

程序设计基础

Fundamental of Programming



清华大学软件学院

刘玉身

liuyushen@tsinghua.edu.cn

09:50 - 12:15 @明理楼321

教学安排

- **教学对象**

- 计算机软件专业**新生**

- **教学目标**

- 掌握一种**基本编程语言**（C/C++语言）
- 掌握程序设计的**基本思路和方法**
- 提高分析问题、解决问题的**能力**
- 为将来学习其他编程语言和专业课打下**基础**
- **兴趣性、实践性、基础性**

学中做，做中学，做中闯，做中创

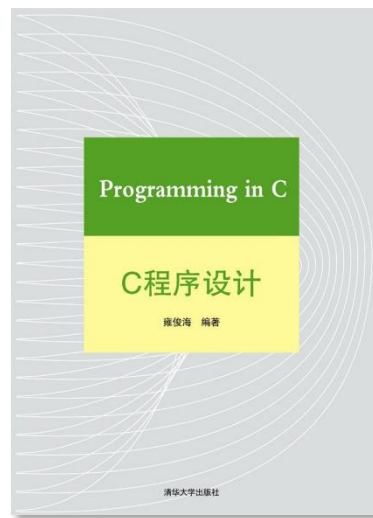
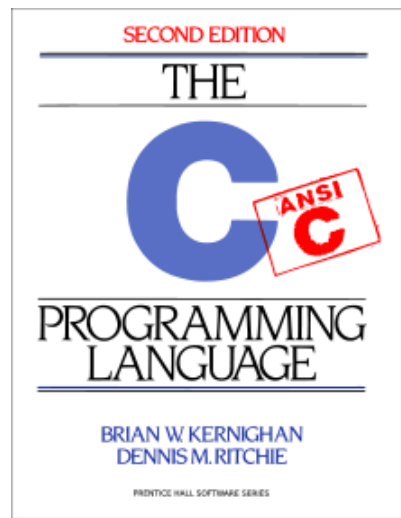
- **教学方法:**

- **课堂讲解：基本概念、C/C++语言的语法、语句，编程解题的基本思路、方法。每堂课要带上纸、笔，进行课题练习。鼓励创新、鼓励上台讲解自己的方法**
- **上机实践：实践性极强的课程，将有大量的编程练习。每周有一次固定上机时间，其余自由上机**
- **上机时间：周四下午13:00-16:00**
- **上机地点：东配楼11区402、406**

教材

• 教材及参考书

- 《程序设计基础（第4版）》 吴文虎、徐明星、邬晓钧 编著，清华大学出版社，2017
- 谌卫军编著，“计算机语言与程序设计”，清华大学出版社
- K & R (Kernighan and Ritchie). **The C Programming Language** (2nd Edition)
- 雍俊海. C程序设计. 北京: 清华大学出版社. 2017.



参考资料

- **参考资料:**

- C Primer Plus (5th Edition)
- [MIT] Practical Programming in C ([open course](#))

- **参考课件:**

- MIT, Stanford, UCB, CMU等大学相关PPT、Course Note

考核方式

- **考核方式：**
 - **上机练习 + 大作业 + 期末考试**
(20%) (20%) (60%)
- **课程站点：网络学堂（新版）**
 - **<http://learn.tsinghua.edu.cn>**
 - **补全学生个人信息：邮件、电话**

25	2023012118	王欢	女	中国	软件学院	软件3 1	本科生	18201652628	3106370146@qq.com
26	2023012119	韩旭杰	男	中国	软件学院	软件3 1	本科生		
27	2023012120	吴昱东	男	中国	软件学院	软件3 1	本科生		

教学计划（48学时）

章节	名称	内容	学时
1	简单程序设计	程序设计概述、编程准备	3
2	代数思维与计算机解题	数据类型、常量、变量、运算符和表达式等	3
3	逻辑思维与计算机解题	关系运算和关系表达式、逻辑运算和逻辑表达式、循环结构	3
4	函数思维与模块化设计	函数定义、函数使用、变量作用范围、函数调用实现过程	3
5	数据的组织与处理——数组	一维数组、二维数组、字符数组、程序举例	6
6	数据的组织与处理——结构	指针的基本概念、指针变量、指针与数组、指针与字符串 结构体、结构体数组与指针、链表等	3
7	数据的组织与处理——文件	文件的基本概念、文件的访问、程序举例	3
8	递归思想与相应算法	递归及其实现、算法举例	9
9	动态规划	动态规划、算法举例	3
10	多步决策	多步决策、算法举例	3
11	宽度优先搜索	宽度优先搜索、算法举例	3
12	深度优先搜索	深度优先搜索、算法举例	3
13	习题讲解、大作业汇报	习题讲解、复习、大作业展示	3

