

LINUX AND C INTRODUCTION

曾歆勋

鸣谢黄博

GETTING STARTED



Pictures from google

HOW TO GET A LINUX OS?

- **Ubuntu Server 18.04 LTS recommend**
 - <https://www.ubuntu.com/download>
 - Burn the iso into your USB. (**ultraiso**)
 - Startup by USB
 - Install

Ubuntu Desktop ›

Download Ubuntu desktop and replace your current operating system whether it's Windows or Mac OS, or, run Ubuntu alongside it.

Do you want to upgrade? Follow our simple guide [↗](#)

Ubuntu Server ›

Whether you want to configure a simple file server or build a fifty thousand-node cloud, you can rely on Ubuntu Server and its five years of guaranteed free upgrades.

Ubuntu Cloud ›

Ubuntu is the reference OS for OpenStack. Try Canonical's OpenStack on a single machine or start building a production cloud on a cluster — just add servers.

Ubuntu flavours ›

Ubuntu flavours offer a unique way to experience Ubuntu with different choices of default applications and settings, backed by the full Ubuntu archive for packages and updates.

Ubuntu for IoT ›

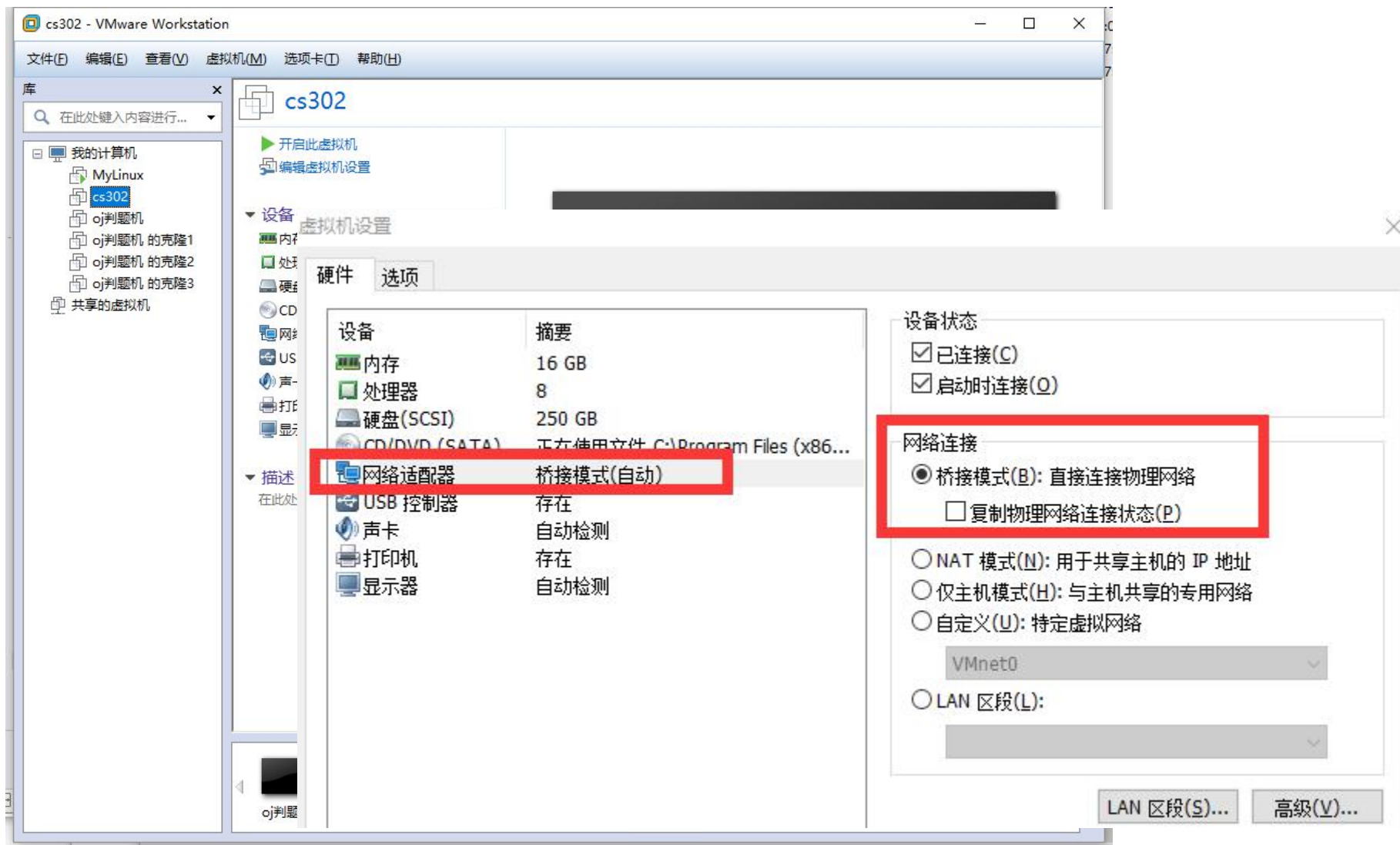
Are you a developer who wants to try snappy Ubuntu Core? The new, transactionally updated Ubuntu for clouds and devices.

HOW TO GET A LINUX OS?

