C PROGRAMMING

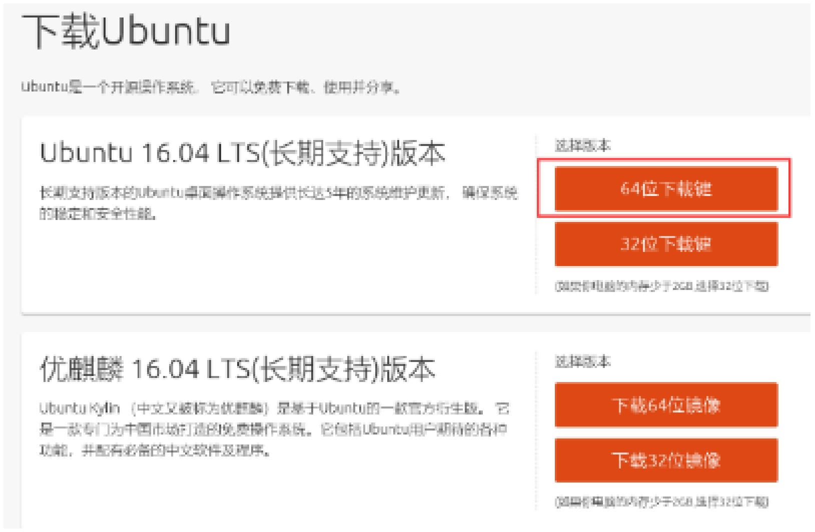
11510899 HUANG BO

GETTING STARTED

Pictures from google

HOW TO GET A LINUX OS?

- **Ubuntu 16.04 LTS recommend**
  - http://cn.ubuntu.com/download/
  - Burn the iso into your USB. (ultraiso)
  - Startup by USB
  - Install



下载Ubuntu

Ubuntu是一个开源操作系统，它可以免费下载、使用并分享。

Ubuntu 16.04 LTS(长期支持)版本

长期支持版本的Ubuntu桌面操作系统提供长达5年的系统维护更新，确保系统的稳定和安全性能。

选择版本

(如果你电脑的内存少于2GB 选择32位下载)

优麒麟 16.04 LTS(长期支持)版本

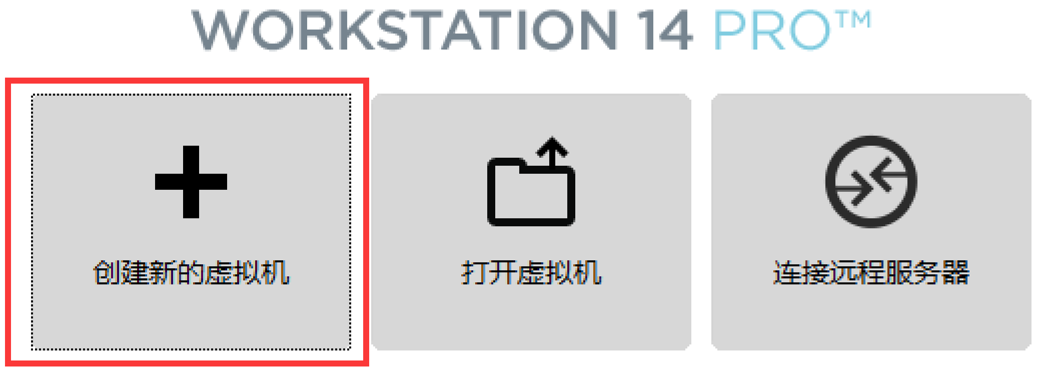
Ubuntu Kylin （中文又被称为优麒麟）是基于Ubuntu的一款官方衍生版。它是一款专门为中国市场打造的免费操作系统。它包括Ubuntu用户期待的各种功能，并配有必备的中文软件及程序。

选择版本

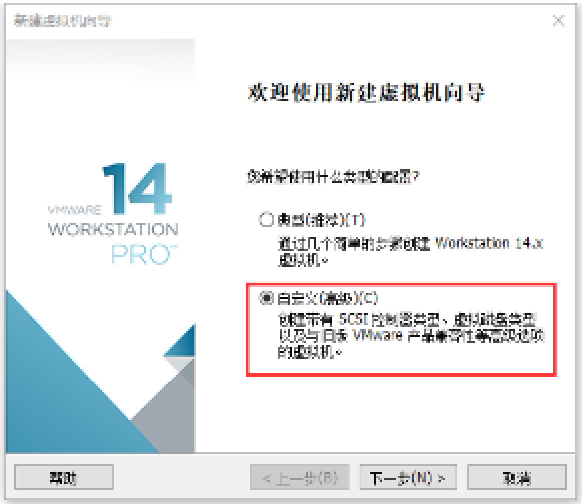
(如果你电脑的内存少于2GB 选择32位下载)

HOW TO GET A LINUX OS?

- **Using virtual machine**
  - http://cn.ubuntu.com/download/
  - Download Vmware



VIRTUAL MACHINE

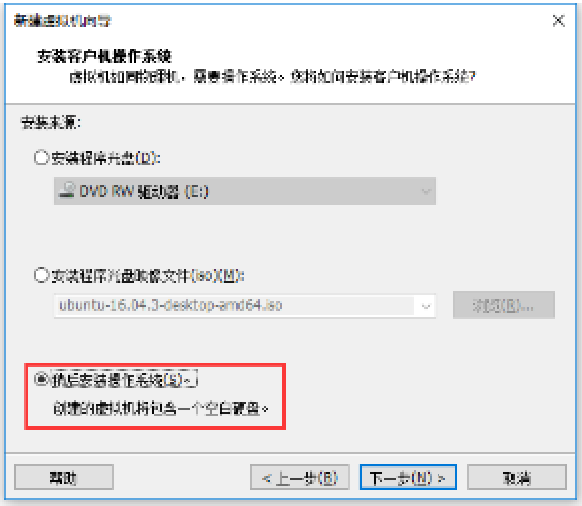

新建虚拟机向导

欢迎使用新建虚拟机向导

您希望使用什么类型的配置?

○ 典型(推荐)(T)
通过几个简单的步骤创建 Workstation 14.x 虚拟机。

[✓ 自定义(高级)(C)]
创建带有 SCSI 控制器类型、虚拟磁盘类型以及与旧版 VMware 产品兼容性等高级选项的虚拟机。


新建虚拟机向导

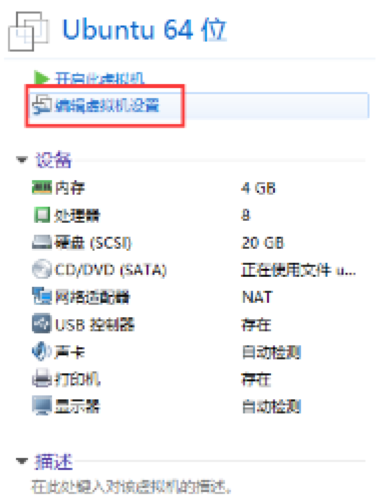
安装客户机操作系统
虚拟机如同物理机，需要操作系统。您将如何安装客户机操作系统?

安装来源:

○ 安装程序光盘(D):
DVD RW 驱动器 (E:)

○ 安装程序光盘映像文件(iso)(M):
ubuntu-16.04.3-desktop-amd64.iso

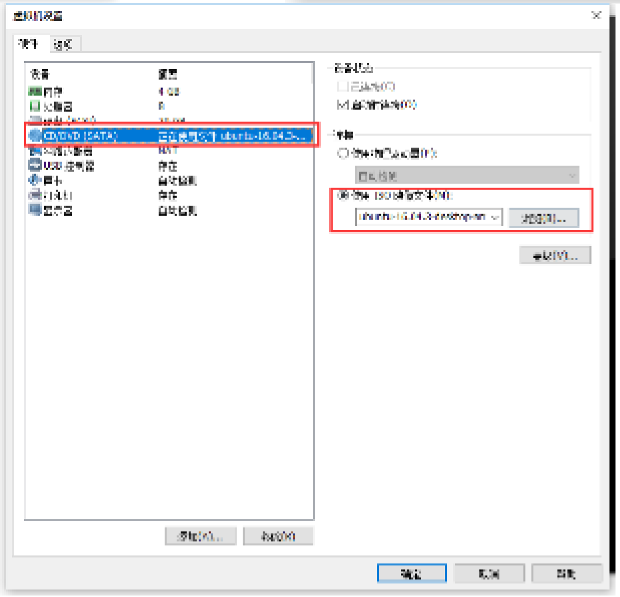
[✓ 稍后安装操作系统(S)。]
创建的虚拟机将包含一个空白硬盘。


Ubuntu 64 位

设备

内存	4 GB
处理器	8
硬盘 (SCSI)	20 GB
CD/DVD (SATA)	正在使用文件 u...
网络适配器	NAT
USB 控制器	存在
声卡	自动检测
打印机	存在
显示器	自动检测

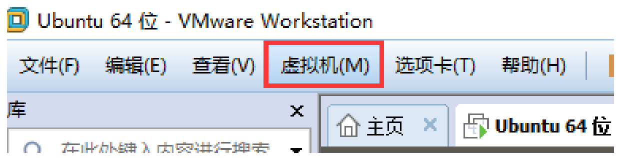
描述
在此处键入对该虚拟机的描述。


虚拟机设置

设备	摘要
内存	4 GB
处理器	8
硬盘 (SCSI)	20 GB
CD/DVD (SATA)	正在使用文件 ubuntu-16.04.3-...
网络适配器	NAT
USB 控制器	存在
声卡	自动检测
打印机	存在
显示器	自动检测

设备状态
☐ 已连接(C)
☑ 启动时连接(O)

连接
○ 使用物理驱动器(P):
自动检测
[✓ 使用 ISO 映像文件(M):]
ubuntu-16.04.3-desktop-am


Ubuntu 64 位 - VMware Workstation

Install Vmware Tools

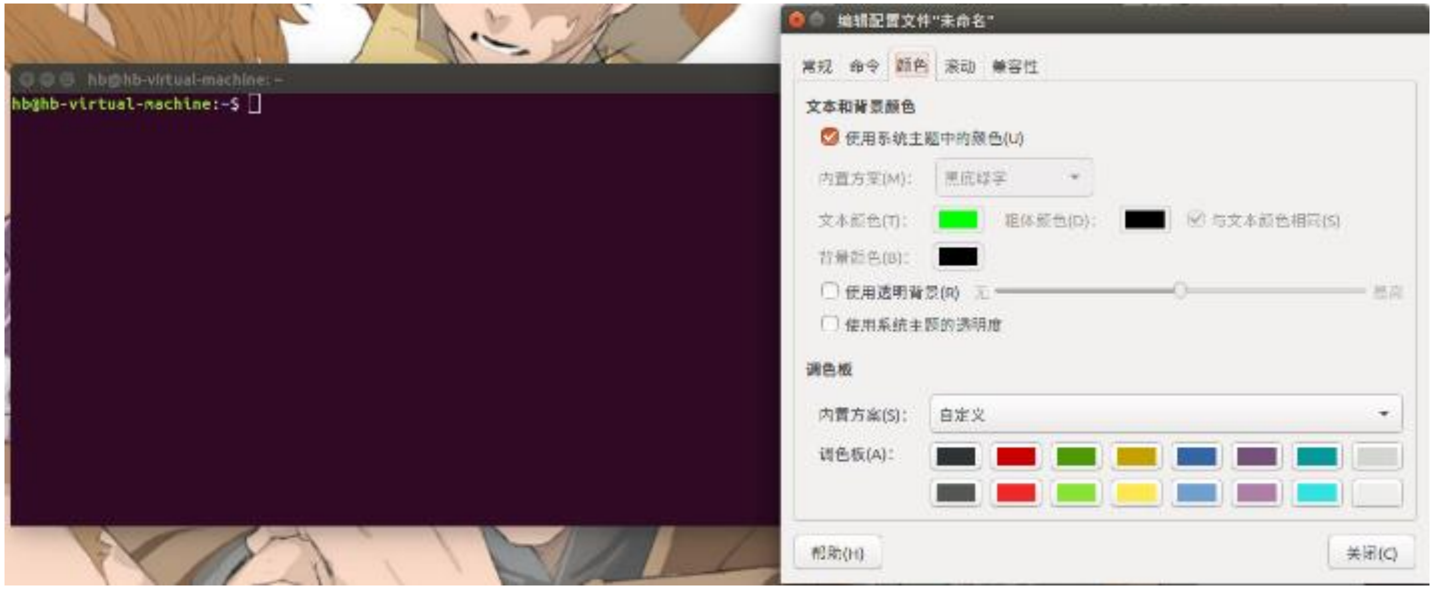
YOUR UBUNTU

Ubuntu 64 位 - VMware Workstation
文件(F) 编辑(E) 查看(V) 虚拟机(M) 选项卡(T) 帮助(H)
我的计算机
Ubuntu 64 位
共享的虚拟机
主页
Ubuntu 64 位
Ubuntu 桌面
16:03