教学小组

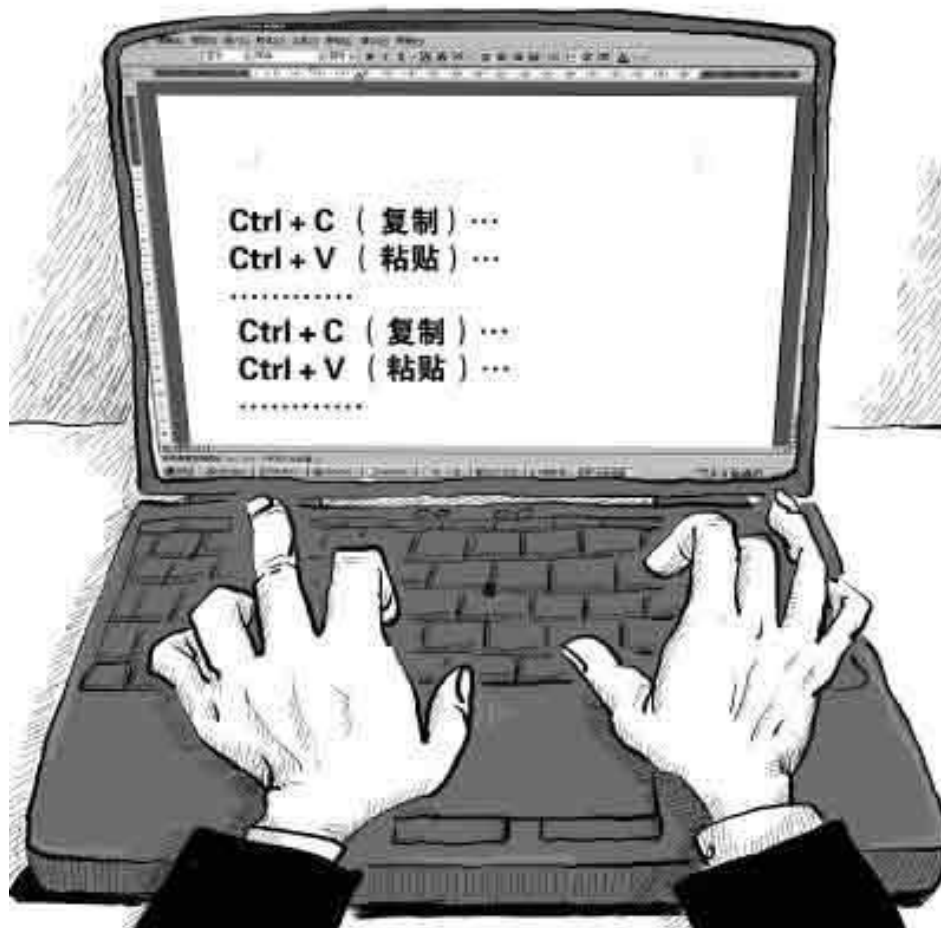
- **任课教师：刘玉身**

- **东主楼10-407, 电话: 62795455, 15910831178**
Email: liuyushen@tsinghua.edu.cn
<https://yushen-liu.github.io/>

- **助教：**

- ① **陈 超: 东主楼10-406, 电话: 13593260961,**
Email: thss15_chenc@163.com

- ② **张文源: 东主楼10-406, 电话: 18801005068,**
Email: zhangwen21@mails.tsinghua.edu.cn



若发现两份相同的程序，均记0分

小调查

- **编程基础**

- **参加过程序设计类比赛（省、市、全国）**
NOIP/NOI/APIO/IOI等
- **使用过C/C++语言或其他编程语言，编写过一些小程序**
- **熟悉电脑操作，熟悉Windows操作系统，熟悉各种应用软件的使用**
- **没有接触过太多的计算机**

—推选一名课代表！

编程基础小调查

- ☐ A 参加过程序设计类比赛（省、市、全国）
NOIP/NOI/APIO/IOI等
- ☐ B 使用过C/C++语言或其他编程语言，编写过一些小程序
- ☐ C 熟悉电脑操作，熟悉Windows操作系统，熟悉各种应用软件的使用
- ☐ D 没有接触过太多的计算机

提交

Any question?

Brief Introduction to C

- **Introduction to C**
- **A Simple C Program**
- **Build VC Programs**

Brief Introduction to C

- **Introduction to C**
- A Simple C Program
- Build VC Programs

What is C?