- Using virtual machine
 - Download **Vmware**



VIRTUAL MACHINE



BASIC COMMAND



Pictures from google

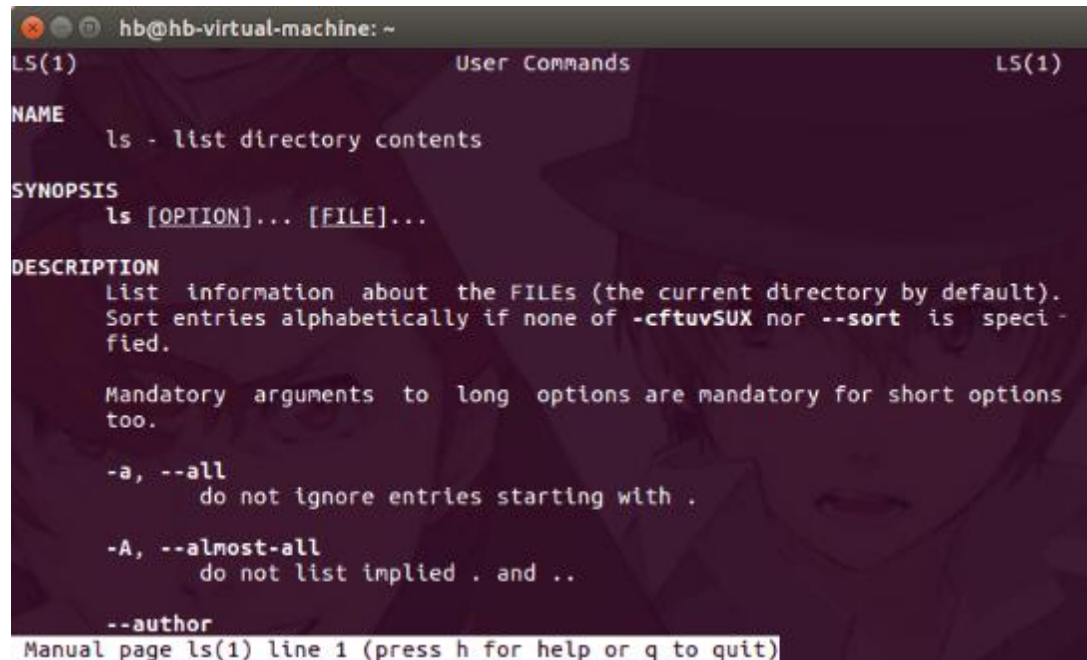
BASIC COMMAND

- **man xxx**
 - show the manual of command xxx
 - you can try “**man man**”

```
hb@hb-virtual-machine: ~  
MAN(1) 手册分页显示工具 MAN(1)  
名称  
man - 在线参考手册的接口  
概述  
man [-C 文件] [-d] [-D] [--warnings[=警告]] [-R 编码] [-L 区域] [-m  
系统[,...]] [-M 路径] [-S 列表] [-e 扩展] [-i|-I] [--regex|--wildcard]  
[--names-only] [-a] [-u] [--no-subpages] [-P 分页程序] [-r 提示] [-7]  
[-E 编码] [--no-hyphenation] [--no-justification] [-p 字符串] [-t]  
[-T[设备]] [-H[浏览器]] [-X[dpi]] [-Z] [[章节] 页 ...] ...  
man -k [apropos 选项] 正则表达式 ...  
man -K [-w|-W] [-S list] [-i|-I] [--regex] [章节] 词语 ...  
man -f [whatis 选项] 页 ...  
man -l [-C 文件] [-d] [-D] [--warnings[=警告]] [-R 编码] [-L 区域] [-P  
分页程序] [-r 提示] [-7] [-E 编码] [-p 字符串] [-t] [-T[设备]]  
[-H[浏览器]] [-X[dpi]] [-Z] 文件 ...  
man -w|-W [-C 文件] [-d] [-D] 页 ...  
man -c [-C 文件] [-d] [-D] 页 ...  
man [-?V]  
描述  
man 是系统的手册分页程序。指定给 man 的 页  
Manual page man(1) line 1 (press h for help or q to quit)
```

BASIC COMMAND

- **ls**
 - list directory contents
 - let's try “**man ls**”



```
hb@hb-virtual-machine: ~  
LS(1) User Commands LS(1)  
NAME  
ls - list directory contents  
SYNOPSIS  
ls [OPTION]... [FILE]...  
DESCRIPTION  
List information about the FILES (the current directory by default).  
Sort entries alphabetically if none of -cftuvSUX nor --sort is speci-  
fied.  
  
Mandatory arguments to long options are mandatory for short options  
too.  
  
-a, --all  
do not ignore entries starting with .  
  
-A, --almost-all  
do not list implied . and ..  
  
--author  
Manual page ls(1) line 1 (press h for help or q to quit)
```


BASIC COMMAND

- **mkdir**
 - make a new directory
 - try **man mkdir** by yourself
 - try to make a directory named OS in HOME(~)



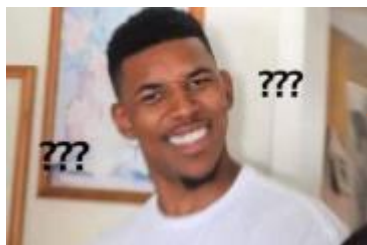
```
hb@hb-virtual-machine: ~  
hb@hb-virtual-machine:~$ mkdir OS  
hb@hb-virtual-machine:~$ ls  
examples.desktop  OS  
Firefox_wallpaper.png  VMwareTools-10.2.0-7259539.tar.gz  
hello              vmware-tools-distrib  
hello.c            公共的  
hb@hb-virtual-machine:~$
```

The screenshot shows a terminal window with a dark background and a faint anime-style illustration. The terminal output shows the successful execution of the 'mkdir OS' command and the subsequent 'ls' command, which lists the newly created 'OS' directory among other files. On the right side of the terminal window, there are vertical Chinese text links: '模板' (Templates), '下载' (Download), '视频' (Videos), '图片' (Images), '桌面' (Desktop), '音乐' (Music), and '文档' (Documents).

BASIC COMMAND

- ls

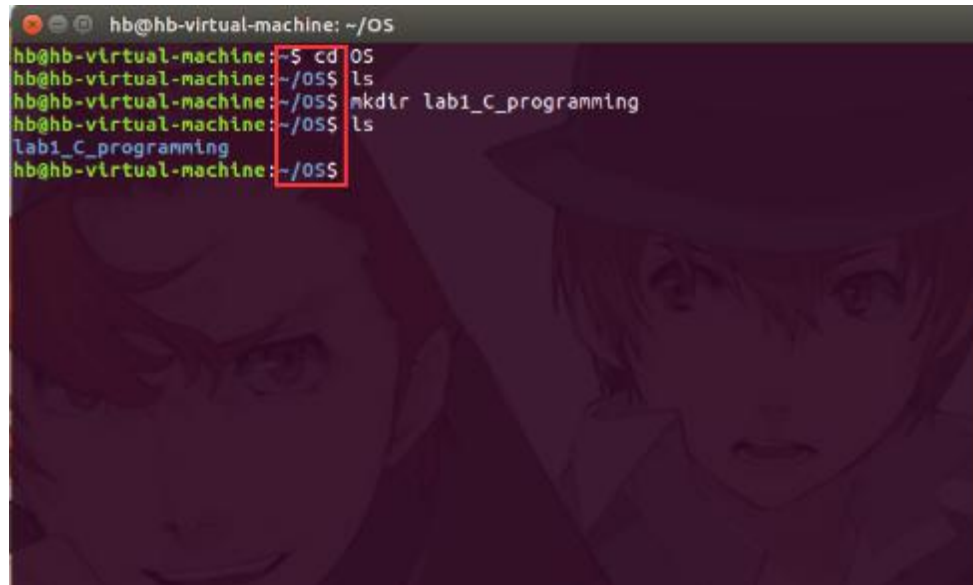
```
hb@hb-virtual-machine: ~  
hb@hb-virtual-machine:~$ ls  
examples.desktop      hello.c  
Firefox_wallpaper.png VMwareTools-10.2.0-7259539.tar.gz  
hello                 vmware-tools-distrib  
hb@hb-virtual-machine:~$
```



- How to go into OS?

BASIC COMMAND

- **cd**
 - change directory
 - let's go to OS directory
 - Try cd /home
 - Try cd ~, cd ..

A terminal window titled 'hb@hb-virtual-machine: ~/OS' with a dark background and a faint anime-style illustration. The terminal shows a sequence of commands: 'cd OS', 'ls', 'mkdir lab1_C_programming', 'ls', and 'cd lab1_C_programming'. The prompt changes from '~/' to '~/OS' after the first 'cd' command. The 'cd lab1_C_programming' command is highlighted with a red box.

```
hb@hb-virtual-machine: ~/OS
hb@hb-virtual-machine:~/OS$ cd OS
hb@hb-virtual-machine:~/OS$ ls
hb@hb-virtual-machine:~/OS$ mkdir lab1_C_programming
hb@hb-virtual-machine:~/OS$ ls
lab1_C_programming
hb@hb-virtual-machine:~/OS$
```

BASIC COMMAND

- **apt-get install vim**
 - This command need root authority. Use sudo to swtich get root authority for a while.
 - apt-get handling packages
 - install means we want to install this package
 - you can man apt-get to learn details

```
hb@hb-virtual-machine:~/OS$ sudo apt-get install vim
[sudo] hb 的密码:
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
vim 已经是最新版 (2:7.4.1689-3ubuntu1.2)。
升级了 0 个软件包，新安装了 0 个软件包，要卸载 0 个软件包，有 366 个软件包未被升级。
hb@hb-virtual-machine:~/OS$
```

EDITOR



Pictures from google

WHY WE NEED EDITOR

- **Server**
 - When you connect to a linux server, sometimes it doesn't have GUI.

