

# 第1章 程序设计和C语言



# Who am I

Ø 郑炜 老师

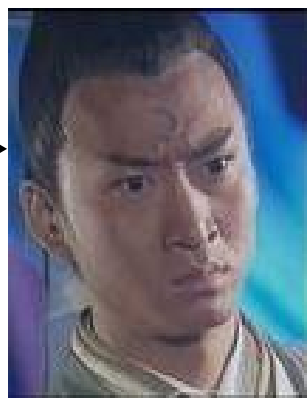
- ▣ 2011- : 工作于厦门大学计算机科学系 ;
- ▣ 2005-2010 : 就读于英国曼彻斯特大学计算机学院 ( 博士 )
- ▣ 2002-2005 : 就读于清华 ( 庆华 ? ) 大学计算机系 ( 硕士 )
- ▣ 1998-2002 : 就读于清华 ( 庆华 ? ) 大学计算机系 ( 本科 )

Ø 联系方式 : [zhengw@xmu.edu.cn](mailto:zhengw@xmu.edu.cn)

- ▣ 欢迎各种提问/各种建议
- ▣ 欢迎交流感悟/探讨人生

there are no  
Dumb Questions

# 大学生生活轨迹



军训后

勤奋路线



学习中

颓废路线



课堂上



毕业后



梦醒时分



考试结束

# 你的大学目标？



# 课程基本信息

## Ø 使用教材

谭浩强《C程序设计》（第四版）

## Ø 课程结构

#1 程序设计和C语言

#2 计算机中的数

#3 顺序结构程序设计

#4 选择结构程序设计

#5 循环结构程序设计

#6 利用数组处理批量数据

#7 利用函数实现模块化程序设计

#8 善于利用指针

#9 用户自己建立数据类型

#10 对文件的输入输出

## Ø 上课方式

理论课：复习回顾，讲解新内容

今天开始，每周二 / 周五一二节

**实验课**：上机练习，各种答疑

第六周开始，每周三晚上

## Ø 考核方式

平时成绩 30%（含作业考勤上机）

期末考试 70%

# 课程网站 l.xmu.edu.cn

Ø 用于共享课件，发布通知，答疑...

Ø 加入网站（选课）步骤

u 用 l.xmu.edu.cn 的用户密码登录


u 找到课程分类

u 找到课程      方法2: <http://l.xmu.edu.cn/course/view.php?id=681>

我的主页 ▶ 信息科学与技术学院 School of Information Science & Technology ▶ 2016-17 ▶ [c2016](#)

方法3

u 用给定的密码选课

搜索课程:  

↓ 方法1

 C语言程序设计2016（郑炜）

C语言程序设计2016（郑炜）

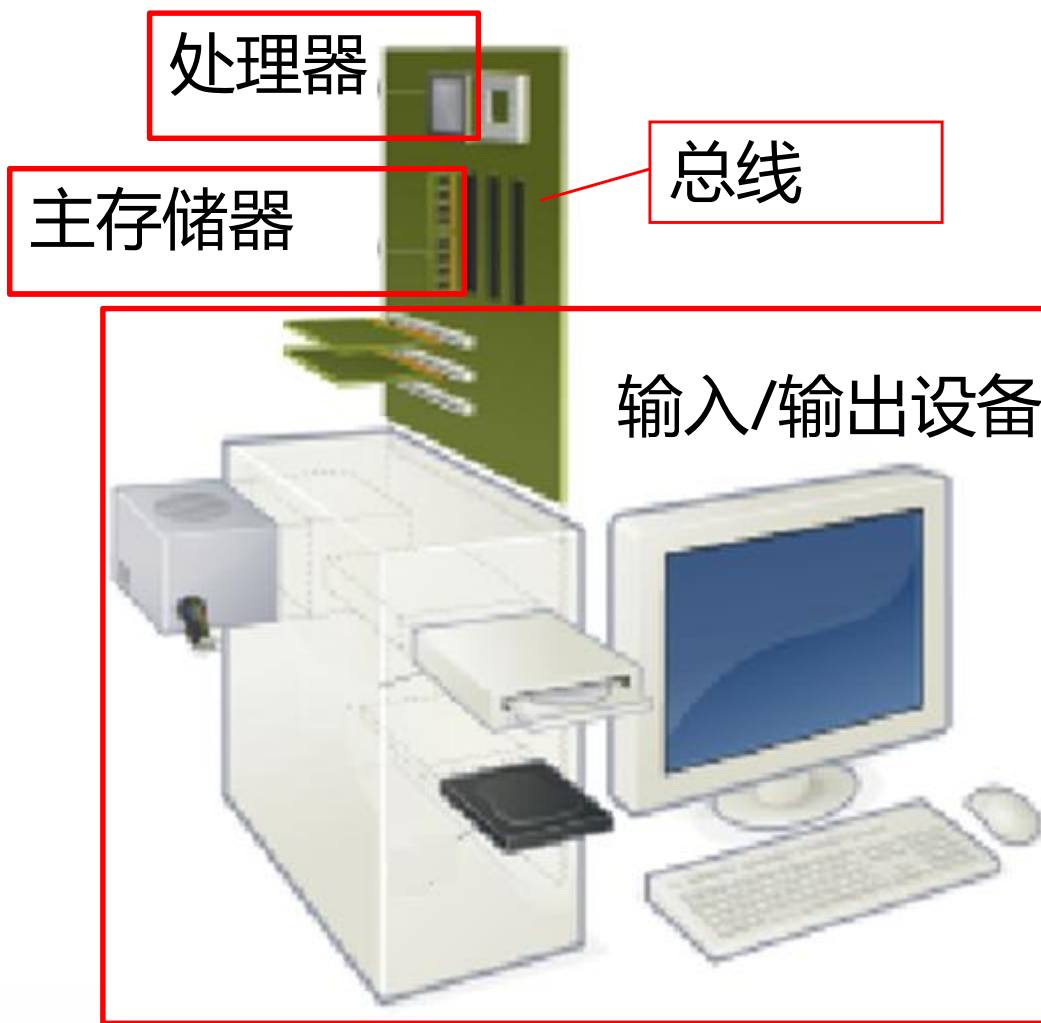
▼ 自助选课 (学生)