要将输入定向到该虚拟机，请在虚拟机内部单击或按 Ctrl+G。

TERMINAL

- **Press Ctrl+Alt+t to open a terminal**
  - Press Ctrl+Alt+F1~F6 go to another world! (Ctrl+Alt+F7 to return)



```
hb@hb-virtual-machine: ~
hb@hb-virtual-machine:~$ 
```

编辑配置文件"未命名"

常规 | 命令 | 颜色 | 滚动 | 兼容性

文本和背景颜色

[✓] 使用系统主题中的颜色(U)

内置方案(M): 黑底绿字

文本颜色(T): 🟩 粗体颜色(D): ⬛ [✓] 与文本颜色相同(S)

背景颜色(B): ⬛

[ ] 使用透明背景(R) 无 — 最高

[ ] 使用系统主题的透明度

调色板

内置方案(S): 自定义

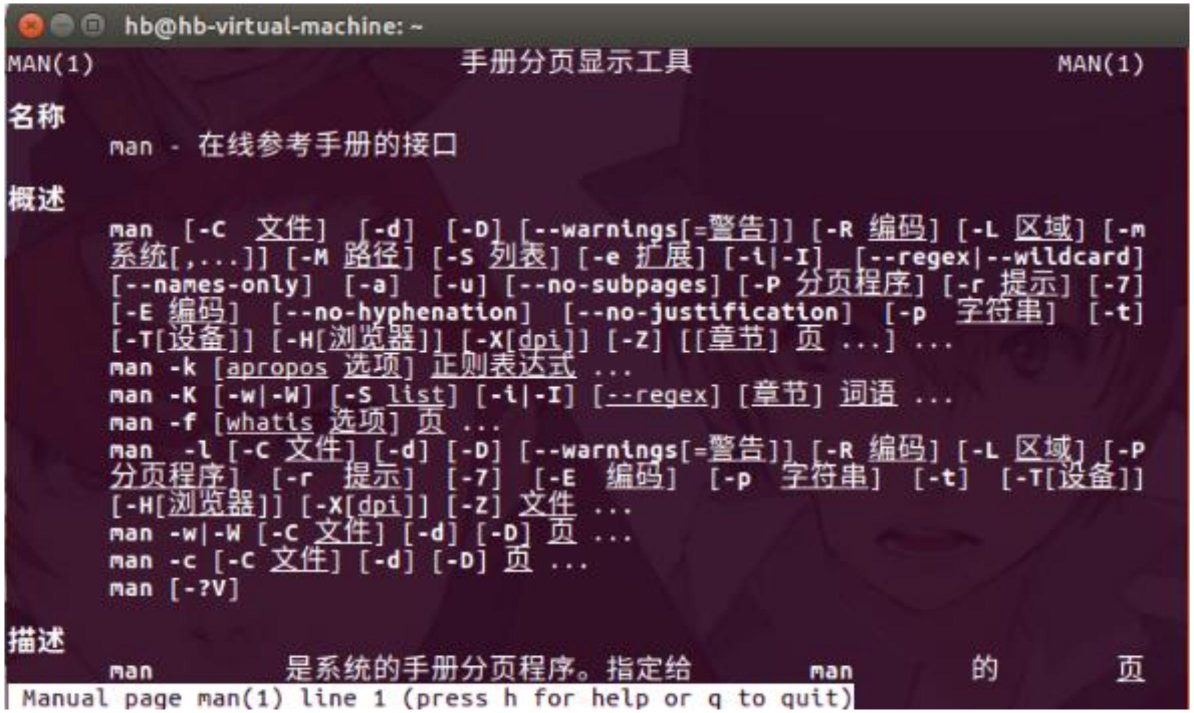
调色板(A):

BASIC COMMAND

Pictures from google

BASIC COMMAND

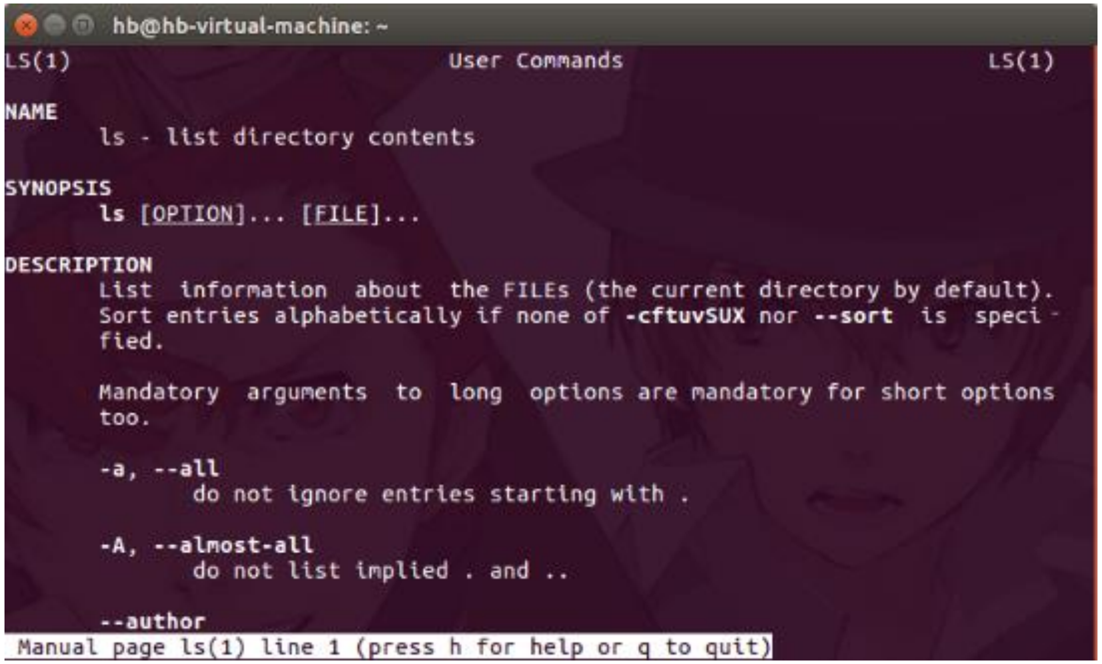
- **man xxx**
  - show the manual of command xxx
  - you can try “man man”



```
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"

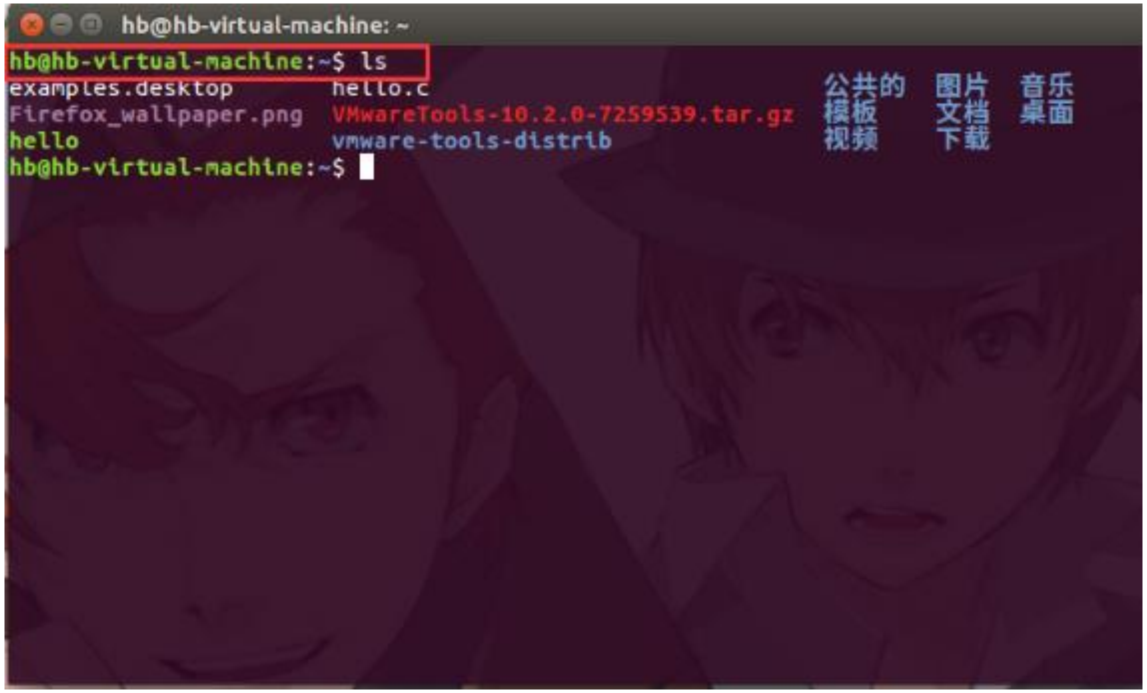


hb@hb-virtual-machine: ~

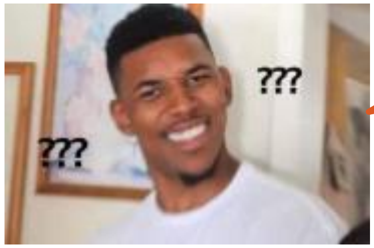
```
LS(1)                        User Commands                        LS(1)

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

- ls



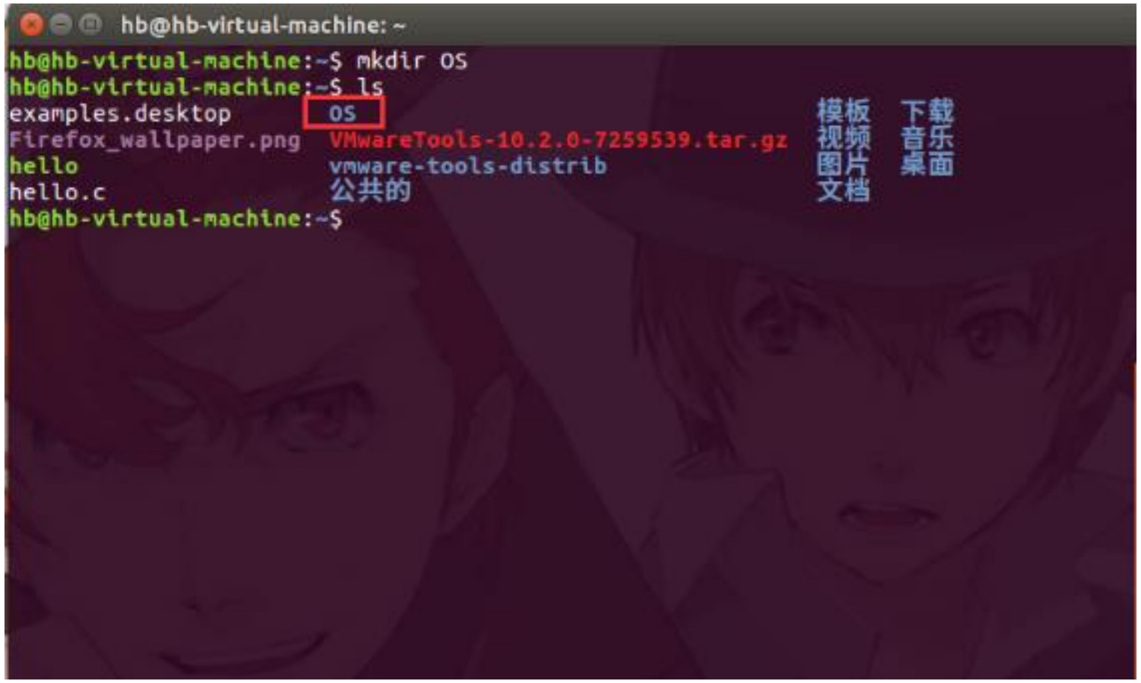
```
hb@hb-virtual-machine:~$ ls
examples.desktop       hello.c                              公共的  图片  音乐
Firefox_wallpaper.png  VMwareTools-10.2.0-7259539.tar.gz    模板    文档  桌面
hello                  vmware-tools-distrib                 视频    下载
hb@hb-virtual-machine:~$
```



- If I want to see the contents in Downloads?(下载)