VIM

- **vim**
 - A powerful editor.
 - You can use vim/vi in terminal to edit files.
 - In order to get full functions about vim, we can install some packages first.

CONFIGURE VIM

- This is not necessary, just let you be more comfortable when using vim.
- Go to HOME
- Using vim command to edit file **.vimrc**

A terminal window with a dark background and light green text. The title bar at the top reads 'hb@hb-virtual-machine: ~'. The command prompt shows 'hb@hb-virtual-machine:~\$ vim .vimrc' with a white cursor at the end of the line.

```
hb@hb-virtual-machine: ~  
hb@hb-virtual-machine:~$ vim .vimrc
```

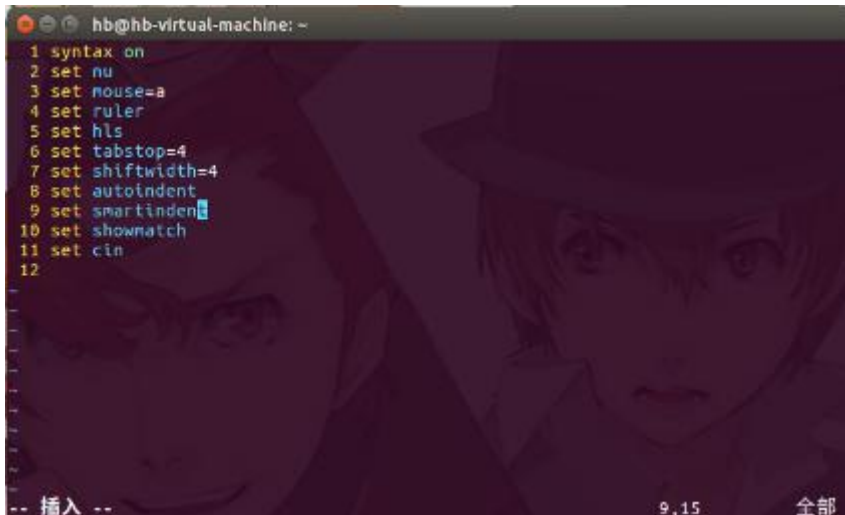

VIM

- **Vim has three modes, they are:**
 - Command mode: you can not input text, everything you input will be command.
 - Insert mode: you can input text. Press **Esc** to return command mode.
 - Last line mode: you can input special command. Such as exit and find string.

VIM

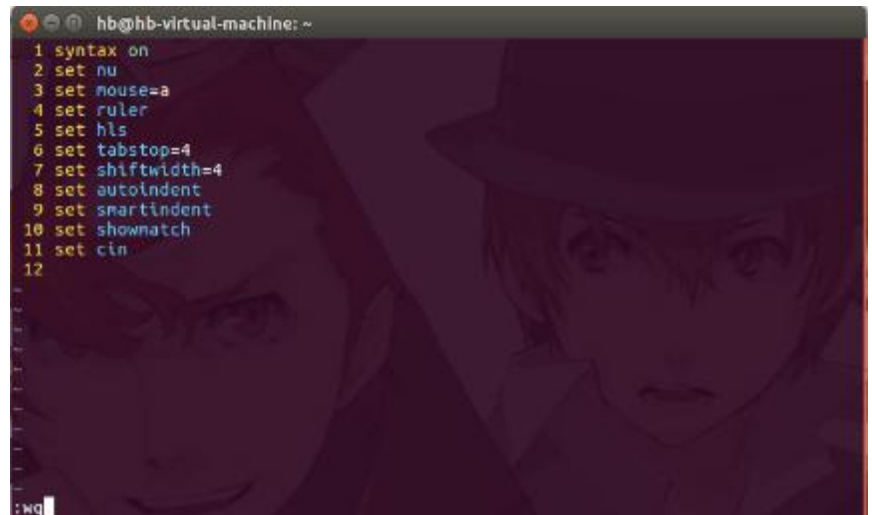
- **Configure**

- Press **i** to go to insert mode
- Input text
- Press **Esc** go back to command mode
- Press **shift + ;** to go to last line mode
- Press **wq** to write and quit



A terminal window titled 'hb@hb-virtual-machine: ~' showing a Vim configuration file. The file contains 12 lines of settings. The cursor is at line 12, column 15, in insert mode. The status bar at the bottom shows '-- 插入 --' (Insert mode), '9,15' (line 9, column 15), and '全部' (All).

```
1 syntax on
2 set nu
3 set mouse=a
4 set ruler
5 set hls
6 set tabstop=4
7 set shiftwidth=4
8 set autoindent
9 set smartindent
10 set showmatch
11 set cin
12
```



A terminal window titled 'hb@hb-virtual-machine: ~' showing the same Vim configuration file. The cursor is at line 12, column 15, in command mode. The status bar at the bottom shows ':wq' (write and quit).

```
1 syntax on
2 set nu
3 set mouse=a
4 set ruler
5 set hls
6 set tabstop=4
7 set shiftwidth=4
8 set autoindent
9 set smartindent
10 set showmatch
11 set cin
12
```

YOU MAY NEED THIS

version 1.1
April 1st, 06
翻译: 2006-5-21

vi / vim 键盘图

Esc 命令模式																	
~ 转换大小写	! 外部过滤器	@ 运行宏	# prev ident	\$ 行尾	% 括号匹配	^ "软" 行首	& 重复:s	* next ident	(句首) 下一句首	"soft" bol down	+ 后一行行首					
· 跳转到标注	1	2	3	4	5	6	7	8	9	0 "硬" 行首	- 前一行行首	= 自动格式化					
Q 切换到 ex 模式	W 下一单词	E 词尾	R 替换模式	T back 'till	Y 拷贝行	U 撤消行内命令	I 到行首插入	O 分段(前)	P 粘贴(前)	{ 段首	}	段尾					
q 录制宏	w 下一单词	e 词尾	r 替换字符	t 'till	y 拷贝 1,3	u 撤消命令	i 插入模式	o 分段(后)	p 粘贴(后)	[杂项]	杂项					
A 在行尾附加	S 删除行并插入	D 删除至行尾	F 行内字符反向查找	G 文尾/行号	H 屏幕顶行	J 合并两行	K 帮助	L 屏幕底行	: ex 命令	" 寄存器标识	行首/列						
a 附加	s 删除字符并插入	d 删除 1,3	f 行内字符查找	g 附加命令 6	h ←	j ↓	k ↑	l →	; 重复 t/T/f/F	' 跳转到标注的行首	\ 未用!						
Z 退出 4	X 退格	C 修改至行末	V 可视行模式	B 前一单词	N 查找上一处	M 屏幕中间行	< 反缩进 3	> 缩进 3	? 向前搜索								
Z 附加命令 5	x 删除(字符)	c 修改 1,3	v 可视模式	b 前一单词	n 查找下一处	m 设置标注	, t/T/f/F 反向	. 重复命令	/ 向后搜索								

动作

移动光标, 或者定义操作的范围

命令

直接执行的命令, 红色命令进入编辑模式

操作

后面跟随表示操作范围的指令

extra

特殊功能, 需要额外的输入

q 后跟字符参数

w,e,b 命令
小写(b): quux(foo, bar, baz);
大写(B): quux(Foo, Bar, Baz);

主要 ex 命令:
:w (保存), :q (退出), :q! (不保存退出)
:e f (打开文件 f),
:%s/x/y/g ('y' 全局替换 'x'),
:h (帮助 in vim), :new (新建文件 in vim),