- C designed by **Dennis Ritchie** at AT&T Bell Labs in 1972
- Influenced by
 - ALGOL 60 (1960),
 - CPL (Cambridge, 1963),
 - BCPL (Martin Richard, 1967),
 - B (Ken Thompson, 1970)
- Traditionally used for system programming, and widely used today
 - UNIX, Linux
 - Windows
 - Oracle DBMS
 - Development in Embedded Systems
 - MATLAB, **Python**



Any other software written by C?

Dennis Ritchie（丹尼斯·里奇）

- **Dennis MacAlistair Ritchie (dmr) 1941-2011**

- C programming language
- Co-author of the book (**K&R C**)
- Unix (with **Ken Thompson(肯·汤普逊)**)
- **Turing Award (图灵奖)** from the ACM in 1983
- National Medal of Technology in 1999
-



- **Why create C ?—— 'looked like a good thing to do'**




http://en.wikipedia.org/wiki/Dennis_Ritchie

Dennis Ritchie（丹尼斯·里奇）

- **Dennis Ritchie**

- Graduated from Harvard University
- Degrees in physics and applied mathematics
- PhD degree under Patrick C. **Fischer** (computational complexity & database)
- Passed away on Oct. 12, 2011
- A week after death of Steve Jobs (Oct. 5, 2011)

- 后人的评价

- “helped shape the digital era.”
- “... if you had a microscope and could look in a computer, you'd see his work everywhere inside.”
- 

http://en.wikipedia.org/wiki/Dennis_Ritchie



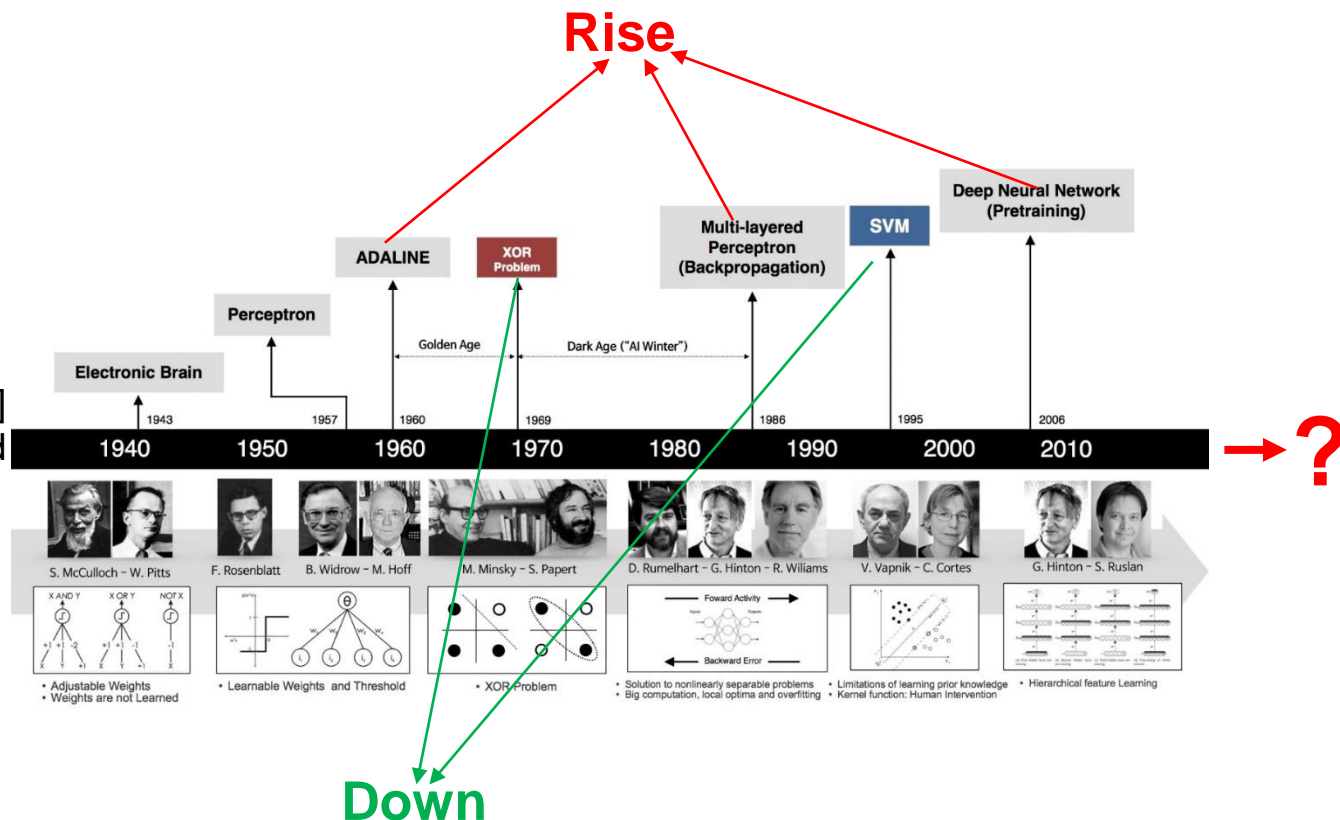
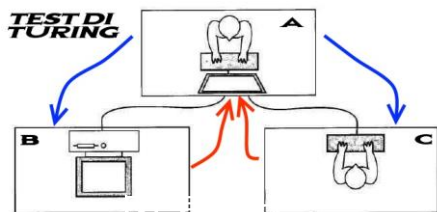
2019年3月27日 —— ACM宣布：深度学习的三位创造者
Geoffrey Hinton, **Yann LeCun** 和 **Yoshua Bengio**获得了2019年的图灵奖