BASIC COMMAND

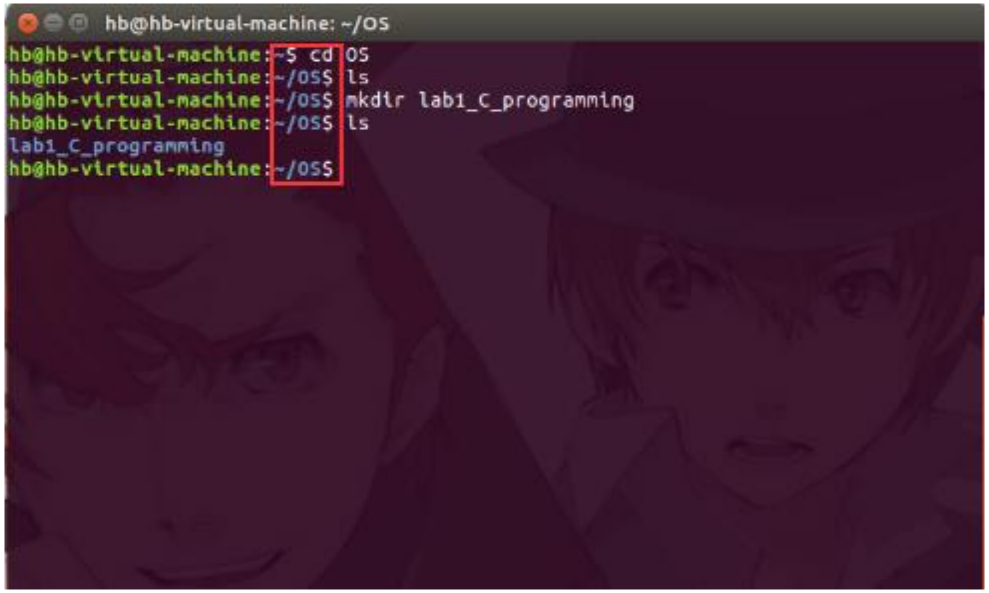
- **mkdir**
  - make a new directory
  - try man mkdir by yourself
  - try to make a directory named OS in HOME(~)



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

BASIC COMMAND

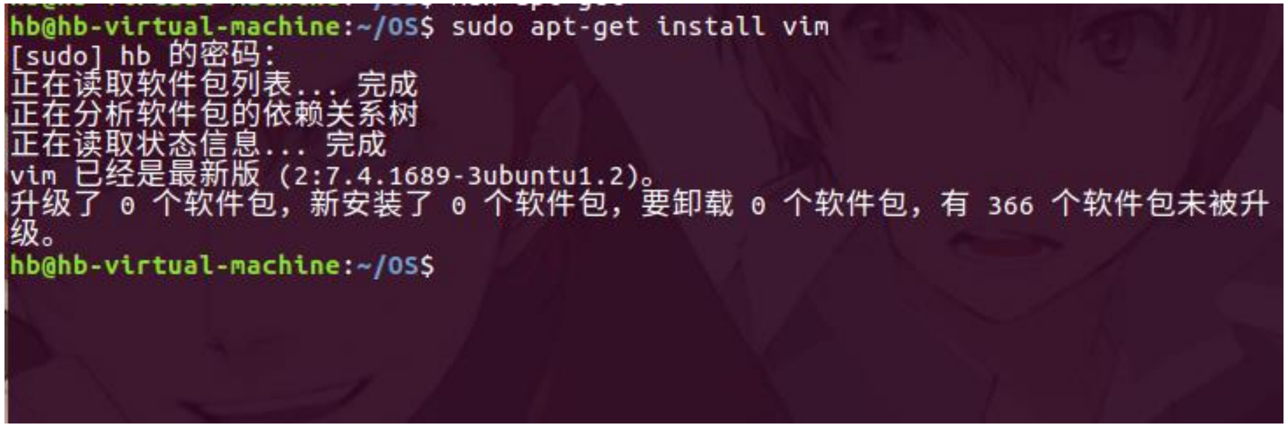
- **cd**
  - change directory
  - try man cd by yourself
  - let’s go to OS directory
  - then create a new directory called lab1_C_programming



```
hb@hb-virtual-machine: ~/OS
hb@hb-virtual-machine:~$ 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

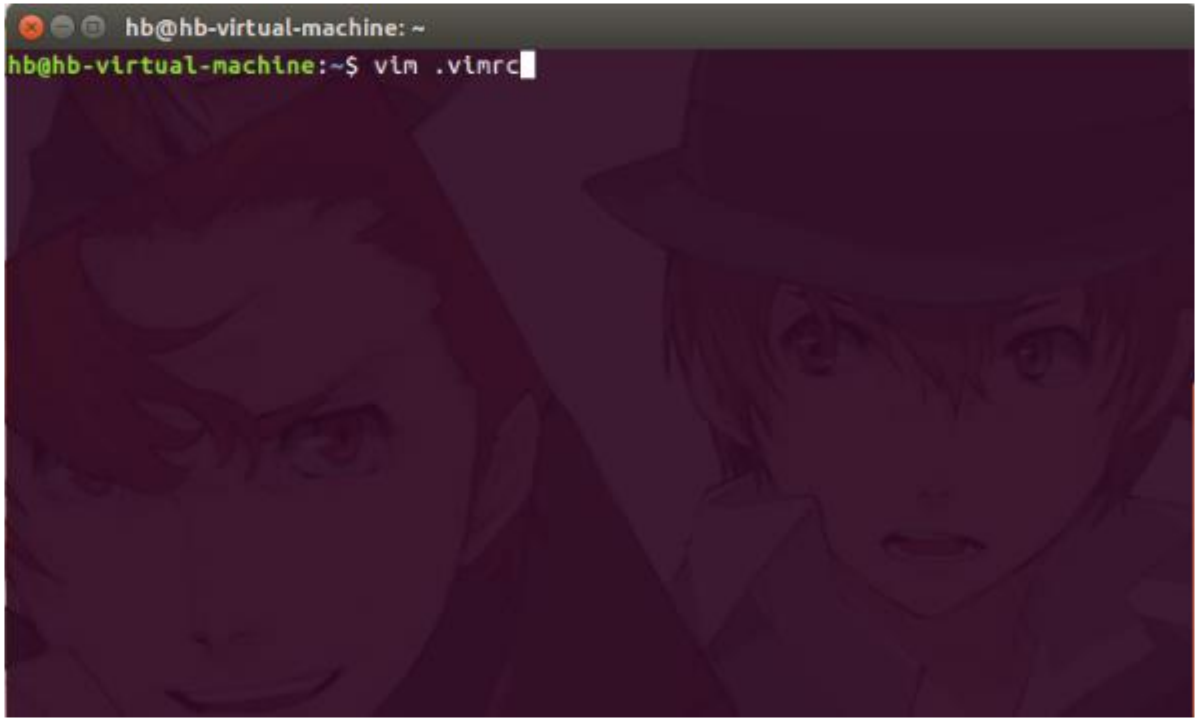
- **Server**
  - When you connect to a linux server, sometimes it doesn’t have GUI (e.g. X-window).

VIM

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



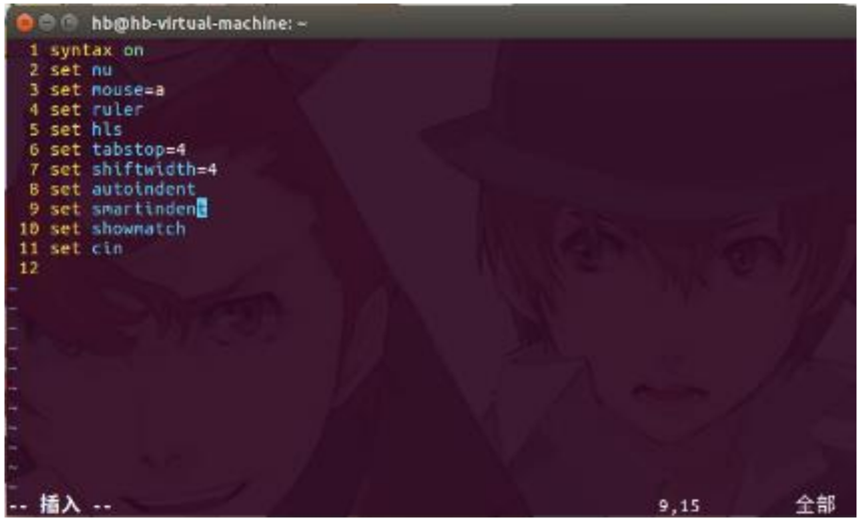
```
hb@hb-virtual-machine:~$ vim .vimrc
```

VIM

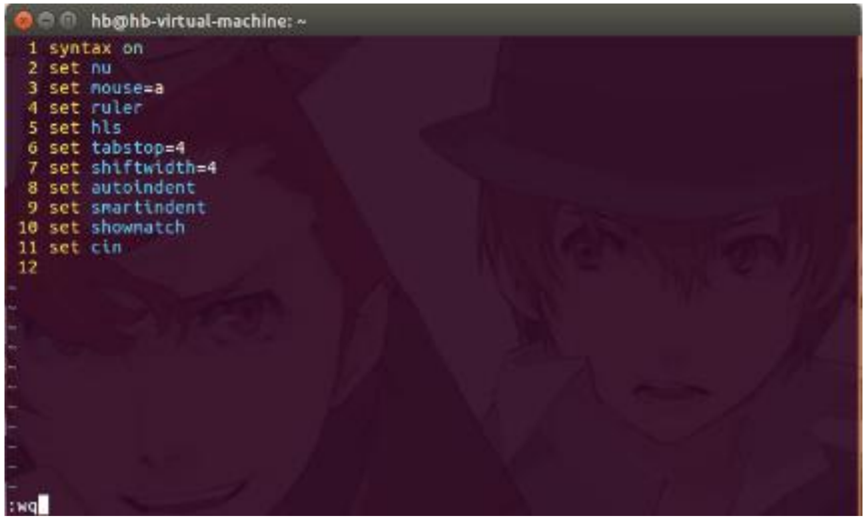
- **Vim has three modes, they are:**
  - Command mode: you can not input text, everything you input will be command.
  - Insert mode: you can input text. Press Esc to return command mode.
  - Last line mode: you can input special command. Such as exit and find string.

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

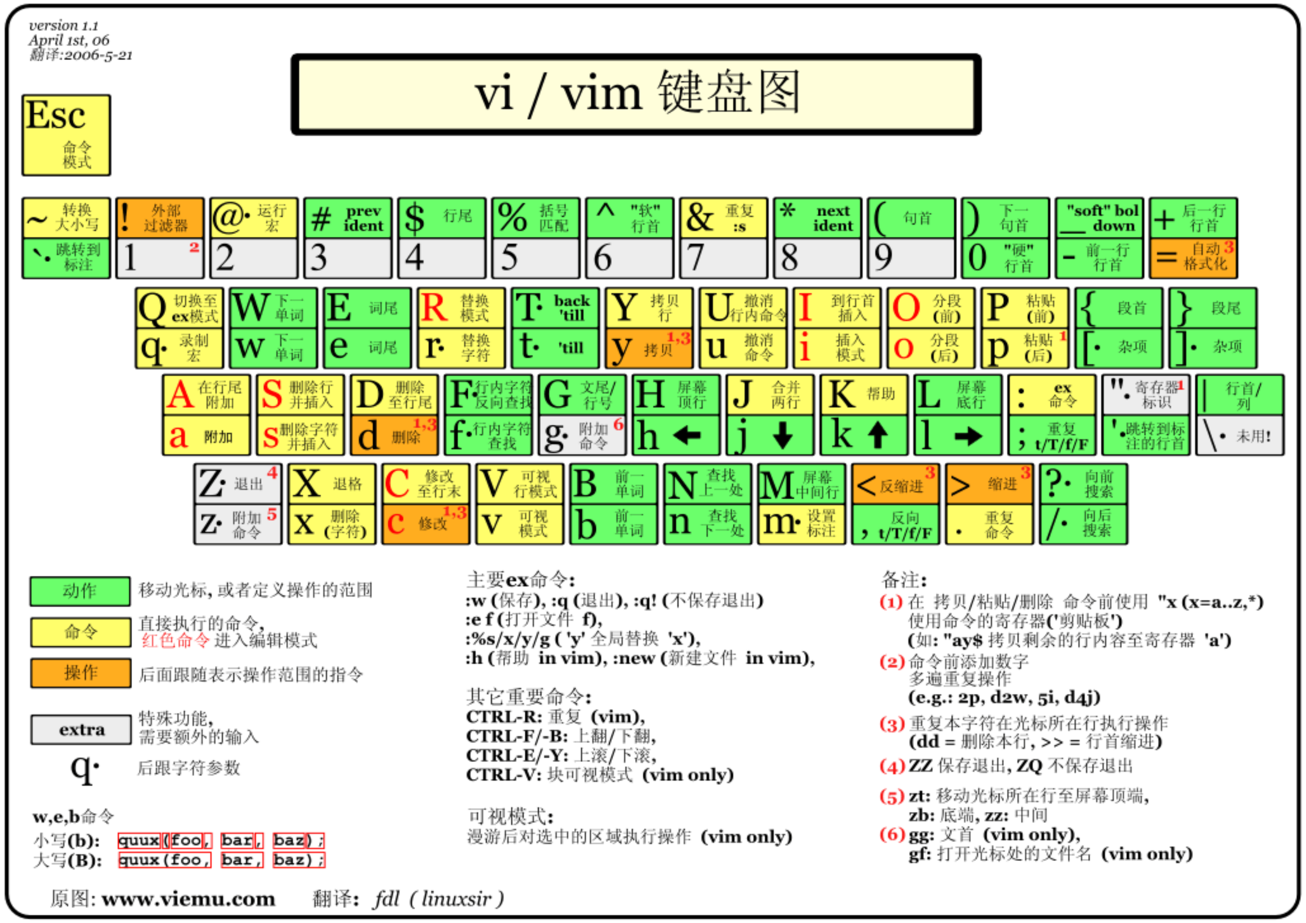