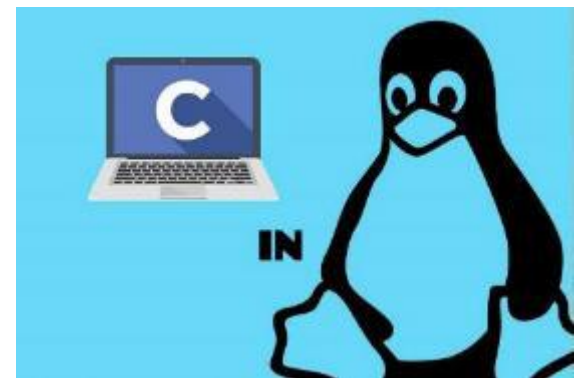
其它重要命令:
CTRL-R: 重复 (vim),
CTRL-F/-B: 上翻/下翻,
CTRL-E/-Y: 上滚/下滚,
CTRL-V: 块可视模式 (vim only)

可视模式:
漫游后对选中的区域执行操作 (vim only)

备注:
(1) 在 拷贝/粘贴/删除 命令前使用 "x (x=a..z,*) 使用命令的寄存器('剪贴板') (如: "ay\$ 拷贝剩余的行内容至寄存器 'a')
(2) 命令前添加数字 多遍重复操作 (e.g.: 2p, d2w, 5l, d4j)
(3) 重复本字符在光标所在行执行操作 (dd = 删除本行, >> = 行首缩进)
(4) ZZ 保存退出, ZQ 不保存退出
(5) zt: 移动光标所在行至屏幕顶端, zb: 底端, zz: 中间
(6) gg: 文首 (vim only), gf: 打开光标处的文件名 (vim only)

原图: www.viemu.com 翻译: fdl (linuxsir)

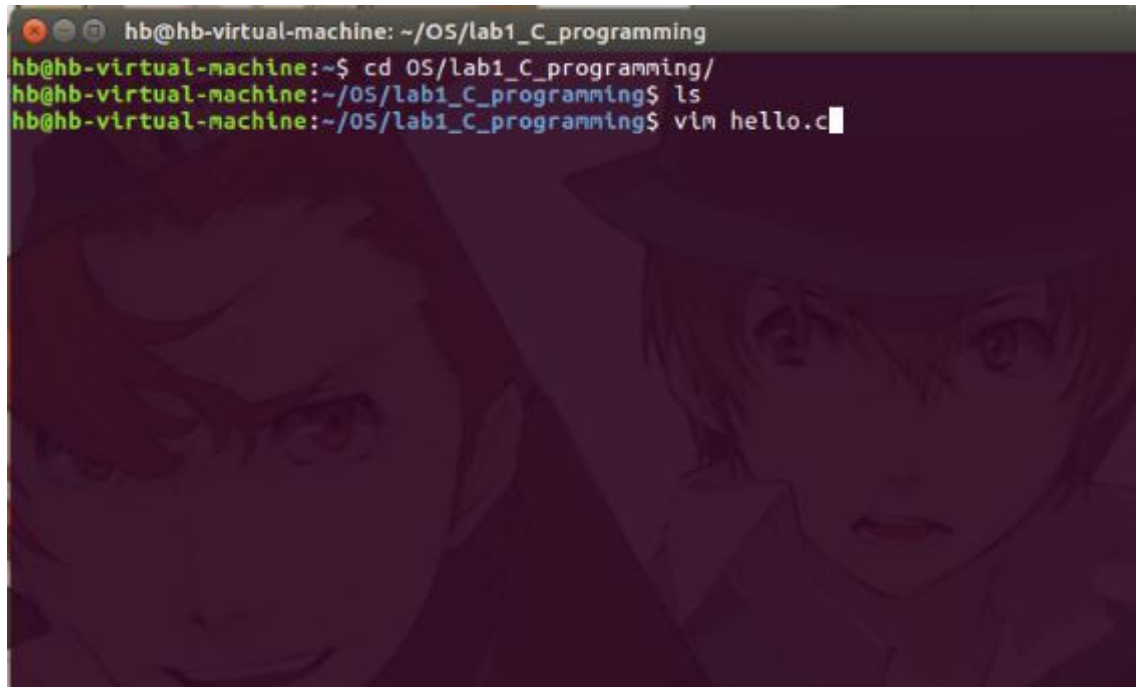
PROGRAMMING



Pictures from google

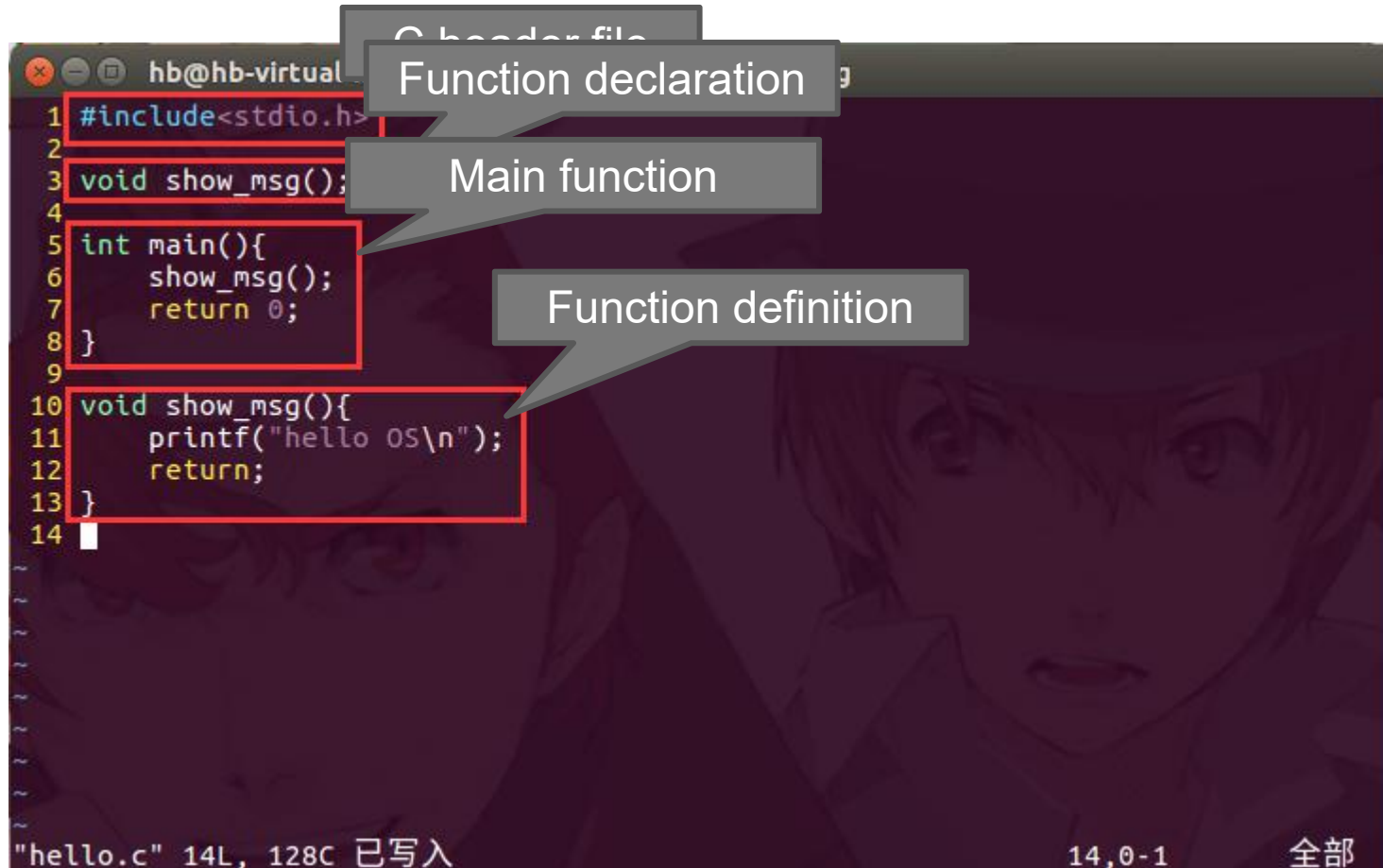
FIRST C PROGRAM

- Now we are ready for our first C program.
 - Go to lab1_C_programming directory
 - And edit file: hello.c