选课密码

☐ 显示密码

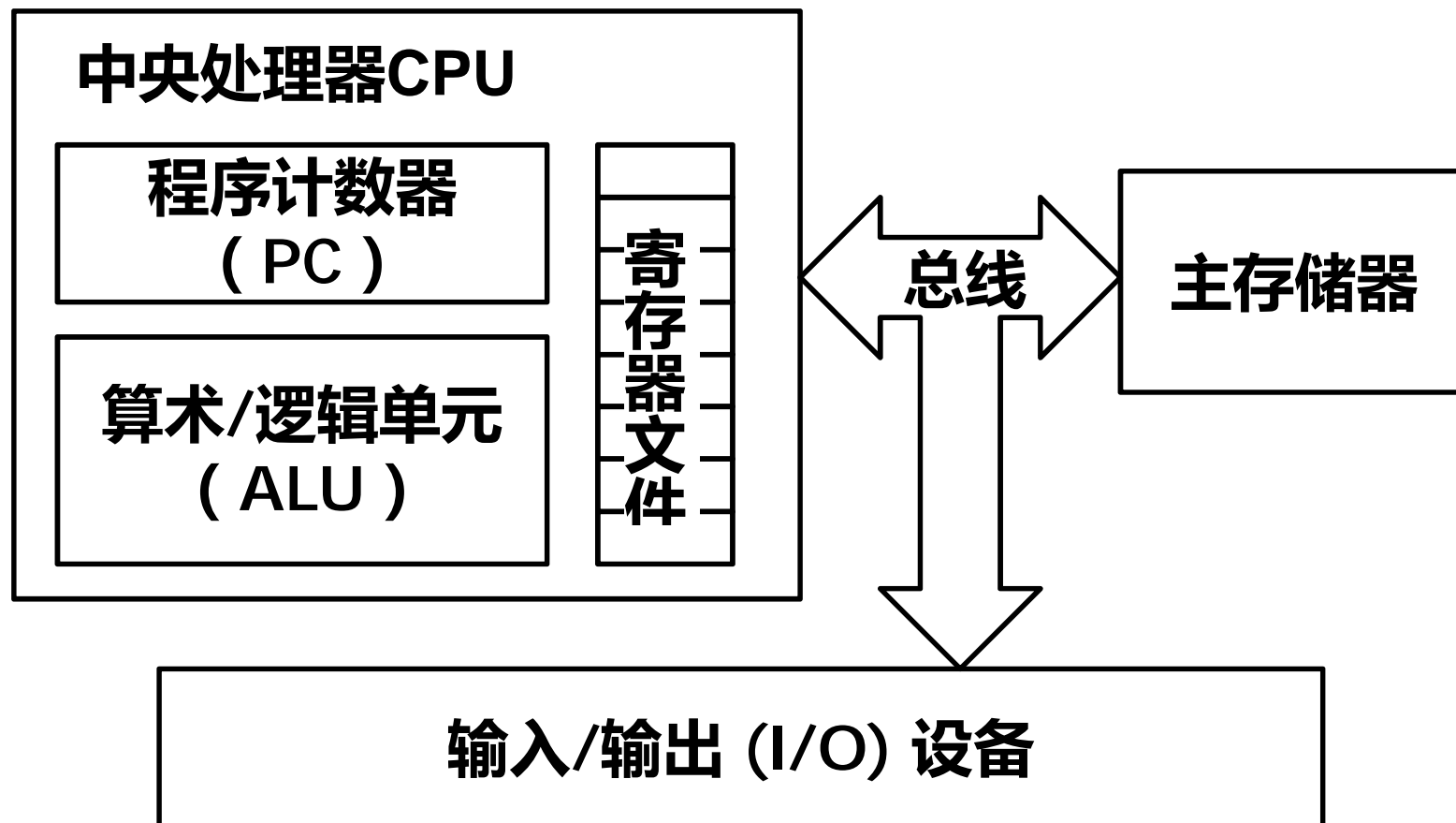
将我加入

# 外行人眼里的计算机





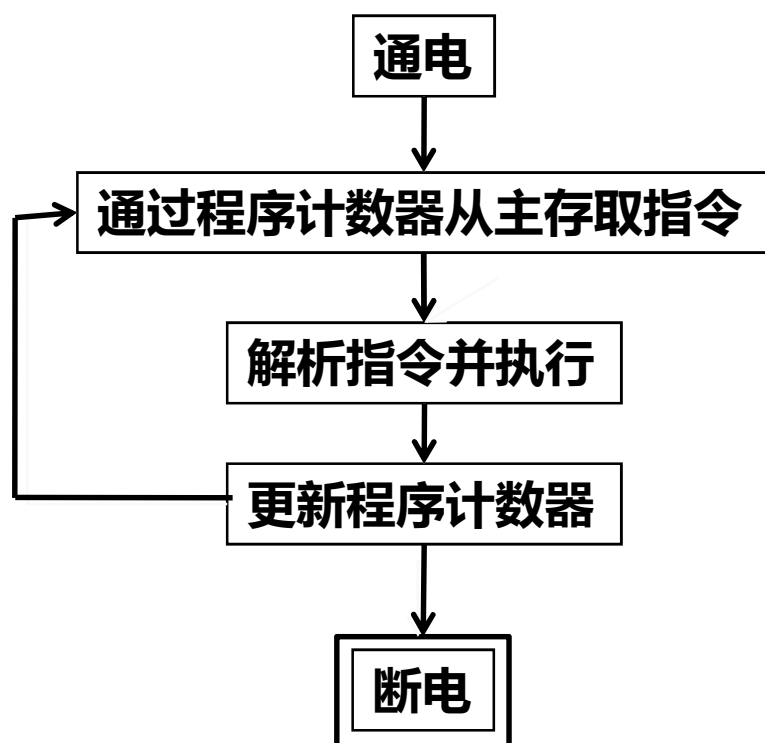
# 内行人眼里的计算机/冯诺依曼结构





# 什么是计算机程序

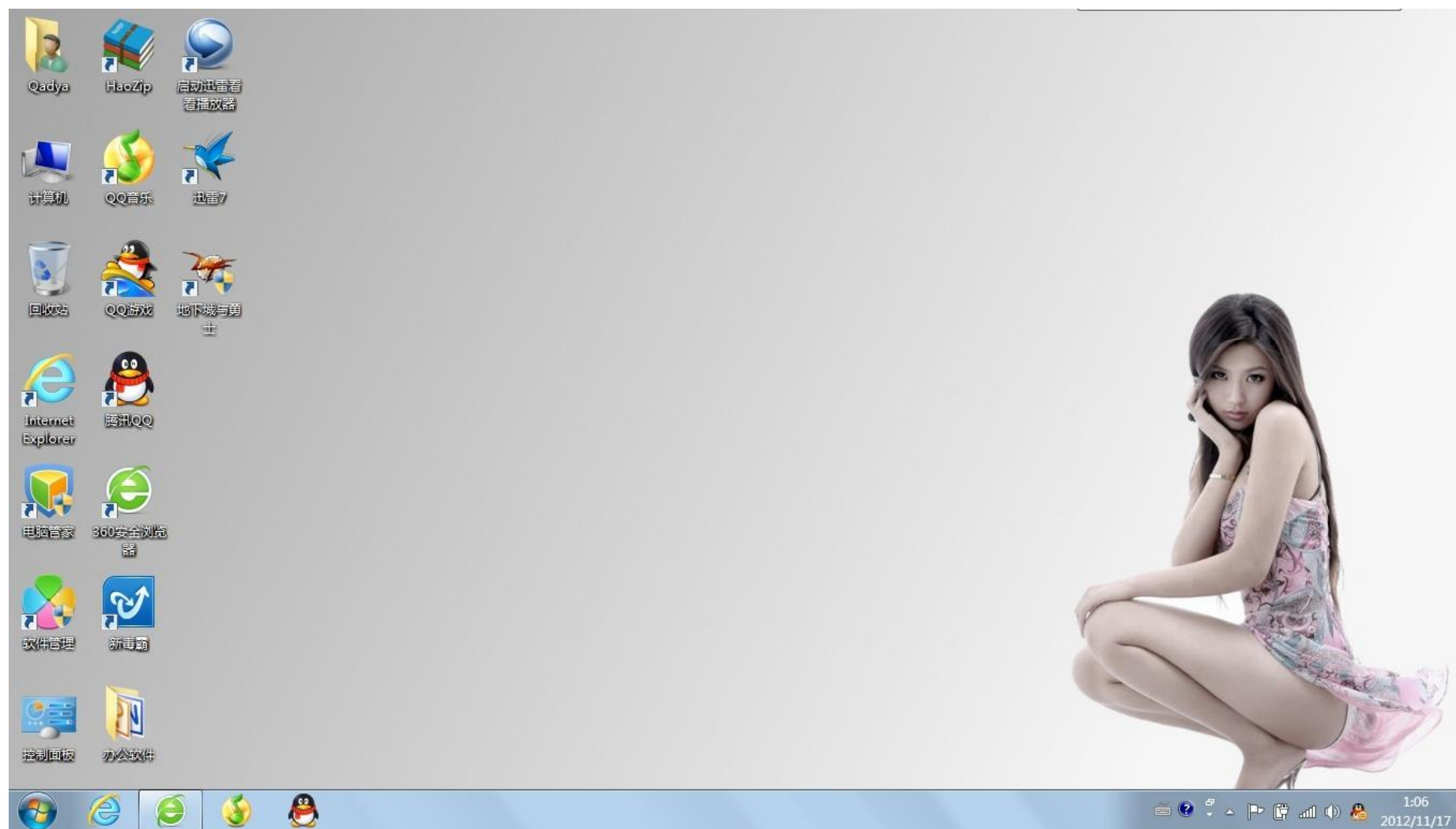
## CPU的简单逻辑



Ø **程序**：一组计算机能识别和执行的通过计算机语言表达的**指令序列**

Ø 只要让计算机开始执行这个程序，计算机就会**自动地、有条不紊地**进行工作

# 你的身边计算机程序无处不在



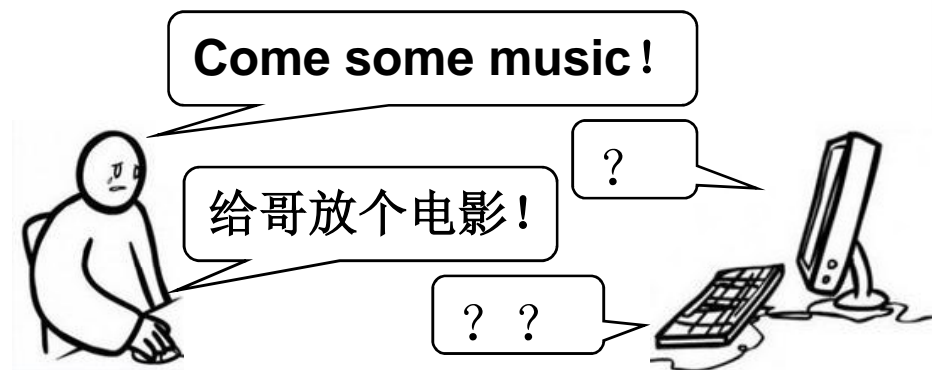
# 什么是计算机语言

Ø **计算机语言**：人和计算机交流信息的、计算机和人都能识别的语言

u 不同于自然语言的特征

l 命令式

u 不正确的计算机使用方式



# 计算机语言的分类和发展

## Ø 计算机语言发展阶段：

# 低级语言

## u 机器语言（由0和1组成的指令）。

## 汇编语言（用英文字母和数字表示指令）

## u 高级语言（接近于人的自然语言，>2500种）

## 面向过程的语言

## 非结构化的语言：Fortran...

## 结构化语言：Pascal, C...

## I 面向对象的语言：C++，Java...



# 机器语言

## Ø 特点

- u 0和1组成的序列；
- u 计算机可直接执行；
- u 针对特定型号的计算机；
- u 符合计算机的执行逻辑，不符合人的思维习惯；

## Ø 缺点

- u 难以移植，不同的CPU指令集不同
- u 难学、难记、难写、难检查、难修改

### 机器语言示例：

```
1. 01100011
2. 11111110
3. 10000110
4. 10101001
5. 1000100111011000
6. 00111100
7. 01010100
8. .....
```