```
syntax on
set nu
set mouse=a
set ruler
set hls
set tabstop=4
set shiftwidth=4
set autoindent
set smartindent
set showmatch
set cin

-- 插入 --                                          9,15          全部
```



```
syntax on
set nu
set mouse=a
set ruler
set hls
set tabstop=4
set shiftwidth=4
set autoindent
set smartindent
set showmatch
set cin

:wq
```

YOU MAY NEED THIS



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

vi / vim 键盘图

🟩 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),

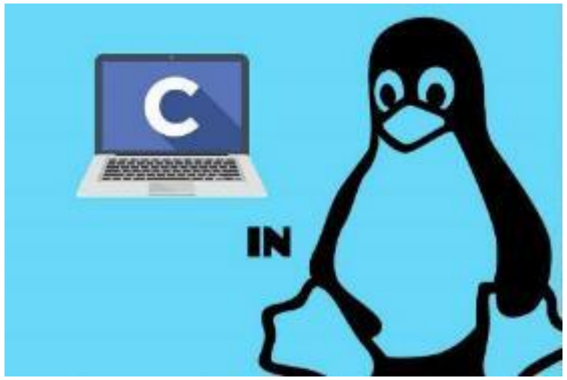
其它重要命令:
CTRL-R: 重复 (vim),
CTRL-F/-B: 上翻/下翻,
CTRL-E/-Y: 上滚/下滚,
CTRL-V: 块可视模式 (vim only)

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, 5i, 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)*

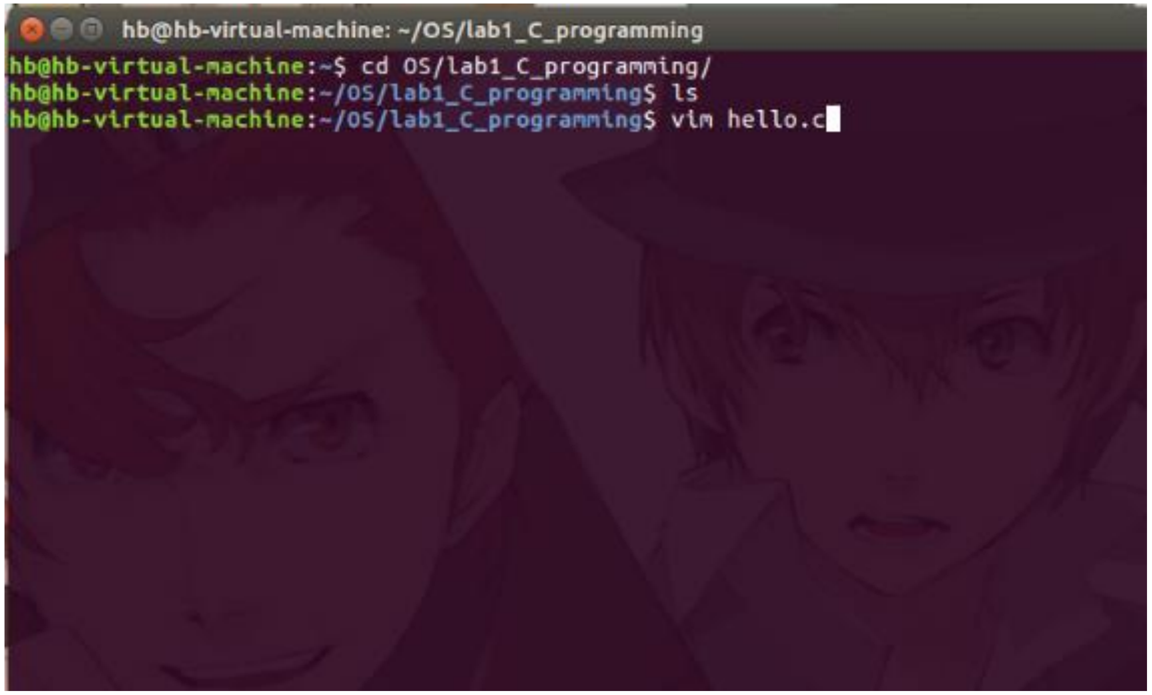
PROGRAMMING



Pictures from google

FIRST C PROGRAM

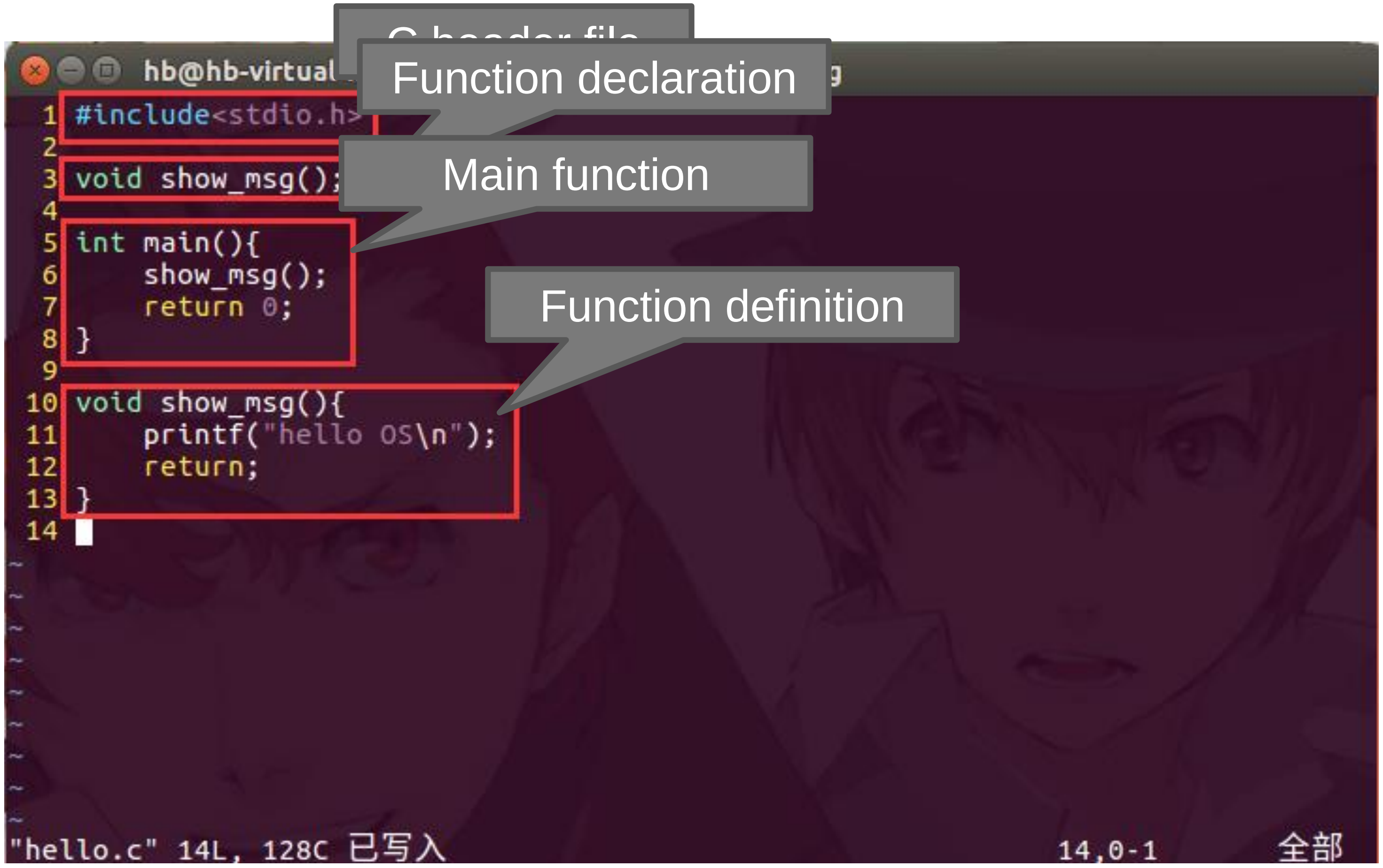
- **Now we are ready for our first C program.**
  - Go to lab1_C_programming directory
  - And edit file: hello.c



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
```

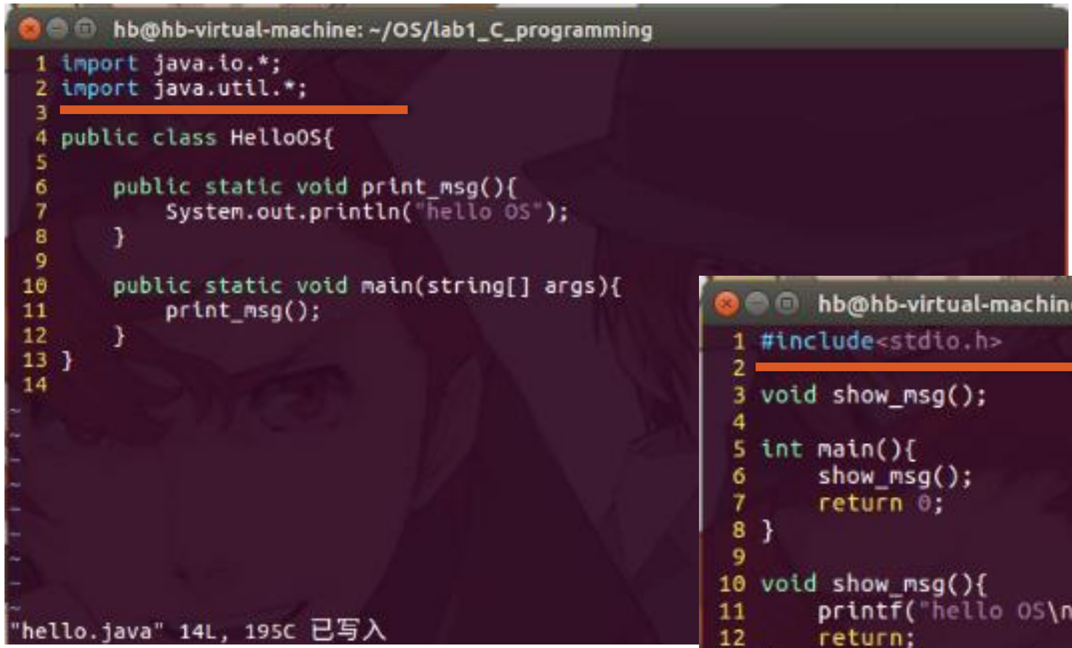
FIRST C PROGRAM



```
#include<stdio.h>

void show_msg();

int main(){
        show_msg();
        return 0;
}

void show_msg(){
        printf("hello OS\n");
        return;
}

```

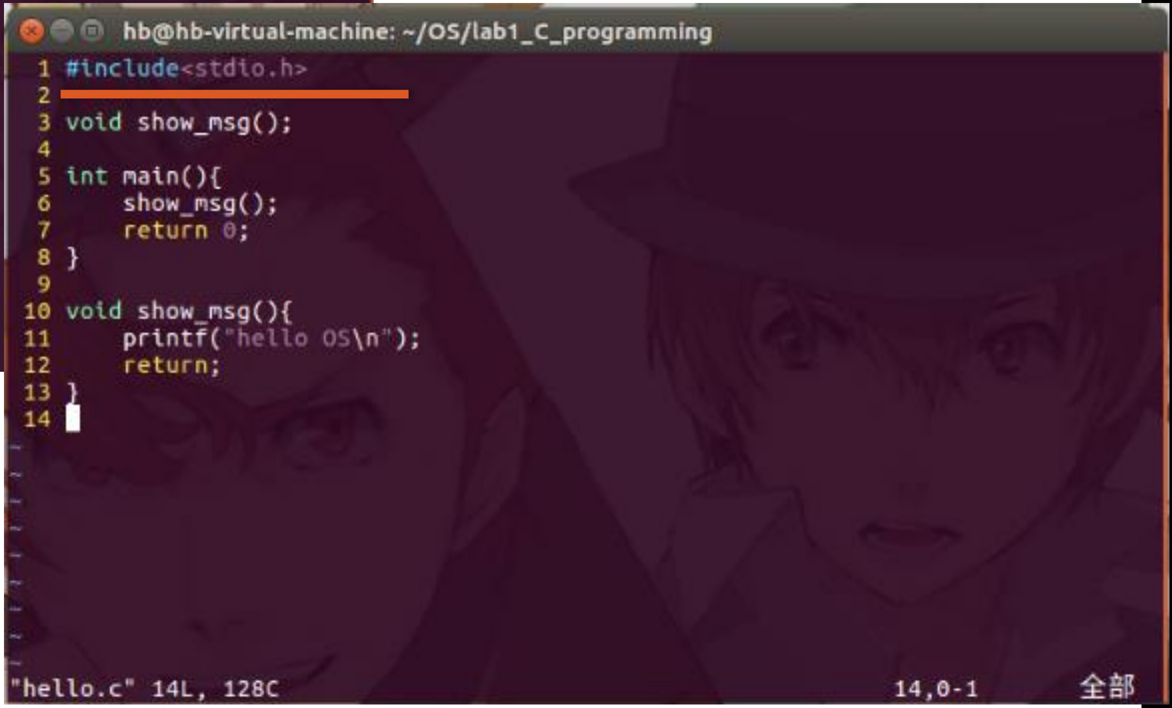
```
"hello.c" 14L, 128C 已写入                                    14,0-1        全部
```