A terminal window titled 'hb@hb-virtual-machine: ~/OS/lab1_C_programming' is shown. The background of the terminal has a dark purple overlay with a faint anime-style illustration of two characters. The terminal text shows the user navigating to the 'OS/lab1_C_programming' directory and then using 'vim' to edit 'hello.c'.

```
hb@hb-virtual-machine: ~/OS/lab1_C_programming
hb@hb-virtual-machine:~$ cd OS/lab1_C_programming/
hb@hb-virtual-machine:~/OS/lab1_C_programming$ ls
hb@hb-virtual-machine:~/OS/lab1_C_programming$ vim hello.c
```

FIRST C PROGRAM



A JAVA PROGRAM

```
hb@hb-virtual-machine: ~/OS/lab1_C_programming
1 import java.io.*;
2 import java.util.*;
3
4 public class HelloOS{
5
6     public static void print_msg(){
7         System.out.println("hello OS");
8     }
9
10    public static void main(string[] args){
11        print_msg();
12    }
13 }
14
```

"hello.java" 14L, 195C 已写入

```
hb@hb-virtual-machine: ~/OS/lab1_C_programming
1 #include<stdio.h>
2
3 void show_msg();
4
5 int main(){
6     show_msg();
7     return 0;
8 }
9
10 void show_msg(){
11     printf("hello OS\n");
12     return;
13 }
14
```

"hello.c" 14L, 128C

14,0-1 全部

FIRST C PROGRAM

- How to run our program?
 - We need compile it!
 - Ubuntu has GCC (GNU Compiler Collection)
 - Let's compile our first c program

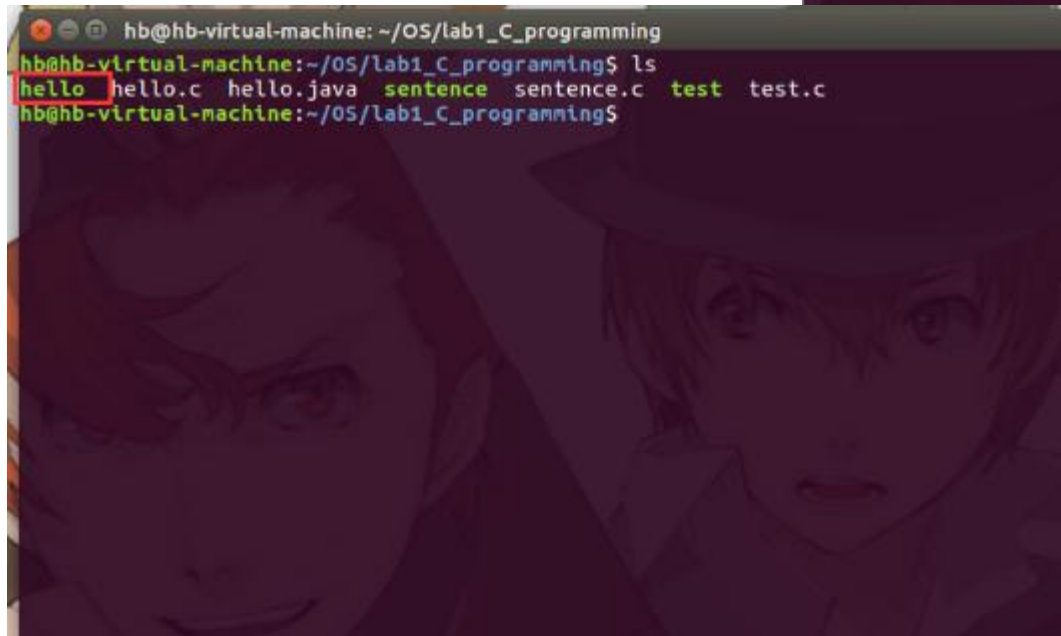
```
root@hb-virtual-machine: ~  
GCC(1)  
NAME  
gcc - GNU project C and C++ compiler  
SYNOPSIS  
gcc [-c|-S|-E] [-std=standard]  
    [-g] [-pg] [-Olevel]  
    [-Wwarn...] [-Wpedantic]  
    [-Idir...] [-Ldir...]  
    [-Dmacro[=defn]...] [-Umacro]  
    [-foption...] [-mmachine-option...]  
    [-o outfile] [@file] infile...  
  
Only the most useful options are listed here; see below for the  
remainder. g++ accepts mostly the same options as gcc.  
DESCRIPTION  
When you invoke GCC, it normally does preprocessing, compilation,  
assembly and linking. The "overall options" allow you to stop this  
process at an intermediate stage. For example, the -c option says not  
to run the linker. Then the output consists of object files output by  
the assembler.  
Manual page gcc(1) line 1 (press h for help or q to quit)
```


ABOUT GCC

- **gcc**
 - -c Compile or assemble the source files, but do not link.
 - -S Stop after the stage of compilation proper; do not assemble.
 - -E Stop after the preprocessing stage; do not run the compiler proper.
 - -o **filename** Place output in file file.
 - If no parameters, gcc will do all things and output an execute file a.out

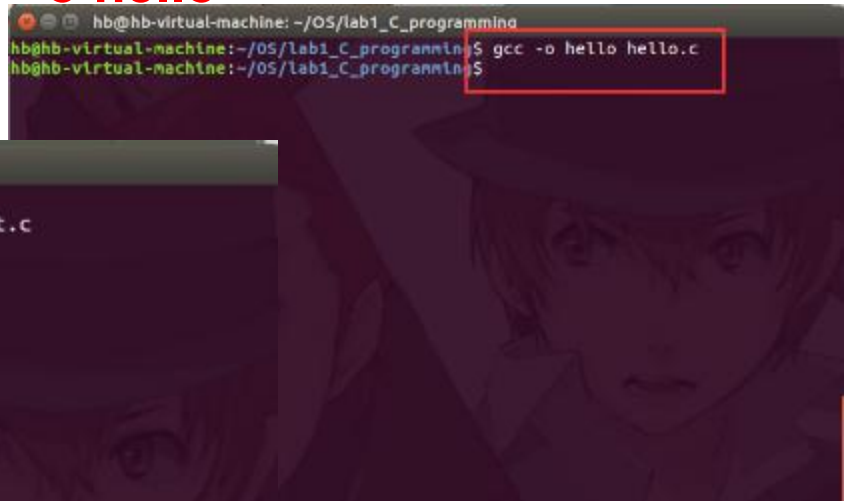
FIRST C PROGRAM

- Input **gcc -o hello hello.c** on terminal
- You can also input **gcc hello.c -o hello**



```
hb@hb-virtual-machine: ~/OS/lab1_C_programming
hb@hb-virtual-machine:~/OS/lab1_C_programming$ ls
hello hello.c hello.java sentence sentence.c test test.c
hb@hb-virtual-machine:~/OS/lab1_C_programming$
```

A terminal window screenshot showing the command 'ls' being executed. The output lists files: 'hello', 'hello.c', 'hello.java', 'sentence', 'sentence.c', 'test', and 'test.c'. The word 'hello' in the output is highlighted with a red box. The background of the terminal window features a faint anime-style illustration of a character.