### 或者写成这样：

```
1. 63fe86a9
2. 89d83c54
3. .....
```

# 汇编语言

Ø 用英文助记符号代表机器语言中的01序列

Ø 汇编指令和机器语言指令一一对应

- u 需要通过“汇编器”翻译成机器语言

- u 可读性比机器语言有所提高（但还是难懂）

- u 可移植性依旧很差

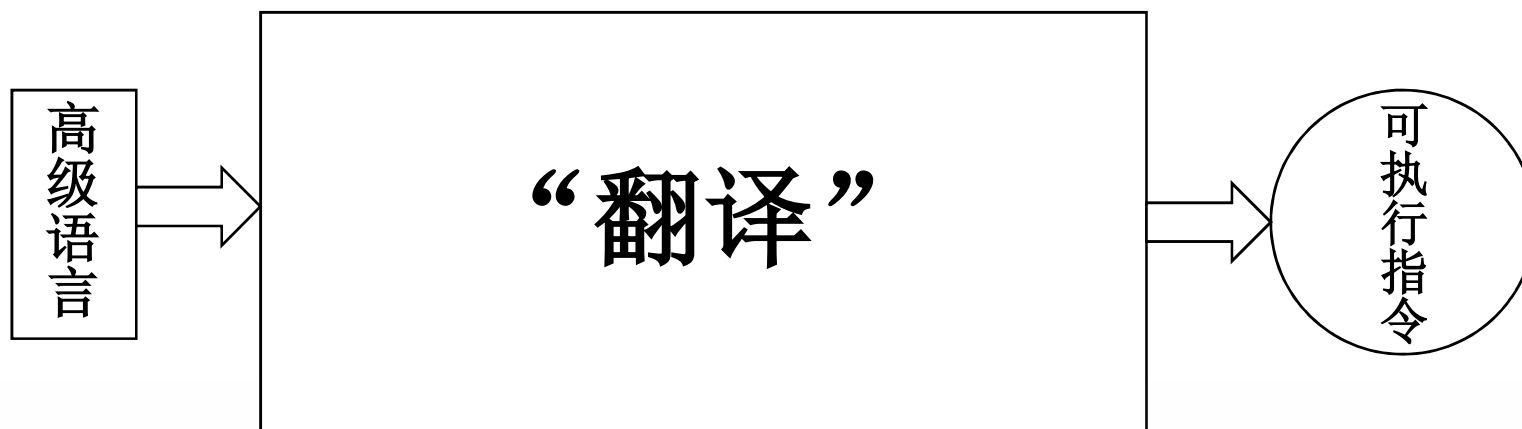
- u 编程逻辑依旧符合机器习惯而不符合人类习惯

机器语言指令	等同的汇编语言指令
01010101	push %ebp
1000100111100101	mov %esp, %ebp
11000011	ret

# 高级语言

Ø 针对机器语言和汇编语言的缺点，为了提高开发程序的效率，各种高级语言相继被发明

- └ 接近人类自然语言的表达习惯，可读性高
- └ 不依赖于计算机，编出的程序能在所有机器上使用
- └ 通过“翻译”，一条高级语言语句通常对应多条汇编（机器）语言语句





# 三种语言的对比

```

27bdf fd0 afbf 0014 0c1002a8 00000000 0c1002a8 afa2001c 8fa4001c
00401825 10820008 0064082a 10200003 00000000 10000002 00832023
00641823 1483fffa 0064082a 0c1002b2 00000000 8fbf0014 27bd0020
03e00008 00001025
  
```

```

addiu    sp,sp,-32
sw       ra,20(sp)
jal      getint
nop
jal      getint
sw       v0,28(sp)
lw       a0,28(sp)
move     v1,v0
beq      a0,v0,D
slt      at,v1,a0
A: beq    at,zero,B
nop
  
```

```

b      C
subu   a0,a0,v1
B: subu v1,v1,a0
C: bne  a0,v1,A
    slt  at,v1,a0
D: jal  putint
    nop
    lw   ra,20(sp)
    addiu sp,sp,32
    jr   ra
    move v0,zero
  
```

```

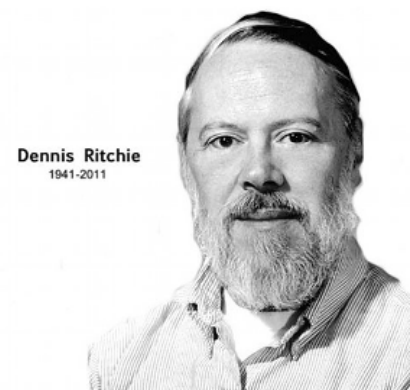
while(b!=0)
{
    temp = a%b;
    a = b;
    b = temp;
}
  
```

# 有关C语言的评价

## C诡异离奇，缺陷重重，却获得了巨大的成功



IEEE Spectrum: 2016年编程语言排行榜 C取代Java成为第一名



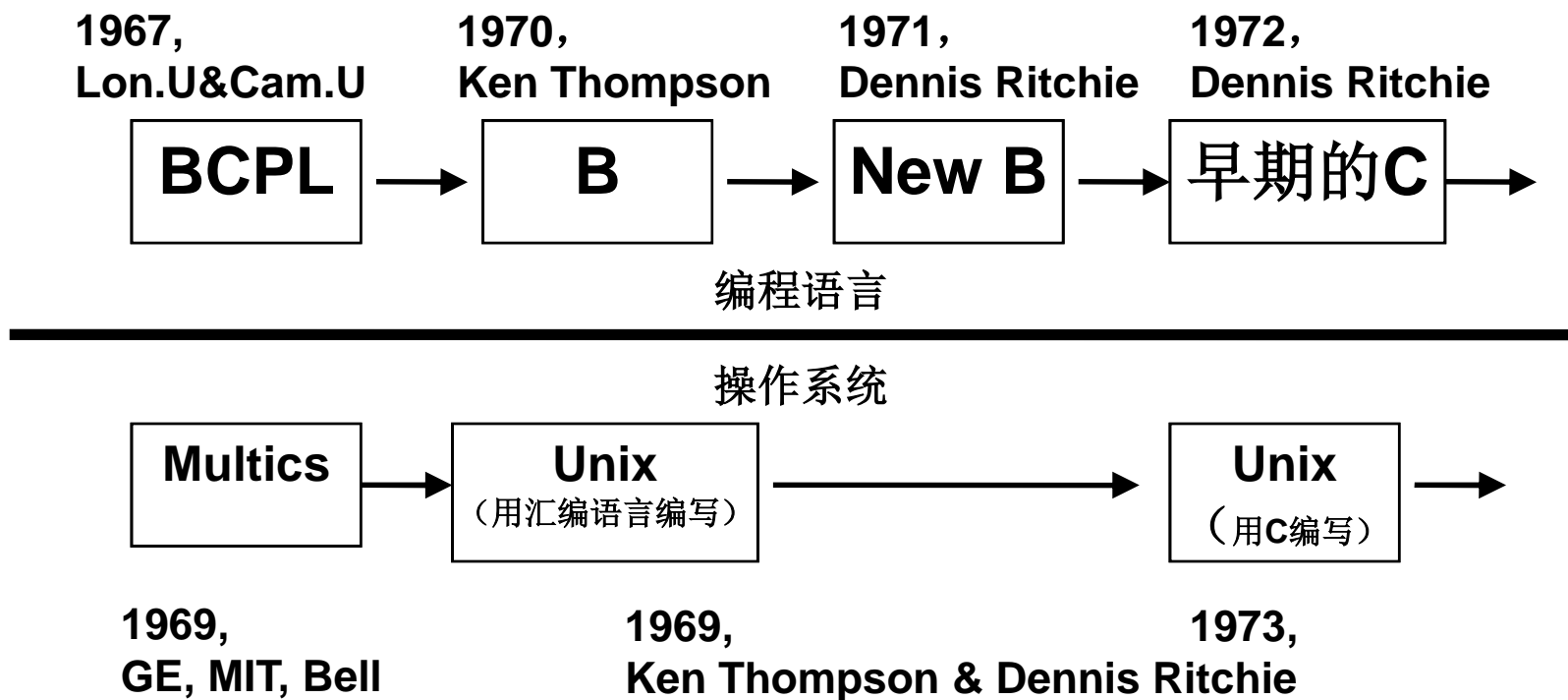
——Dennis Ritchie

1941-2011

(C语言之父，图灵奖获得者)