A JAVA PROGRAM



hb@hb-virtual-machine: ~/OS/lab1_C_programming

```
import java.io.*;
import java.util.*;

public class HelloOS{

    public static void print_msg(){
        System.out.println("hello OS");
    }

    public static void main(string[] args){
        print_msg();
    }
}

~
~
~
~
~
~
~
~
~
"hello.java" 14L, 195C 已写入
```

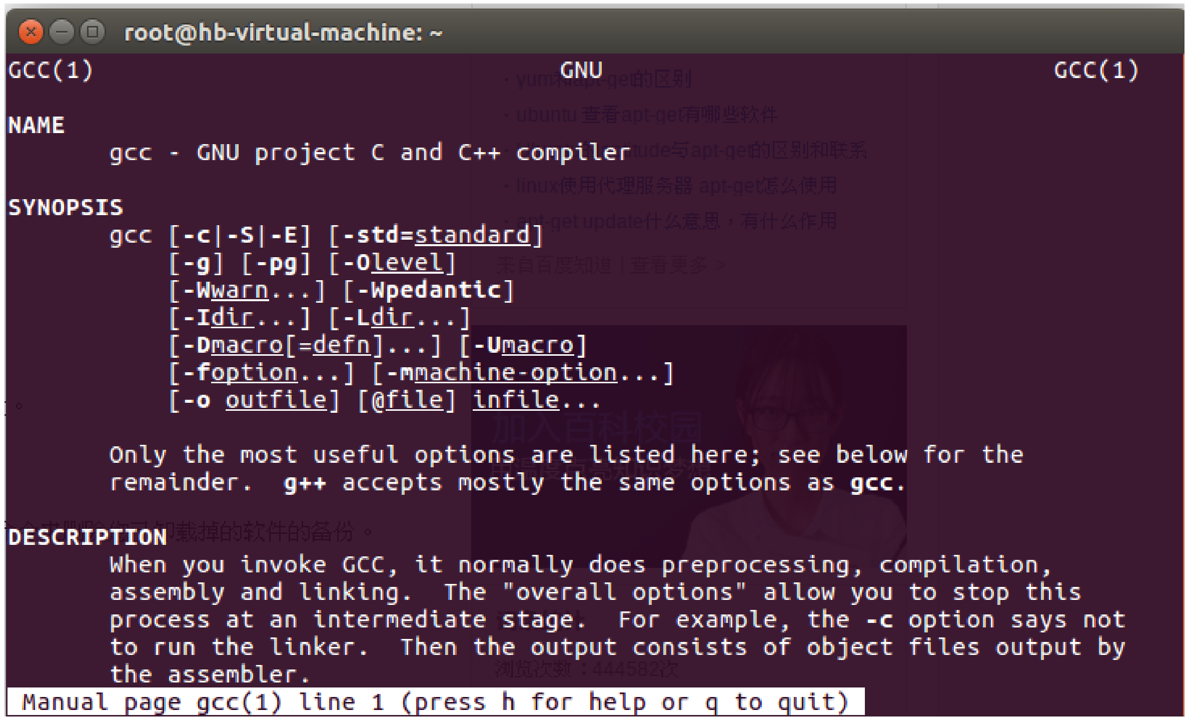


hb@hb-virtual-machine: ~/OS/lab1_C_programming

```
#include<stdio.h>

void show_msg();

int main(){
        show_msg();
        return 0;
}

void show_msg(){
        printf("hello OS\n");
        return;
}

~
~
~
~
~
~
~
~
~
"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



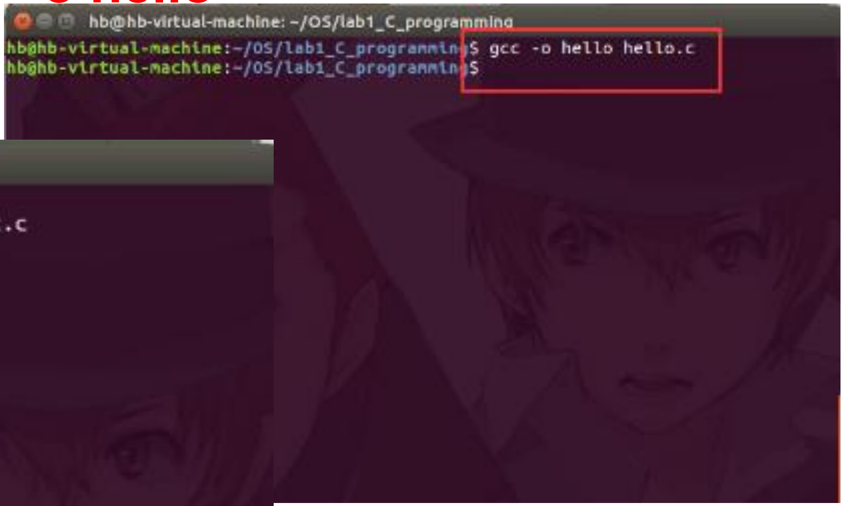
```
root@hb-virtual-machine: ~
GCC(1)                             GNU                              GCC(1)

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

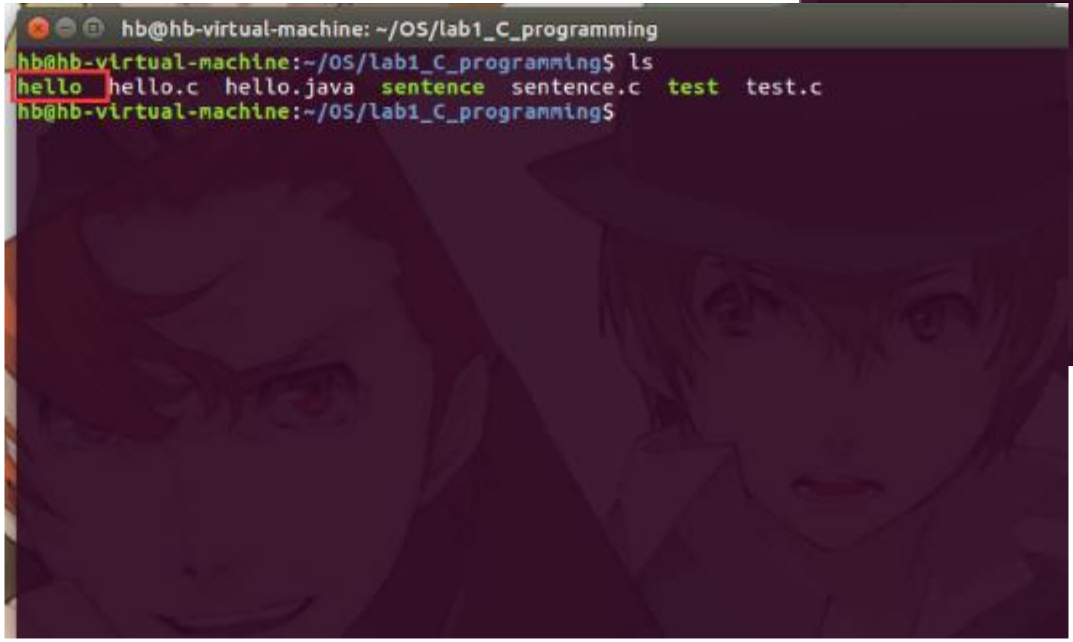
- **gcc**
  - -c Compile or assemble the source files, but do not link.
  - -S Stop after the stage of compilation proper; do not assemble.
  - -E Stop after the preprocessing stage; do not run the compiler proper.
  - -o filename Place output in file file.
  - If no parameters, gcc will do all things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



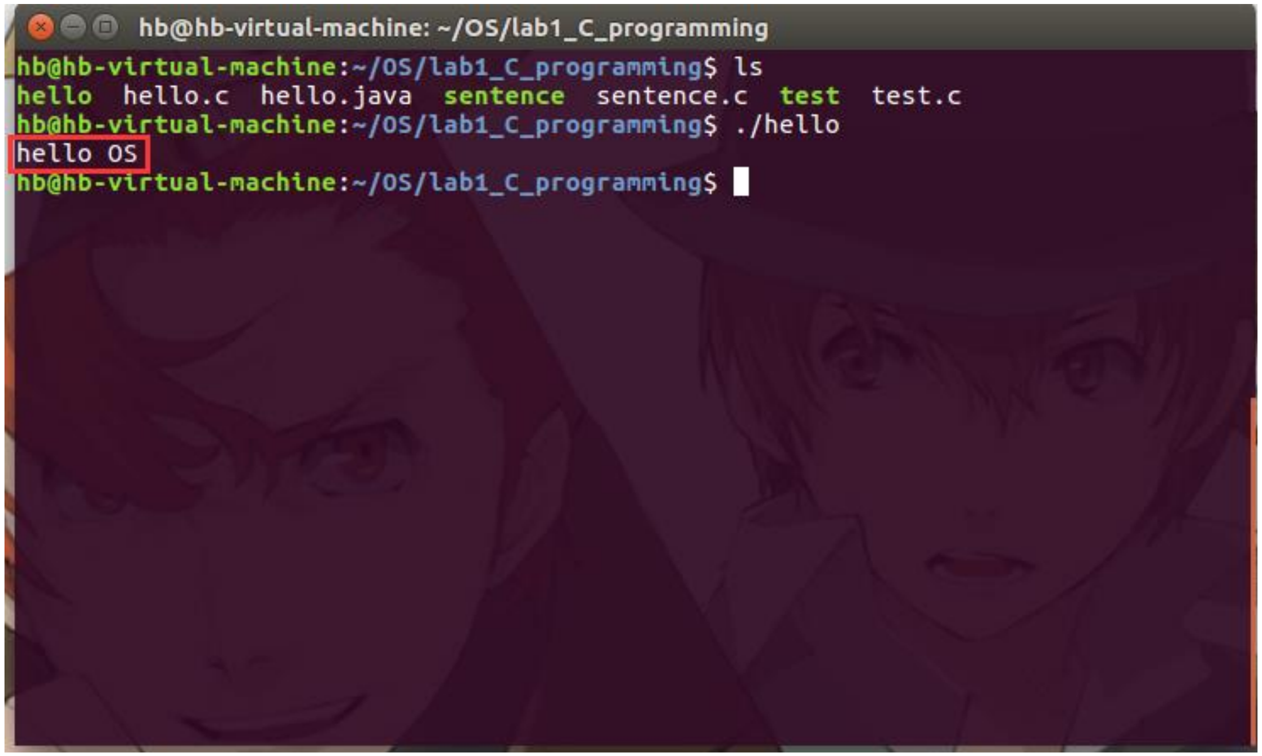
```
hb@hb-virtual-machine: ~/OS/lab1_C_programming
hb@hb-virtual-machine:~/OS/lab1_C_programming$ gcc -o hello hello.c
hb@hb-virtual-machine:~/OS/lab1_C_programming$
```



```
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$
```