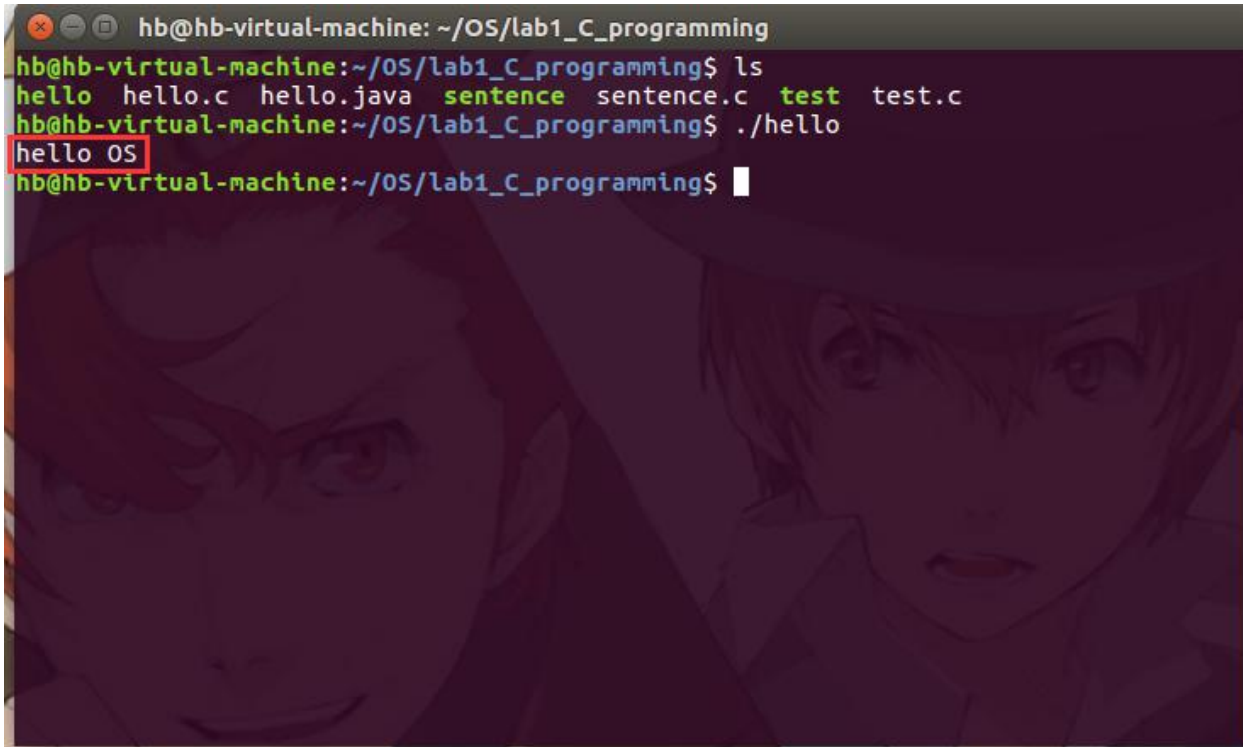


```
hb@hb-virtual-machine: ~/OS/lab1_C_programming
hb@hb-virtual-machine:~/OS/lab1_C_programming$ gcc -o hello hello.c
hb@hb-virtual-machine:~/OS/lab1_C_programming$
```

A terminal window screenshot showing the command 'gcc -o hello hello.c' being executed. The command and its successful execution are highlighted with a red box. The background of the terminal window features a faint anime-style illustration of a character.

FIRST C PROGRAM

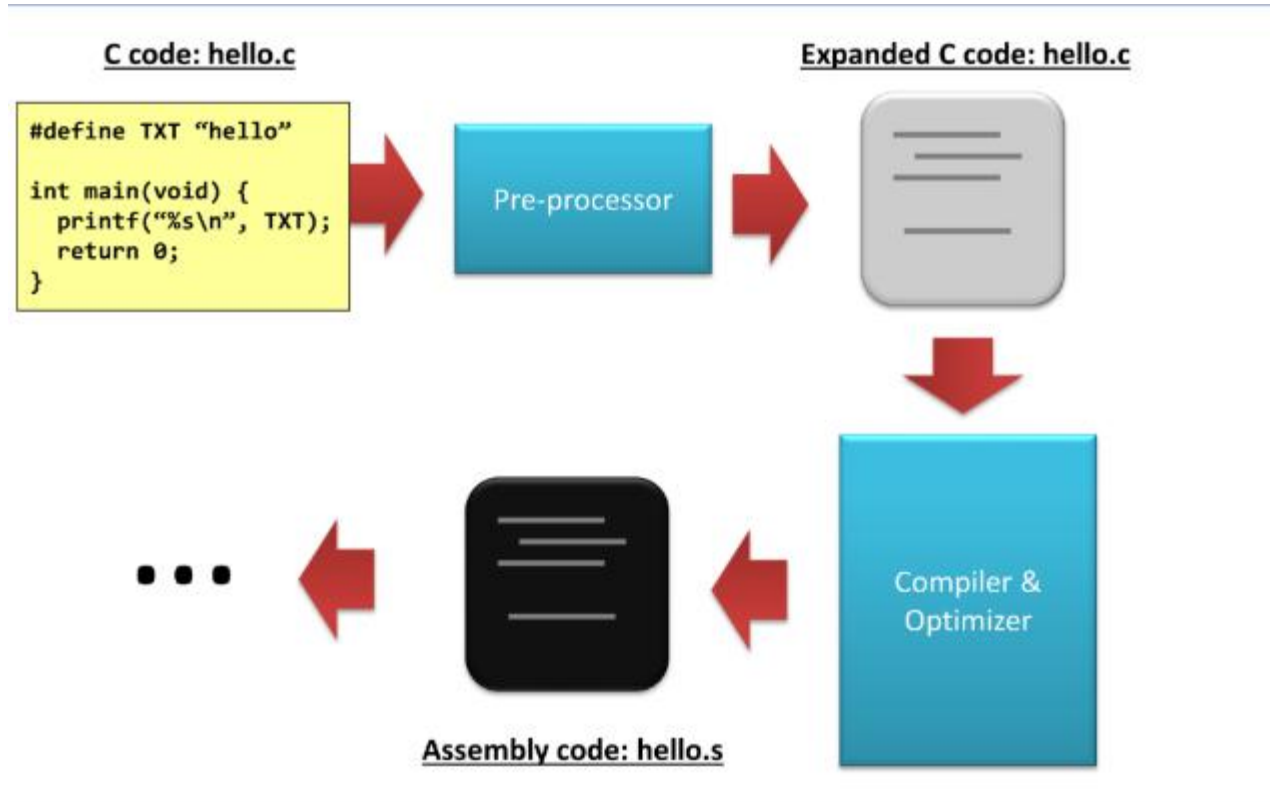
- Let's run it!
 - `./hello`

A terminal window titled 'hb@hb-virtual-machine: ~/OS/lab1_C_programming' displays the following commands and output:

```
hb@hb-virtual-machine:~/OS/lab1_C_programming$ ls
hello hello.c hello.java sentence sentence.c test test.c
hb@hb-virtual-machine:~/OS/lab1_C_programming$ ./hello
hello OS
hb@hb-virtual-machine:~/OS/lab1_C_programming$
```

