORTER** variable names. (e.g. MatrixIndexNumber -> MtIndNo, ElectricalReadingRoom -> ERdgRm)
- **INDENTATION**(Tab)
- **{}**
- **BLANK** between variables and symbols
- <= **80 Characters** / Line
- **LOOP** in loop blocks: <3. (while, if, for)
- **DEFINE** variables first, then use. (#define pi 3.1415926)
- Numbers appear in code lines: as **FEW** as possible.

```
x = cos(2 * frequency * 3.1415926 + 20);
```

```
#define pi 3.1415926
#define offset 20
x = cos(2 * frequency * pi + offset);
```

Makefile

```
CC = g++
MAINSRCS = multi_source.C
OTHSRCS = say_hello.C
CFLAGS = -g –Wall -Werror
SRCS = $(MAINSRCS) $(OTHSRCS)
OBJS = $(SRCS:.C=.o)
TARGETS = $(MAINSRCS:.C=)

%.o: %.C
		$(CC) $(CFLAGS) -o $@ -c $<

all: $(TARGETS)

$(TARGETS): $(OBJS)
		$(CC) $(CFLAGS) -o $(TARGETS) $(OBJS)

clean:
		rm -f $(OBJS) $(TARGETS)
```

Project Test Cases

- The given “test1” and “test1-result” is way less than enough.
- Create test cases as many as possible to test the correctness of your program.
- Boundary conditions, error conditions, complex conditions.
- May require to submit several test cases later.
- 100 Score Project:
  - Correctness of Program
  - Considerate Testcases
  - Nice Coding Style

Honor Code

- NO coding-detail-related conversations.
- NO copy, even mass modification of code.
- NO unauthorized help.

- Compilation & Decompilation

- Similar or Structurally Equivalent? -> Honor Code.