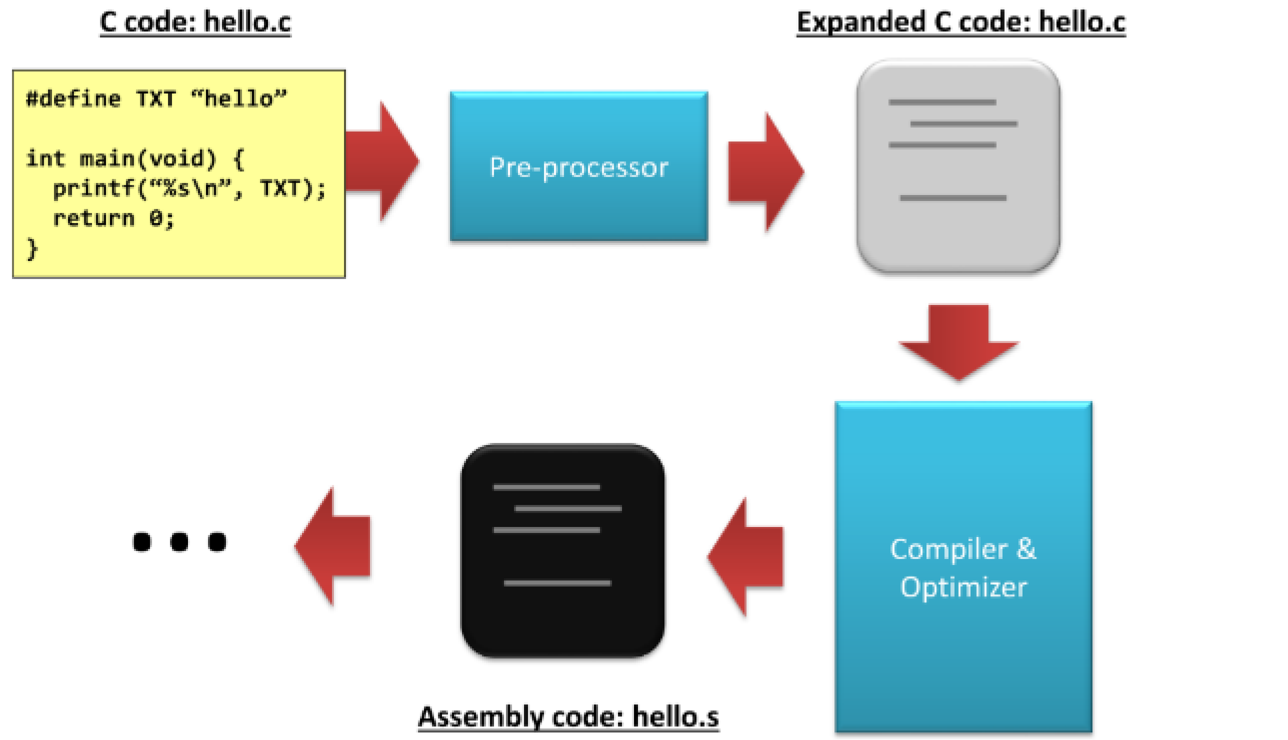
FIRST C PROGRAM

- **Let's run it!**
  - ./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$ ./hello
hello OS
hb@hb-virtual-machine:~/OS/lab1_C_programming$ 
```

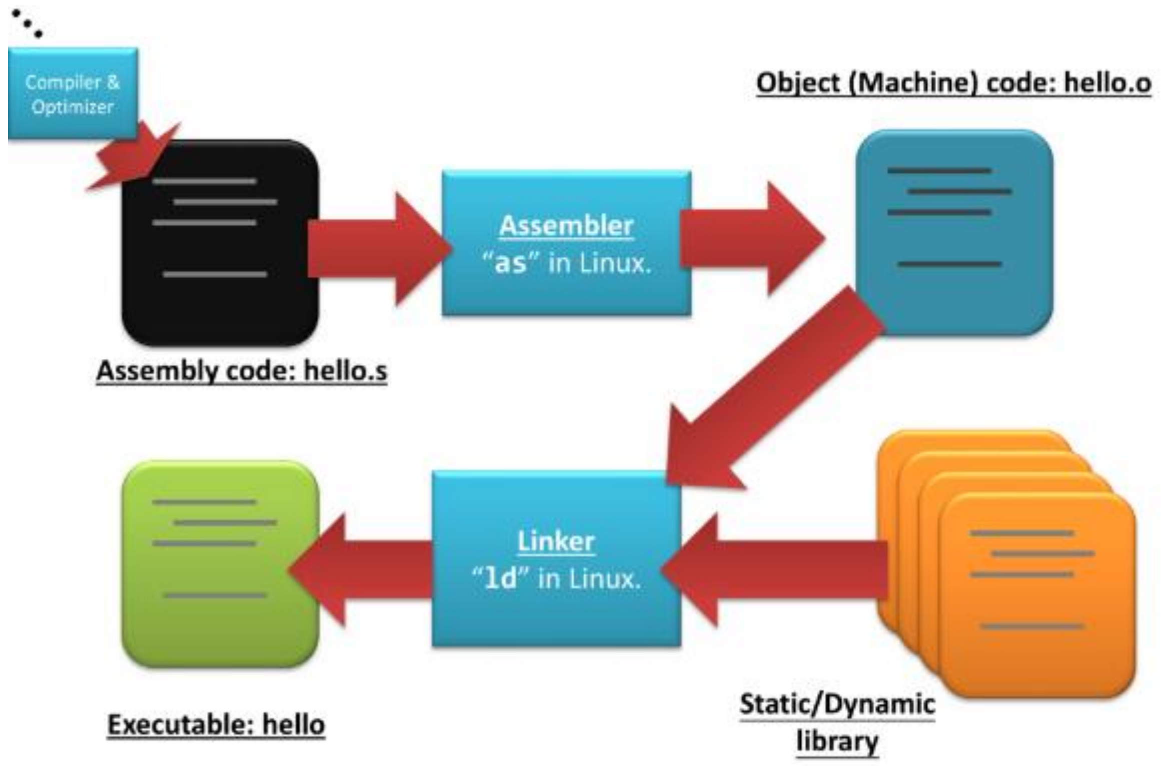
WHAT HAPPENS?



```
#define TXT "hello"

int main(void) {
  printf("%s\n", TXT);
  return 0;
}
```

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

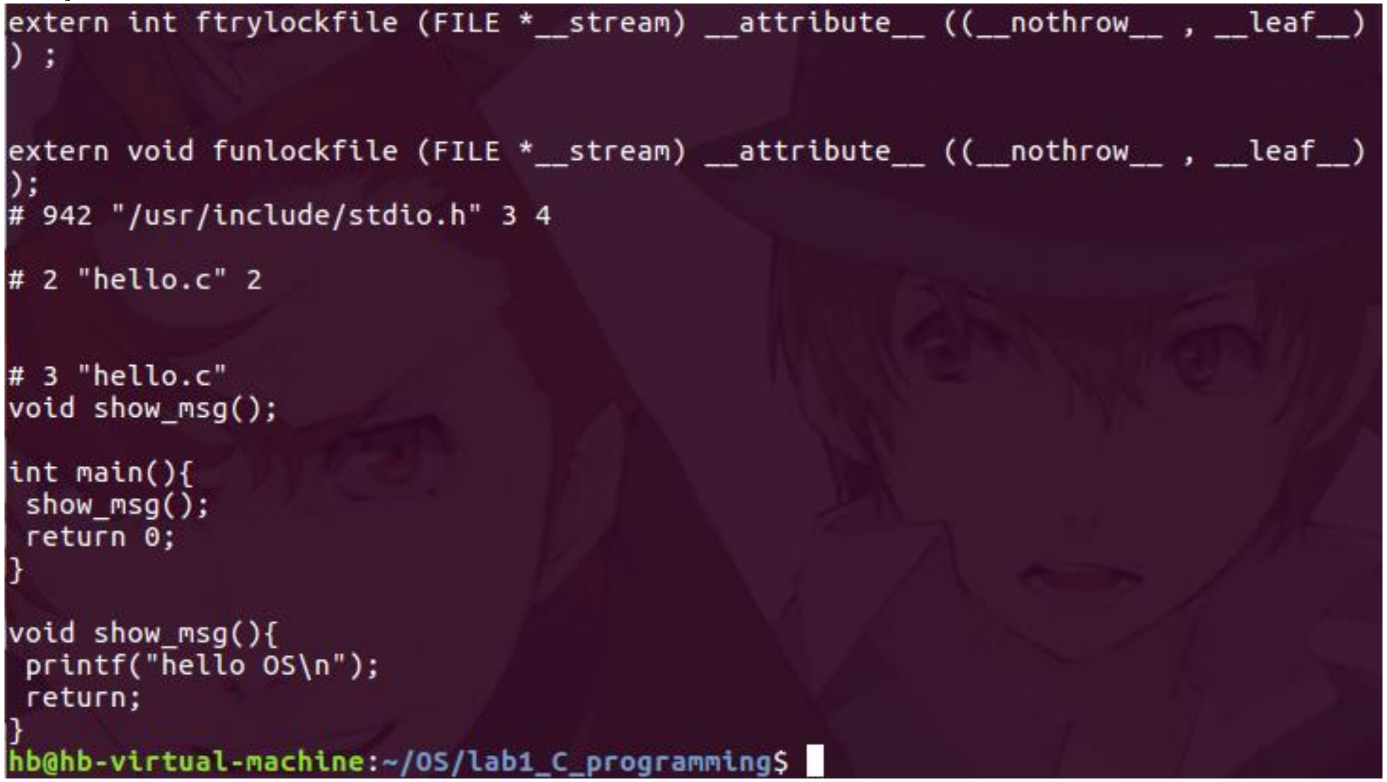
WHAT HAPPENS?



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

WHAT HAPPENS?

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



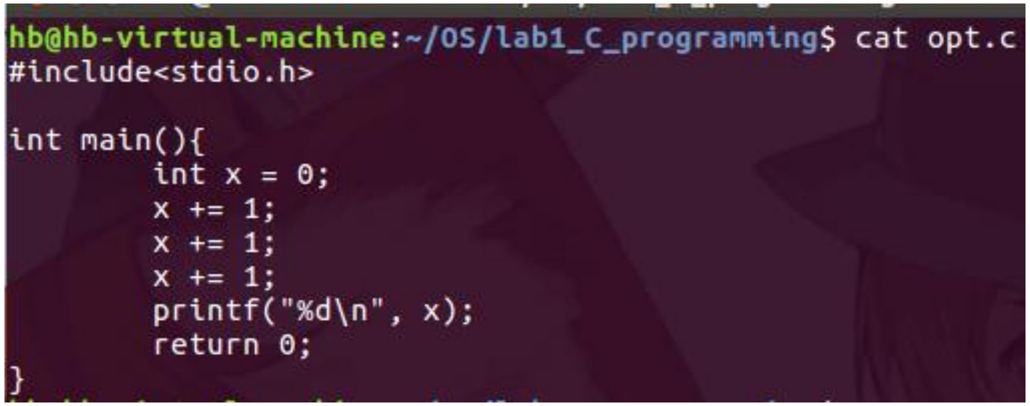
```
extern int ftrylockfile (FILE *__stream) __attribute__ ((__nothrow__ , __leaf__)
) ;


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.

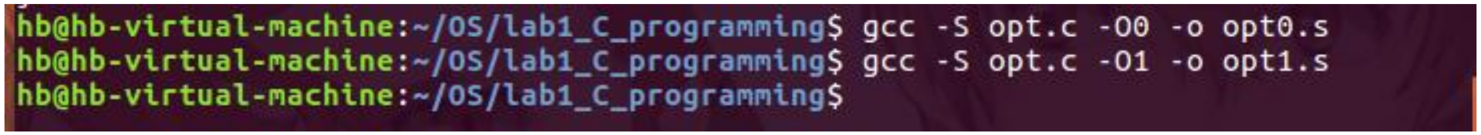
MORE ABOUT OPTIMIZER

- 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;
}
```

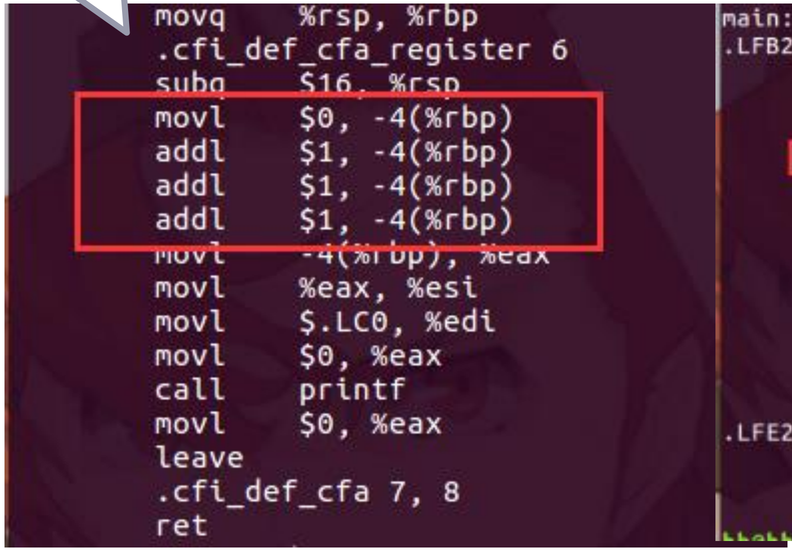
- We open the optimizer to get assembly code