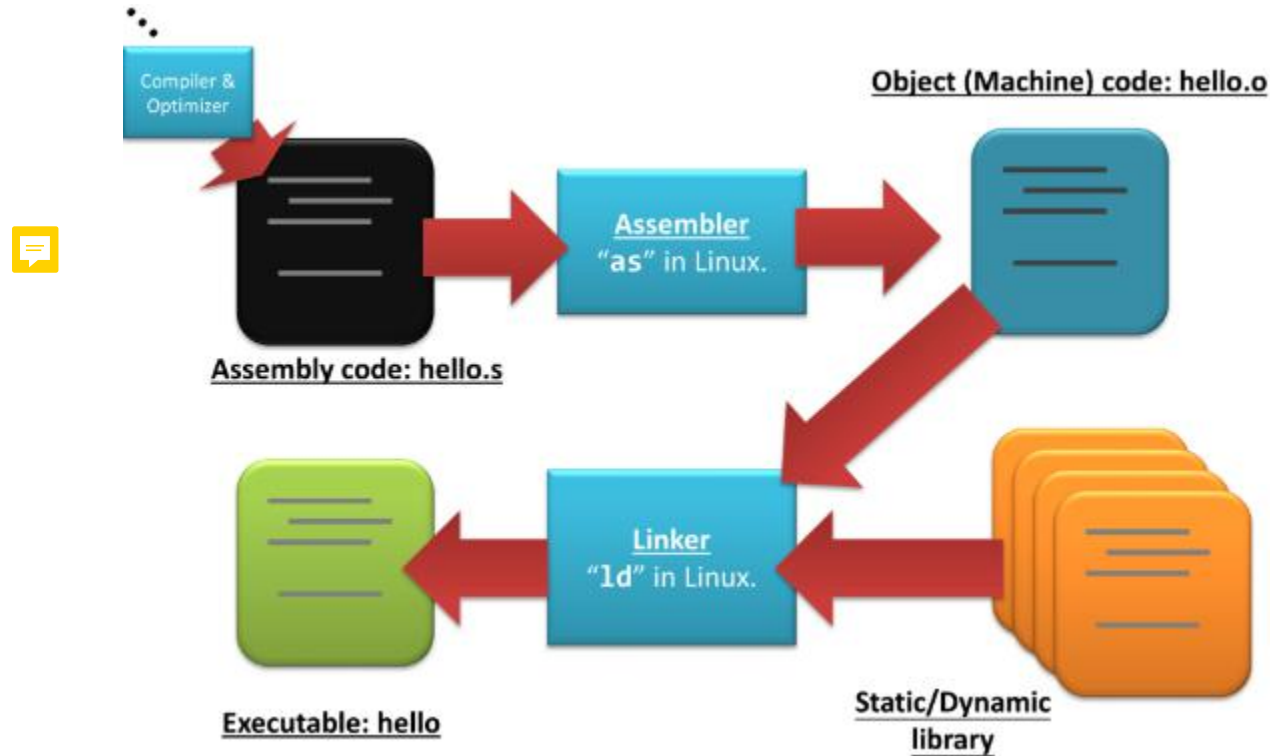
The output 'hello OS' is highlighted with a red box. The terminal window has a dark background with a faint anime-style illustration of two characters.

WHAT HAPPENS?



Pictures from: <https://calvinkam.github.io/csci3150-Fall17-lab3/building-a-program.html>

WHAT HAPPENS?



Pictures from: <https://calvinkam.github.io/csci3150-Fall17-lab3/building-a-program.html>

WHAT HAPPENS?

- Pre-processor
- Input **gcc -E hello.c**
 - Replace #include

```
extern int ftrylockfile (FILE *__stream) __attribute__ ((__nothrow__ , __leaf__))  
) ;  
  
extern void funlockfile (FILE *__stream) __attribute__ ((__nothrow__ , __leaf__))  
);  
# 942 "/usr/include/stdio.h" 3 4  
  
# 2 "hello.c" 2  
  
# 3 "hello.c"  
void show_msg();  
  
int main(){  
    show_msg();  
    return 0;  
}  
  
void show_msg(){  
    printf("hello OS\n");  
    return;  
}  
hb@hb-virtual-machine:~/OS/lab1_C_programming$
```

WHAT HAPPENS?

- **compiler and optimizer**
 - First check syntax and analyze it.
 - Then produce assembly code.
 - Optimizer will improve the code quality.

MORE ABOUT OPTIMIZER

- Consider this example opt.c

```
hb@hb-virtual-machine:~/OS/lab1_C_programming$ cat opt.c
#include<stdio.h>

int main(){
    int x = 0;
    x += 1;
    x += 1;
    x += 1;
    printf("%d\n", x);
    return 0;
}
```

- We open the optimizer to get assembly code

```
hb@hb-virtual-machine:~/OS/lab1_C_programming$ gcc -S opt.c -O0 -o opt0.s
hb@hb-virtual-machine:~/OS/lab1_C_programming$ gcc -S opt.c -O1 -o opt1.s
hb@hb-virtual-machine:~/OS/lab1_C_programming$
```


MORE ABOUT OPTIMIZER

opt0.s

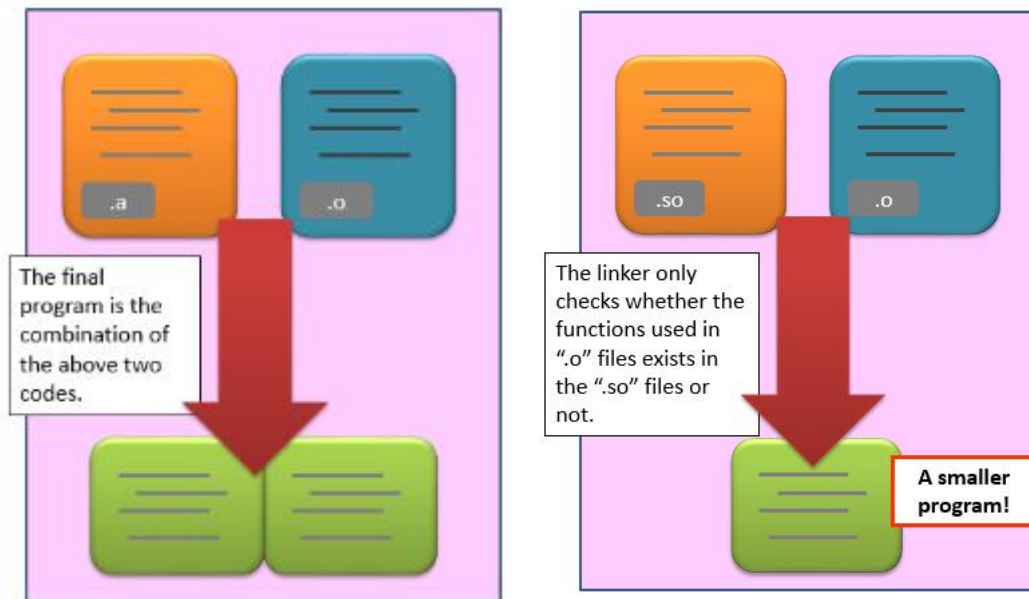
```
movq    %rsp, %rbp
.cfi_def_cfa_register 6
subq    $16, %rsp
movl    $0, -4(%rbp)
addl    $1, -4(%rbp)
addl    $1, -4(%rbp)
addl    $1, -4(%rbp)
movl    -4(%rbp), %eax
movl    %eax, %esi
movl    $.LC0, %edi
movl    $0, %eax
call    printf
movl    $0, %eax
leave
.cfi_def_cfa 7, 8
ret
```