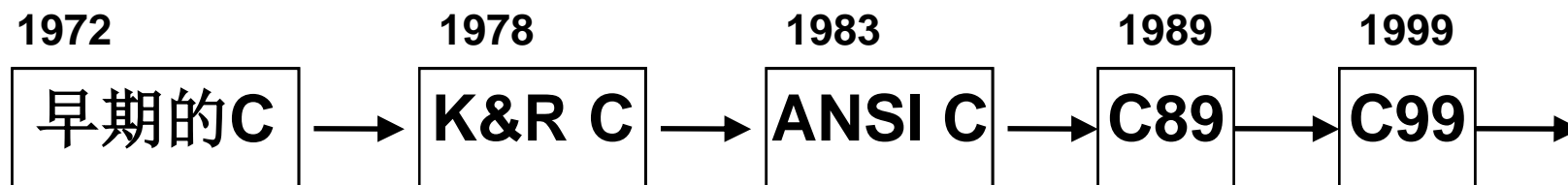
# C语言的史前阶段

Ø C语言的产生竟源于一个失败项目：Multics



Ø 天意？巧合？BCPL="Basic Combined Programming Language" or "Before C Programming Language"?

# C语言的标准化过程



Ø **K&R C**: Brian W. Kernighan and Dennis Ritchie, The C Programming Language (1978)

Ø **ANSI C**: 第一个C语言标准草案

Ø **C89** : 完整 , 目前流行的C语言编译系统多以此为标准

Ø **C99** : 我们的教材上的叙述以及**上机测评**以此为依据

# 无处不在的C语言

Ø C语言是一种用途广泛、功能强大的编程语言，既可用于编写应用软件，又能用于编写系统软件和嵌入式软件。因此C语言问世以后得到迅速推广。

Amdahl

Burroughs

Cray

⋮

Zilog

支持C语言的硬件系统从A到Z都存在！



# C语言的主要特点

- Ø 语言简洁紧凑，使用方便灵活。
- Ø 运算符丰富。
- Ø 数据类型丰富。
- Ø 具有结构化的控制语句。
- Ø 语法限制不太严格，程序设计自由度大。
- Ø 具有低级语言许多功能。
- Ø 可移植性好。
- Ø 生成目标代码质量高，程序执行效率高。

这个阶段也许你只需要知道这句话：



# 需要牢记的基本C程序代码框架

```
#include <stdio.h>
int main()
{

    return 0;

}
```

↓ 类似于一个模板 ↓

请 假 条	
尊敬的学校领导：	
我因_____需向您请____假共____天（即____月____日至____月____日），望给予准假为盼。	
此条	
请假人：_____	
_____年____月____日	

Ø 运行：执行结果为空

Ø 功能：定义了一个返回值和函数体为空的main函数（或叫主函数）

注意！C程序必须有且只能有一个 main 函数



# 简单的C程序：例1.1

【例1.1】要求在屏幕上输出以下一行信息。

This is a C program.

Ø 解题思路：

在主函数中用printf函数原样输出以上文字。

```
1. #include <stdio.h>           //这是编译预处理指令
2. int main( )                  //定义主函数
3. {                            //函数开始的标志
4.     printf("This is a C program.\n"); //输出所指定的一行信息
5.     return 0;                //函数执行完毕时返回0
6. }                             //函数结束的标志
```

# 例1.1 代码解析

需函数库的输入输出函数时引入

1. #include <stdio.h>

预处理部分

2. int main( )

基本部分

3. {

4. printf("This is a C program.\n");

5. return 0;

输出函数

6. }

# 例1.1 代码解析

1. `#include <stdio.h>`

2. `int main( )`

3. { 主函数类型

4. `printf("This is a C program.\n");`

5. `return 0;`

6. }

当main函数执行结束前  
将整数0作为函数值，C99标准

# 例1.1 代码解析

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

```
2. int main( )
```

```
3. {
```

```
4.     printf("This is a C program.\n");
```

```
5.     return 0;
```

```
6. }
```

函数头部

函数体

# 例1.1 代码解析

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

```
2. int main( )
```

```
3. {
```

输出函数

字符串

```
4. printf( "This is a C program.\n" );
```

```
5. return 0;
```

```
6. }
```

输出语句

# 例1.1 代码解析

1. `#include <stdio.h>`

2. `int main( )`

3. `{`

4. `printf("This is a C program.\n");`

5. `return 0;`

6. `}`

运行结果

```
This is a C program.  
Press any key to continue.
```

输出语句

# 例1.1 代码解析

1. `#include <stdio.h>`

2. `int main( )`

3. `{`

4. `printf("This is a C program.\n");`

5. `return 0;`

6. `}`

```
This is a C program.  
Press any key to continue.
```

换行符



# 例1.1 代码解析

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

```
2. int main( )
```

```
3. {
```

```
4.     printf("This is a C program.\n");
```

```
5.     return 0;
```

```
6. }
```

表示语句结束



新技能get√!

printf原样输出（新手级）

# 简单的C程序：例1.2

**【例1.2】 求两个给定整数变量之和。**

Ø 解题思路：

- └ 设置3个变量

- └ a和b用来存放两个整数

- └ sum用来存放和数

- └ 用赋值运算符“=”把结果传送给sum

# 例1.2 代码解析

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

```
2. int main( )
```

sum is 579

```
3. {
```

```
4.     int a,b,sum;  定义整型变量a,b,sum
```

```
5.     a = 123;      } 对变量a,b赋值
```

```
6.     b = 456;
```

```
7.     sum = a + b;  将a与b的和赋给sum
```

```
8.     printf("sum is %d\n",sum);
```

```
9.     return 0;
```

```
10. }
```



新技能getV!

整数变量的定义/赋值/运算（新手级）

# 例1.2 代码解析

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

```
2. int main( )
```

```
3. {
```

```
4.     int a,b,sum;
```

```
5.     a = 123;
```

```
6.     b = 456;
```

```
7.     sum = a + b;
```

```
8.     printf("sum is %d\n",sum);
```

```
9.     return 0;
```

```
10. }
```



新技能getv!