```
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$
```

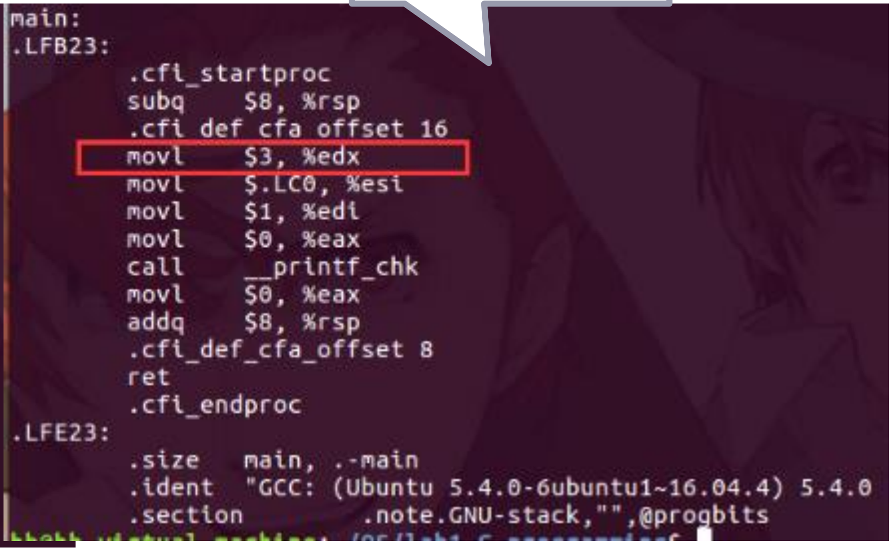
MORE ABOUT OPTIMIZER

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
```

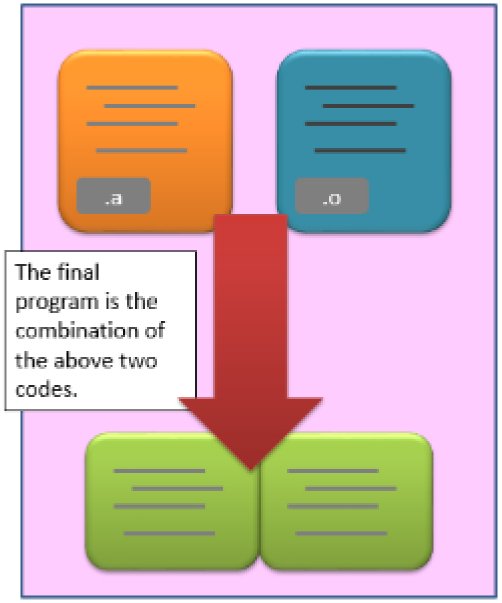
opt1.s

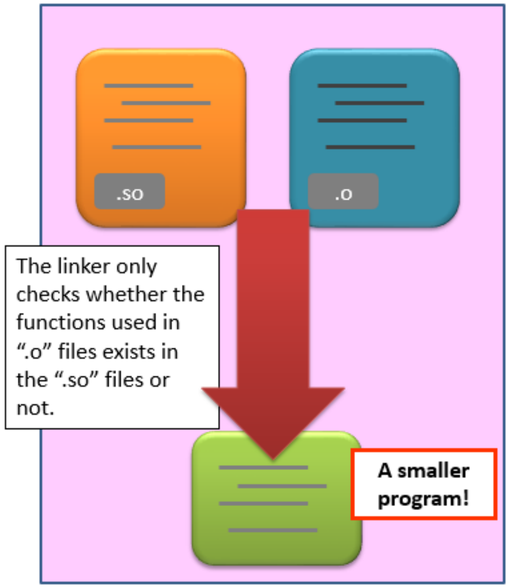


```
main:
.LFB23:
        .cfi_startproc
        subq    $8, %rsp
        .cfi_def_cfa_offset 16
        movl    $3, %edx
        movl    $.LC0, %esi
        movl    $1, %edi
        movl    $0, %eax
        call    __printf_chk
        movl    $0, %eax
        addq    $8, %rsp
        .cfi_def_cfa_offset 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.


.a
.o
The final program is the combination of the above two codes.


.so
.o
The linker only checks whether the functions used in ".o" files exists in the ".so" files or not.
A smaller program!

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

C LANGUAGE

Pictures from google

MORE ABOUT C

- **Data type**

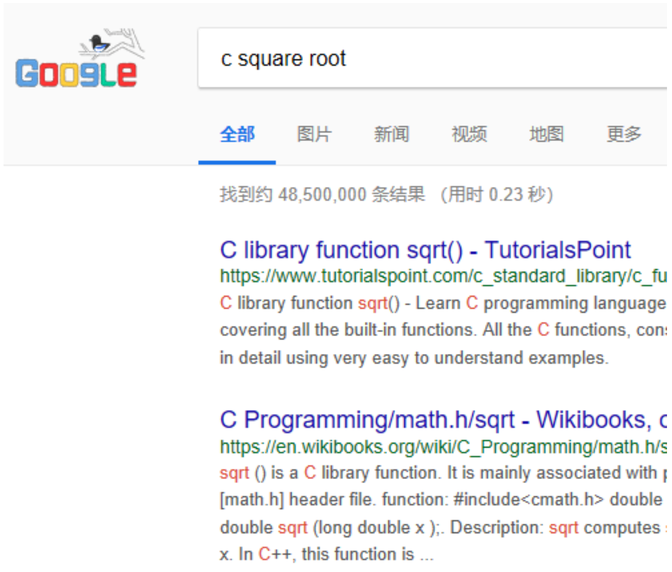
java	C	data
char	char	A character
boolean	bool	Ture or False
int	int	32-bit integer
long	long long	64-bit integer
float	float	32-bit float
double	double	64-bit float
<T>[]	<T>[]	array
String	char*	string

MORE ABOUT C

- **Sentence**
  - if … else …
  - while
  - for
  - do while
  - switch
  - ……

MORE ABOUT C

- **Libraries**
  - math.h
  - algorithm
  - stdlib.h
  - ……
  - When you need a special function, you can search them on the internet
  - If you like, you can write your own.


c square root

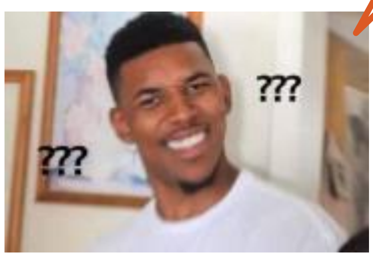
找到约 48,500,000 条结果 （用时 0.23 秒）

C library function sqrt() - TutorialsPoint
https://www.tutorialspoint.com/c_standard_library/c_fu
C library function sqrt() - Learn C programming language
covering all the built-in functions. All the C functions, con
in detail using very easy to understand examples.

C Programming/math.h/sqrt - Wikibooks, c
https://en.wikibooks.org/wiki/C_Programming/math.h/s
sqrt () is a C library function. It is mainly associated with
[math.h] header file. function: #include<cmath.h> double
double sqrt (long double x);. Description: sqrt computes
x. In C++, this function is ...

EXAMPLE

I want a program which can solve the following problem. I will input two integers x and y. (0 <= x, y <= 10) If x = 0, then calculate the summation from 0 to y. Else calculate the square root of y.(integer part is enough)

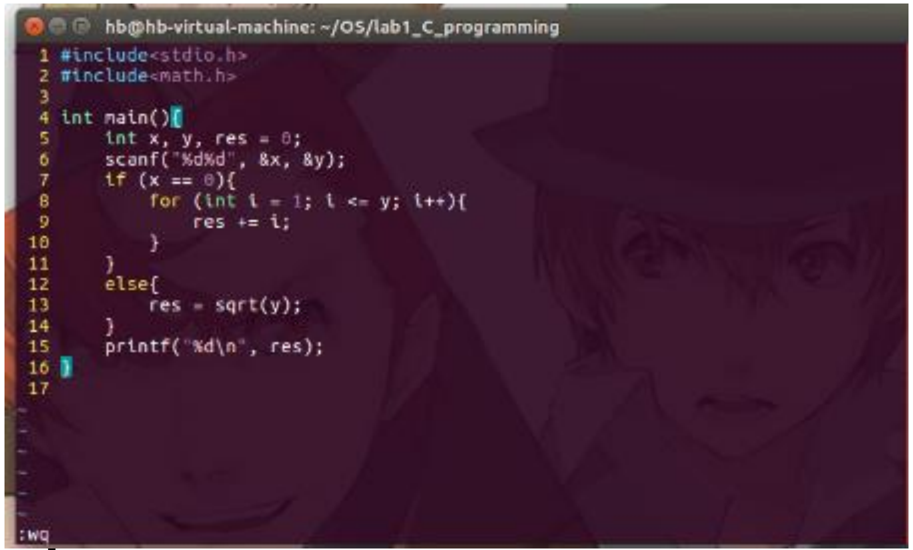


EXAMPLE

- **OK, Let’s try it!**
  - Using “ if ” to check whether x is 0. And then using “ for ” loop to calculate the summation. Using square root function (we find in by google)
  - Let’s try this.

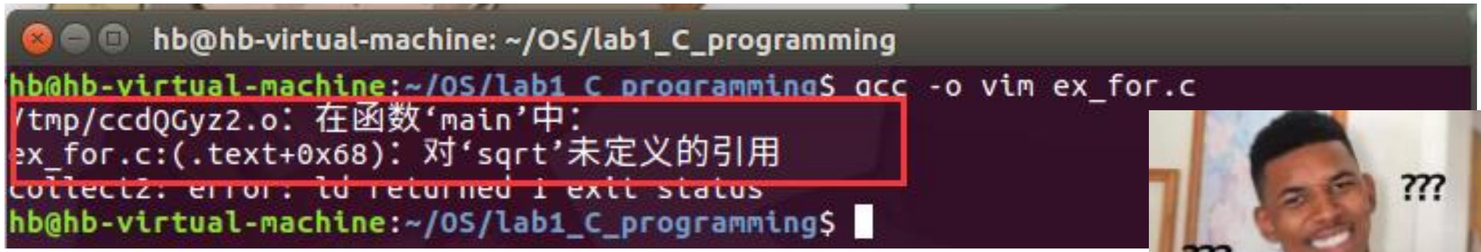
EXAMPLE

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

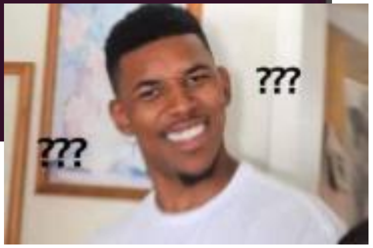


```c
#include<stdio.h>
#include<math.h>

int main(){
    int x, y, res = 0;
    scanf("%d%d", &x, &y);
    if (x == 0){
        for (int i = 1; i <= y; i++){
            res += i;
        }
    }
    else{
        res = sqrt(y);
    }
    printf("%d\n", res);
}

~
~
~
~
~
:wq
```

- Then compile it.

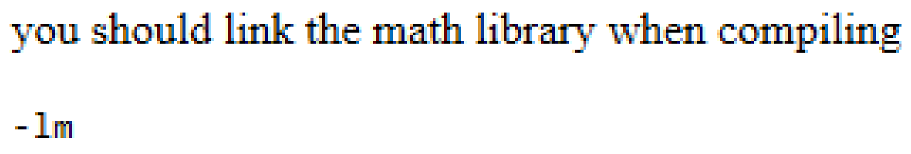


```
hb@hb-virtual-machine:~/OS/lab1_C_programming$ gcc -o vim ex_for.c
/tmp/ccdQGyz2.o: 在函数‘main’中:
ex_for.c:(.text+0x68): 对‘sqrt’未定义的引用
collect2: error: ld returned 1 exit status
hb@hb-virtual-machine:~/OS/lab1_C_programming$
```



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!

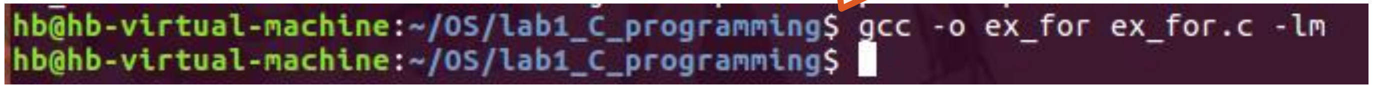

you should link the math library when compiling