opt1.s

```
main:
.LFB23:
.cfi_startproc
subq    $8, %rsp
.cfi_def_cfa_offset 16
movl    $3, %edx
movl    $.LC0, %esi
movl    $1, %edi
movl    $0, %eax
call    __printf_chk
movl    $0, %eax
addq    $8, %rsp
.cfi_def_cfa_offset 8
ret
.cfi_endproc
.LFE23:
.size   main, .-main
.ident  "GCC: (Ubuntu 5.4.0-6ubuntu1~16.04.4) 5.4.0"
.section .note.GNU-stack,"",@progbits
```

WHAT HAPPENS?

- Finally, Linker will link share library or static library with your code. And form executable file.



- Pictures from: <https://calvinkam.github.io/csci3150-Fall17-lab3/assembler-and-linker.html>

C LANGUAGE



Pictures from google

WHY WE NEED C

编程语言排行榜 TOP 50 榜单

排名	编程语言	流行度	对比上月	年度明星语言
1	Java	15.876%	▼ 1.028%	2015, 2005
2	C	12.424%	▼ 0.913%	2017, 2008
3	Python	7.574%	▼ 0.72%	2010, 2007, 2018
4	C++	7.444%	▼ 0.714%	2003
5	Visual Basic .NET	7.095%	▲ 0.636%	
6	JavaScript	2.848%	▼ 0.454%	2014
7	C#	2.846%	▼ 0.438%	
8	PHP	2.271%	▼ 0.409%	2004
9	SQL	1.900%	▼ 0.377%	
10	Objective-C	1.447%	▼ 0.334%	2012, 2011

EXAMPLE

Please write a simple program.

Get 2 integer from user input

Print the square root of their square sum.

EXAMPLE

- We write the code and save it. Do you remember how to use vim?

```
1 #include<stdio.h>
2 #include<math.h>
3
4 int main(){
5     int x, y;
6     double res;
7     scanf("%d%d", &x, &y);
8     res = sqrt(pow(x, 2) + pow(y, 2));
9     printf("%f\n", res);
10 }
```

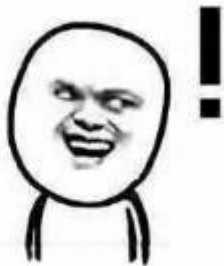
- Then compile it.

```
mark@ubuntu:~/os_teaching$ gcc sqrt.c
/tmp/ccq9z1iW.o: In function `main':
sqrt.c:(.text+0x47): undefined reference to `pow'
sqrt.c:(.text+0x64): undefined reference to `pow'
sqrt.c:(.text+0x6e): undefined reference to `sqrt'
collect2: error: ld returned 1 exit status
```



EXAMPLE

- Do not be afraid of meeting problems, we have many ways to solve it.
- Let's search it on the internet.
- <https://stackoverflow.com/questions/13228111/c-undefined-reference-to-sqrt>
- Here is the solution!



我想到了！

you should link the math library when compiling

`-lm`

EXAMPLE

OOPs, it works.
Thanks to
stackoverflow!

- Let's try again.

```
mark@ubuntu:~/os_teaching$ gcc sqrt.c -lm
mark@ubuntu:~/os_teaching$
```

- And it works!

```
mark@ubuntu:~/os_teaching$ ./a.out
4
5
6.403124
mark@ubuntu:~/os_teaching$
```


STRANGE CODE

```
1 #include<stdio.h>
2 int main(){
3     int a[10];
4     for(int i=0;i<=10;i++)
5     {
6         a[i] = 0;
7         // printf("%p %p\n", &a[i], &i);
8     }
9 }
10
```

Sometimes these code will cause infinite loop, sometimes will not.
Why?

Try yourself

POINTER

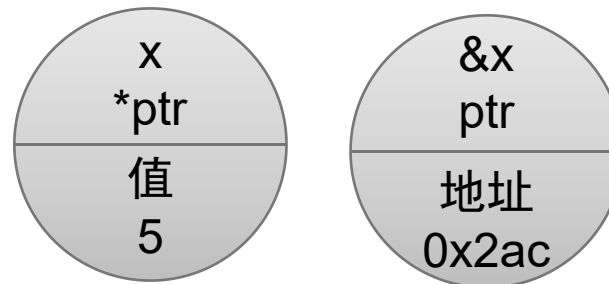
- 指针——C语言的精髓

```
1 #include<stdio.h>
2
3 void add(int* x){
4     (*x)++;
5 }
6
7 int main(){
8     int x=0;
9     add(&x);
10    printf("%d\n",x);
11 }
12
```

& 取址操作符

* 取值操作符

```
int x = 5;
int *ptr = &x;
```



You will meet pointer many times in this course.

EXERCISE



Pictures from google

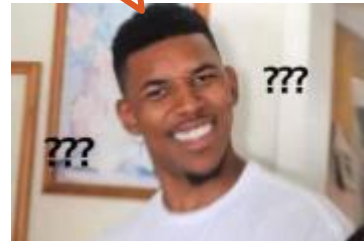
EXERCISE

- **gcc exercise**
 - Try `gcc hello.c` what do you find?
 - Try `gcc -c hello.c` what do you find?
 - Try `gcc -E hello.c` what do you find?
 - Try `gcc -S hello.c` what do you find?
 - How about add `-o` output to these commands?

EXERCISE

- **Be aware**
 - How to use array
 - The average number

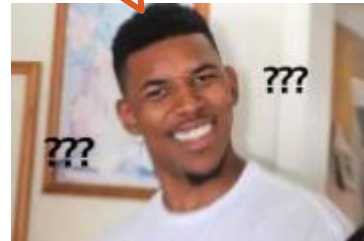
I will input 20 integers, please calculate the maximum, minimum and the average number of these 20 integers.



EXERCISE

- **Be aware**
 - How to sort them?
 - Can you use library?

I will input n integers, ($1 \leq n \leq 100$),
please sort them by ascending order.

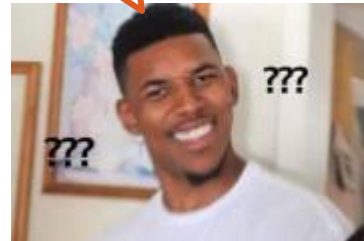


EXERCISE

- **Be aware**
 - spaces

Give you an integer n . Please print
the following picture. (ex. $n = 7$)
 $n \leq 11$, n is odd.

```
  *
 ***
*****
*****
*****
 ***
  *
```



EXERCISE

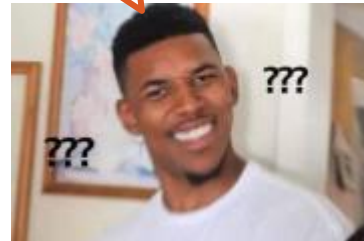
- **Be aware**
 - directions

Give you an integer n . Please print
the following picture. (ex. $n = 9$)
 $n \leq 100$, n is a square number

123

894

765



THANK YOU