用printf输出变量（入门级）

用sum的值替代

希望输出的字符

sum is

579

# 简单的C程序：例1.3

**【例1.3】求两个整数中的较大者。**

Ø 解题思路：

- └ 用一个函数实现求两个整数中的较大者
- └ 在主函数中调用此函数并输出结果

# 例1.3 代码解析

```
1. #include <stdio.h>
2. int main( )
3. {
4.     int max(int x,int y);
5.     int a,b,c;
6.     scanf("%d,%d",&a,&b);
7.     c = max(a,b);
8.     printf("max=%d\n",c);
9.     return 0;
10.}
11.int max(int x,int y)
12.{
13.    int z;
14.    if (x > y) z = x;
15.    else z = y;
16.    return(z);
17.}
```

预处理

主函数

max函数

## 例1.3 代码解析

```
2. int main( )
3. {
4.     int max(int x,int y);
5.     int a,b,c;
6.     scanf("%d,%d",&a,&b);
7.     c = max(a,b);
8.     printf("max=%d\n",c);
9.     return 0;
10. }
```

将x和y中较大者  
值返回给主函数

```
11. int max(int x,int y)
12. {
13.     int z;
14.     if (x > y) z = x;
15.     else z = y;
16.     return(z);
17. }
```



# 例1.3 代码解析

```
2. int main( )
3. {
4.     int max(int x,int y);
5.     int a,b,c;
6.     scanf("%d,%d",&a,&b);
7.     c = max(a,b);
8.     printf("max=%d\n",c);
9.     return 0;
10. }
```

```
11. int max(int x,int y)
12. {
13.     int z;
14.     if (x > y) z = x;
15.     else z = y;
16.     return(z);
17. }
```

# C语言程序结构的特点

## 1. 一个程序由一个或多个源程序文件组成

- u 小程序往往只包括一个源程序文件

- l 如例1.1, 1.2, 1.3, 以及你们这学期所学所写的代码.....

- u 一个源程序文件中可以包括三个部分：

- l 预处理指令 `#include <stdio.h>`等

- l 全局声明 在函数之外进行的数据声明

- l 函数定义 每个函数用来实现一定的功能

# C语言程序结构的特点

## 2. 函数是C程序的主要组成部分

u 一个C程序是由一个或多个函数组成的

l 例1.1，例1.2只有一个函数

l 例1.3有两个函数

u 必须包含一个main函数（只能有一个）

u 每个函数都用来实现一个或几个特定功能

u 被调用的函数可以是库函数，也可以是自己编制设计的函数

```
int main( )
{
    int max(int x,int y);
    int a,b,c;
    scanf("%d,%d",&a,&b);
    c = max(a,b);
    printf("max=%d\n",c);
    return 0;
}
```

```
int max(int x, int y)
{
    int z;
    if (x > y) z = x;
    else z = y;
    return(z);
}
```

# C语言程序结构的特点

## 3. 一个函数包括两个部分：（详见第七章）

u 函数首部

函数的第1行

int

max

( int x, int y )

```
int max(int x, int y)
```

```
{  
    int z;  
    if (x > y) z = x;  
    else z = y;  
    return(z);  
}
```

函数类型

函数名

参数类型

参数名

若函数无参，在括弧中写void或空括弧

`int main(void)` 或 `int main()`

# C语言程序结构的特点

## 3. 一个函数包括两个部分：（详见第七章）

U 函数体：就是函数首部下面用{ }括起来的部分

### I 声明部分

μ 定义在本函数中所用到的变量

μ 对本函数所调用函数进行声明

### I 执行部分

μ 由若干个语句组成，指定在函数中所进行的操作

μ 可以是空函数，如  
void dump() { }

```
int main( )  
{  
声明  int max(int x,int y);  
      int a,b,c;  
      scanf("%d,%d",&a,&b);  
执行  c = max(a,b);  
      printf("max=%d\n",c);  
      return 0;  
}
```

```
int max(int x, int y)  
{  
声明  int z;  
      if (x > y) z = x;  
执行  else z = y;  
      return(z);  
}
```

# C语言程序结构的特点

4. 程序总是从main函数开始执行

5. C程序对计算机的操作由C语句完成

└ C程序书写格式是比较自由的

└ 一行内可以写几个语句

└ 一个语句可以分写在多行上

└ 为清晰起见，习惯上每行只写一个语句

```
int main( )
{
    int max(int x,int y);
    int a,b,c;

    scanf("%d,%d",&a,&b);
    c = max(a,b);
    printf("max=%d\n",c);
    return 0;
}

int max(int x, int y)
{
    int z;

    if (x > y) z = x;
    else z = y;
    return(z);
}
```

# C语言程序结构的特点

6. 每个数据声明和语句最后必须有分号

7. 可以通过预处理调用标准库中的函数

u 要使用printf或scanf, 需使用  
#include <stdio.h>

8. 程序应当包含适当注释, 增加可读性

```
#include <stdio.h>
int main( )
{
    int max(int x,int y);
    int a,b,c;
    scanf("%d,%d",&a,&b);
    c = max(a,b);
    printf("max=%d\n",c);
    return 0;
}
int max(int x, int y)
{
    int z;
    if (x > y) z = x;
    else z = y;
    return(z);
}
```

# 关于注释

Ø 注释部分对代码没有影响，对运行不起作用

Ø C语言允许用两种注释方式：

**//：单行注释**

┆ 可单独占一行

┆ 可出现在一行中其他内容的右侧

**/\*.....\*/：块式注释**

┆ 可包含多行，如：

```
1. /* This is
2.    a C Program
3.    */
```

```
... //这是编译预处理指令
... //定义主函数
... //函数开始的标志
... //输出所指定的一行信息
... //函数执行完毕时返回0
... //函数结束的标志
```