```
-lm
```

EXAMPLE

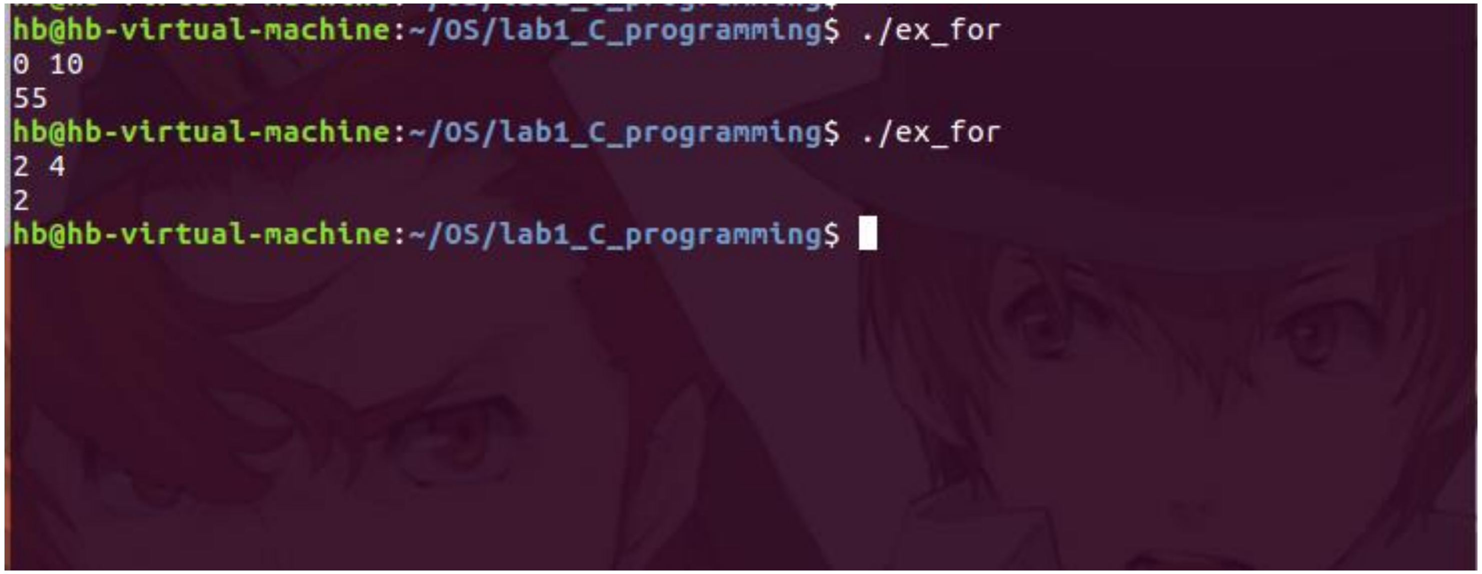
OOPs, it works. Thanks to stackoverflow!

- Let’s try again.



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

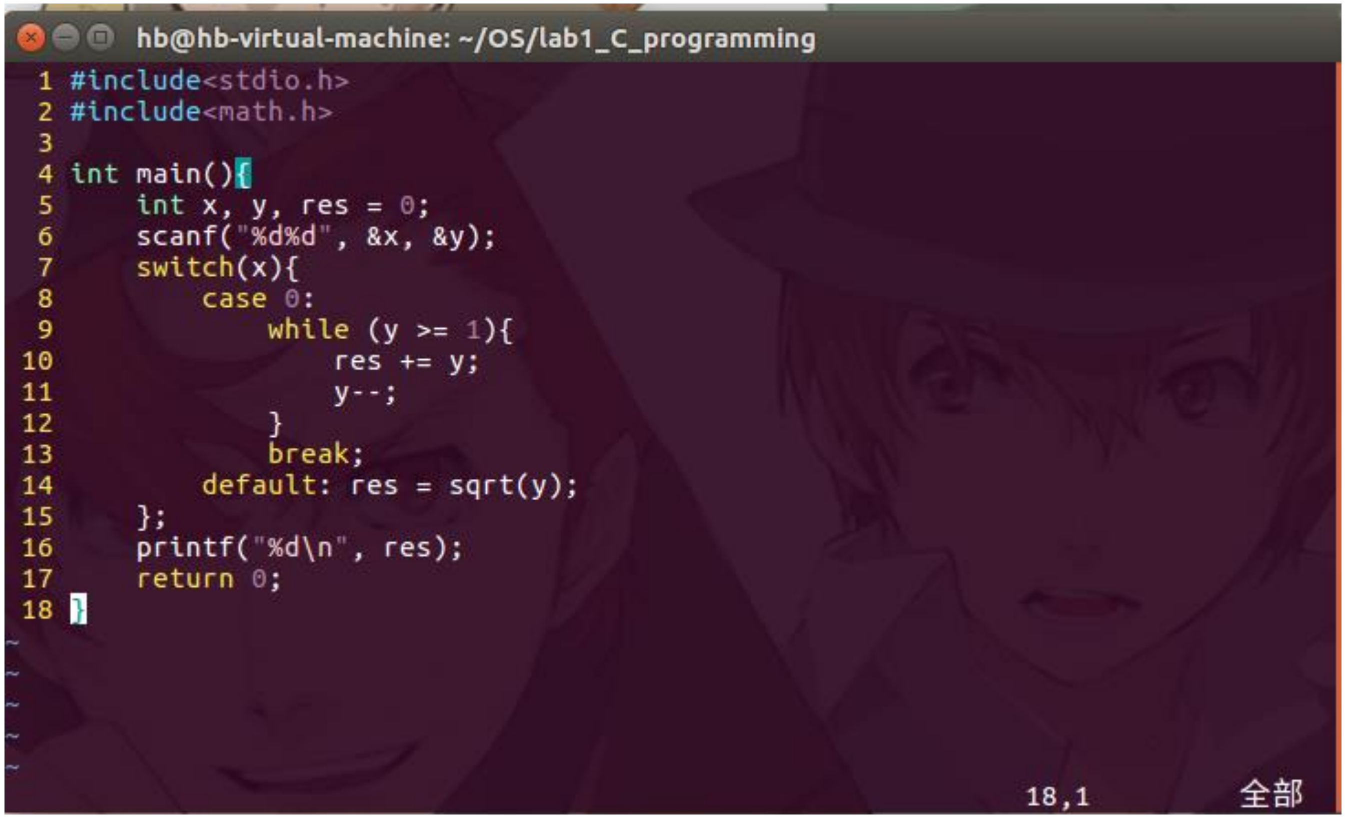
- And it works!



```
hb@hb-virtual-machine:~/OS/lab1_C_programming$ ./ex_for
0 10
55
hb@hb-virtual-machine:~/OS/lab1_C_programming$ ./ex_for
2 4
2
hb@hb-virtual-machine:~/OS/lab1_C_programming$ 
```

EXAMPLE

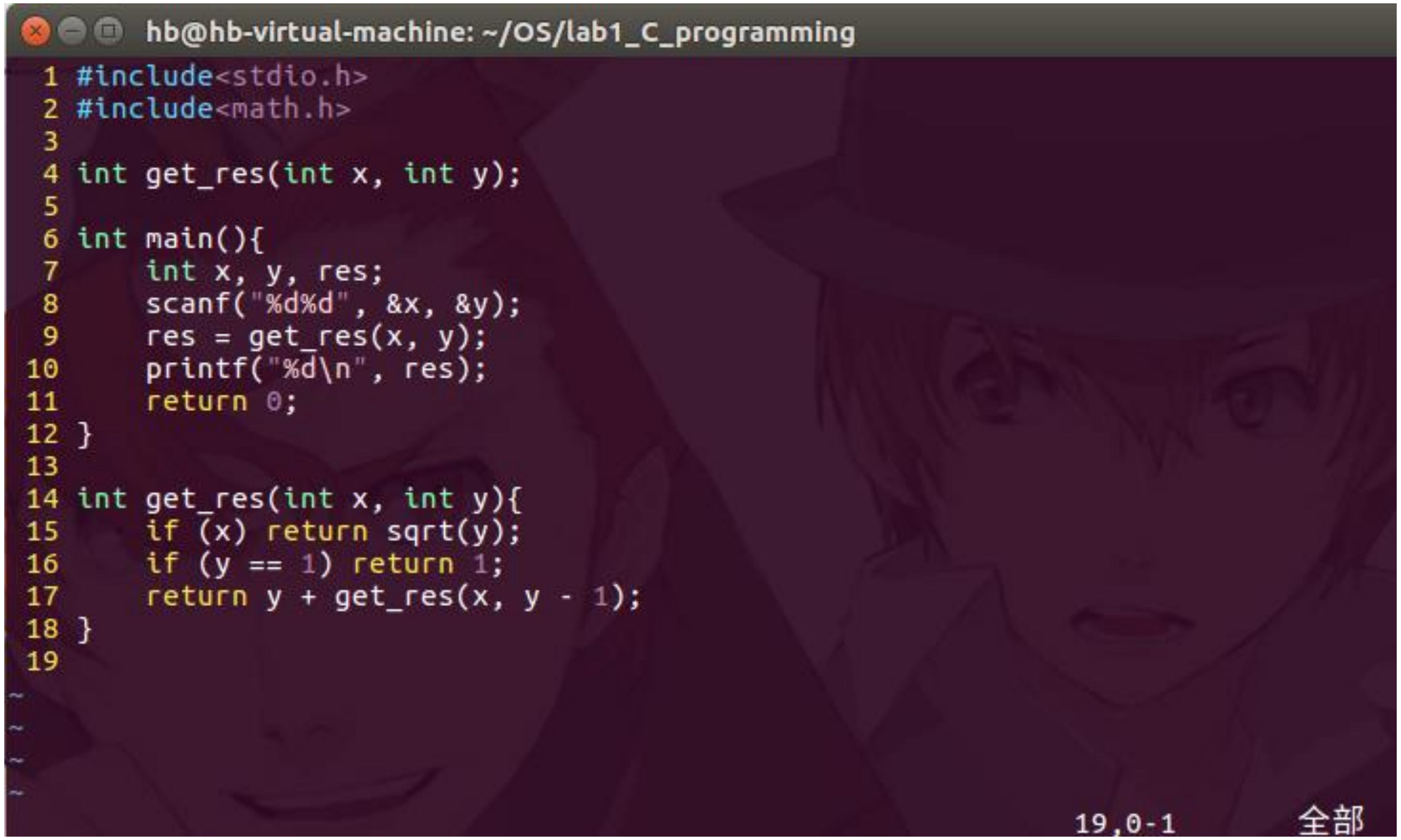
- **We can also replace if with switch, and replace for with while.**



```
#include<stdio.h>
#include<math.h>

int main(){
    int x, y, res = 0;
    scanf("%d%d", &x, &y);
    switch(x){
        case 0:
            while (y >= 1){
                res += y;
                y--;
            }
            break;
        default: res = sqrt(y);
    };
    printf("%d\n", res);
    return 0;
}
~
~
~
~
~
                                                  18,1          全部
```

EXAMPLE

- **How about using functions?**


hb@hb-virtual-machine: ~/OS/lab1_C_programming

```
#include<stdio.h>
#include<math.h>

int get_res(int x, int y);

int main(){
    int x, y, res;
    scanf("%d%d", &x, &y);
    res = get_res(x, y);
    printf("%d\n", res);
    return 0;
}

int get_res(int x, int y){
    if (x) return sqrt(y);
    if (y == 1) return 1;
    return y + get_res(x, y - 1);
}

~
~
~
~
                                                    19,0-1        全部
```

EXERCISE

Pictures from google

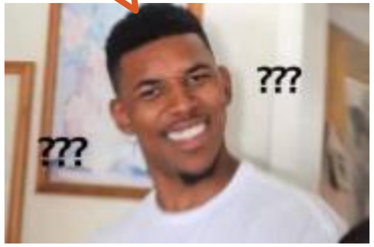
EXERCISE

- **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

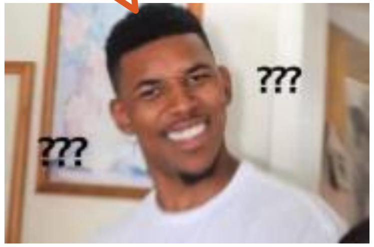
I will input 20 integers, please calculate the maximum, minimum and the average number of these 20 integers.



EXERCISE

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

I will input n integers,(1 <= n <= 100), please sort them by ascending order.

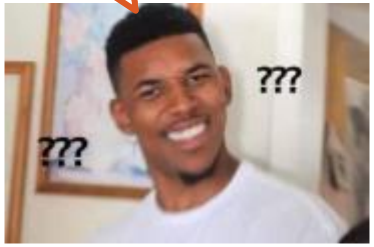


EXERCISE

- **Be aware**
  - spaces

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

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

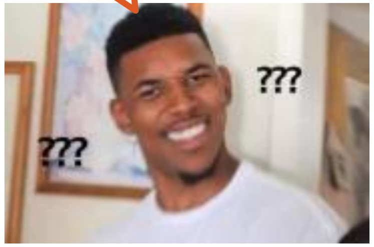


EXERCISE

- **Be aware**
  - directions

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

```
123
894
765
```



THANK YOU