# 程序设计的任务

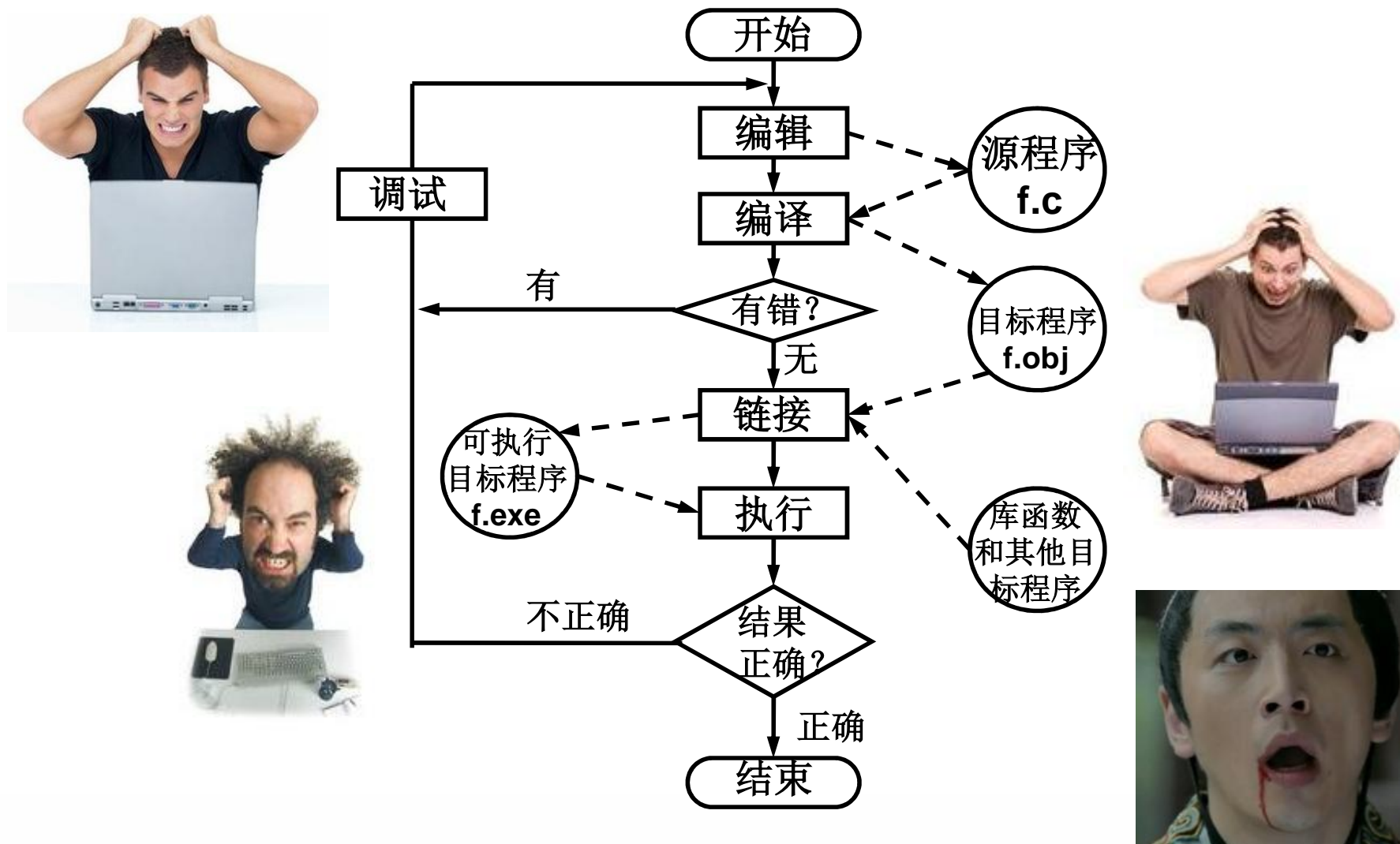
1. 问题分析
2. 设计出解题的方法和具体步骤
3. 编写程序
4. 对源程序进行编辑、编译和连接
5. 运行程序，分析结果
6. 编写程序文档

# 开发C程序的完美流程

1. 上机输入和编辑源程序（.c文件）
2. 对源程序进行编译（.obj文件）
3. 进行连接处理（.exe文件）
4. 运行可执行程序，得到正确结果



# 开发C程序的真实流程



# 编程工具的选择

Ø 集成环境（IDE）：把程序的编辑、编译、链接、运行和调试等操作集中在一个界面

u如：CodeBlocks，Visual Studio，.....

Ø 命令行开发

u如：GCC

I Windows：基于MinGW

I Linux：自带

Ø 建议：先掌握一种，然后举一反三

# 本课程推荐

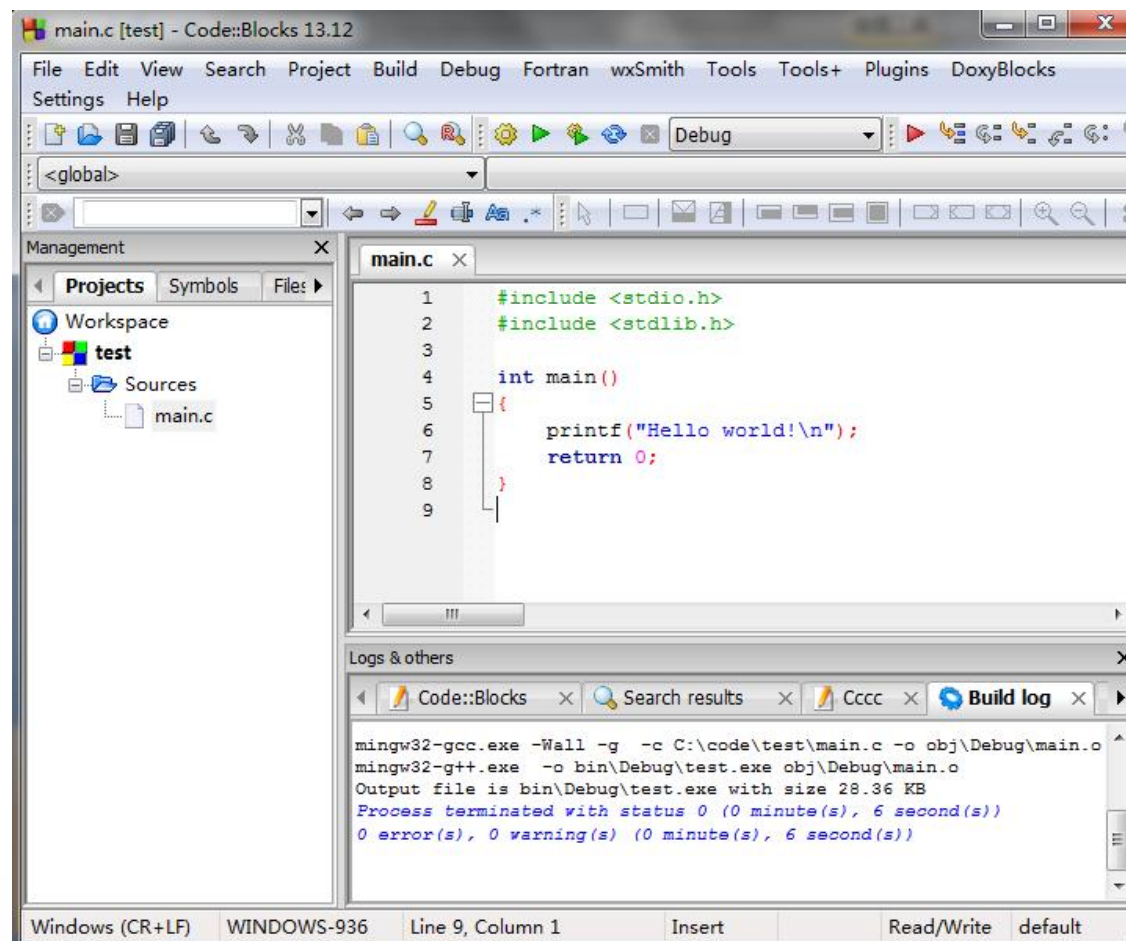
## Code::Blocks 13.12

【windows用户】  
下载安装课程网站上的

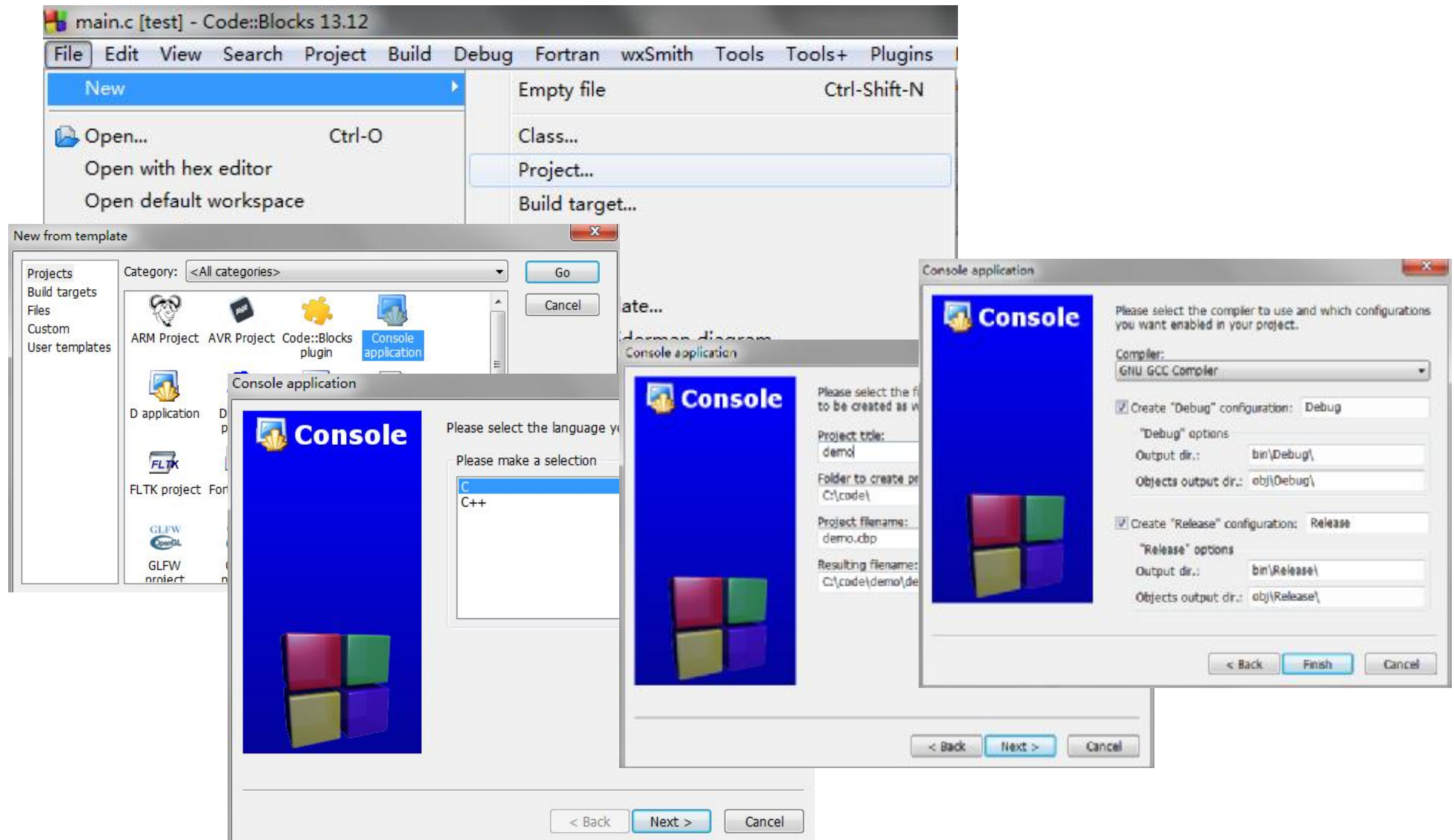
 codeblocks-13.12mingw-setup

【mac用户】  
下载安装课程网站上的

 CodeBlocks-13.12-mac



# Code::Blocks创建新项目步骤



# 进阶：GCC的安装与配置

**【攻略0】百度搜索相关下载资源及安装说明，如**  
<http://jingyan.baidu.com/article/ff42efa91946b8c19e2202f4.html>

**【攻略1】课程网站上“windows命令行常用命令参考”**

**【攻略2】本课件随后的gcc命令行开发步骤说明**



# GCC命令行开发步骤

- Ø 用任何文字编辑器（如记事本），编写下列代码并存放在c:\code\test.c

```
#include <stdio.h>
int main()
{
    printf("Hello, World!");
    return 0;
}
```

- Ø 一步到位的编译：`C:\code>gcc test.c -o test`

- Ø 运行程序：`C:\code>test`  
`Hello, world`



# GCC命令行开发步骤

Ø 实际上，一步到位的编译包含四个步骤

1. 预处理 `C:\code>gcc -E test.c -o test.i`
2. 编译为汇编码 `C:\code>gcc -S test.i -o test.s`
3. 编译为目标文件 `C:\code>gcc -c test.s -o test.o`
4. 链接 `C:\code>gcc test.o -o test`

Ø 依次执行这四条命令，与直接执行“gcc test.c -o test”同样可得到可执行文件 test.exe

# 编程如何快速入门（功夫在课外）

## Ø 阅读 + 练习

### └ 阅读：书籍 + 代码示例 + 网络资源

- └ 书：纸版书（图书馆）+ 电子书（百度 + 各种网盘）
- └ 经典的示例代码：先在入门书找基本例子，再到网络上找有趣的例子

### └ 练习：编程不是看会的，不是听会的，是练会的

- └ 看书的同时身边应该有台可以用于编程的计算机
- └ 在实践中验证所学的知识，积累经验和技巧

# 作业 2016/09/18

1. 参照本章例题编写一个程序，在屏幕上输出你的姓名，学号和籍贯。

## 注意事项：

- (1) 作业写在纸上；
- (2) 作业纸抬头写上学号和姓名；
- (3) 周二（9月20日）课间休息时间交给助教

**上机练习（不用交）：安装CodeBlocks，并编写第一章（4，5，6，7）**

# 助教信息

**助教：侯士伟**

E-mail: 18106907663@163.com  
手机：18106907663

**助教：刘佳雯**

E-mail: 374921173@qq.com  
手机：15632304995

助教职责包括：

收/发/批改作业，上机辅导，答疑等等