图灵奖是计算机科学领域的最高奖

人工智能的发展史



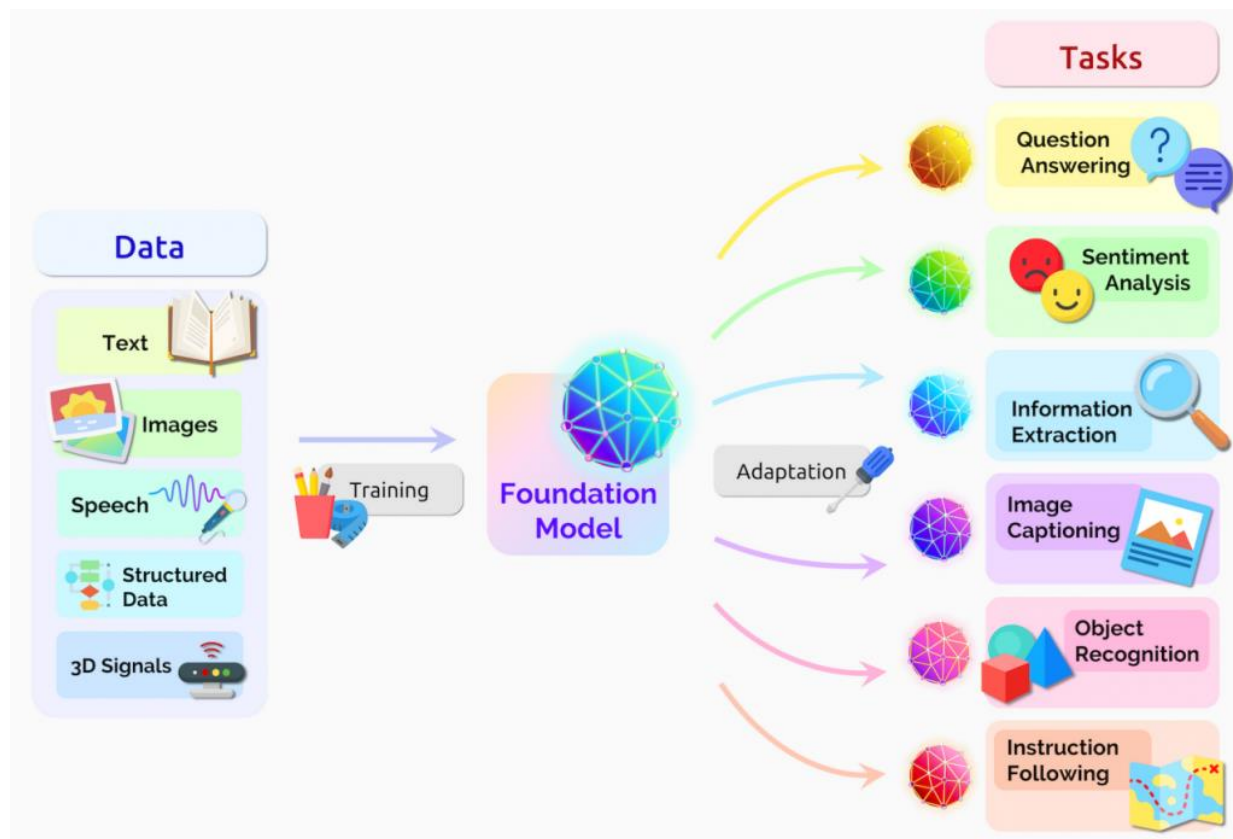
---Alan Turing [1950]
《Computing Machinery and Intelligence》



新一代人工智能的四要素：算法、算力、大数据、知识

大模型

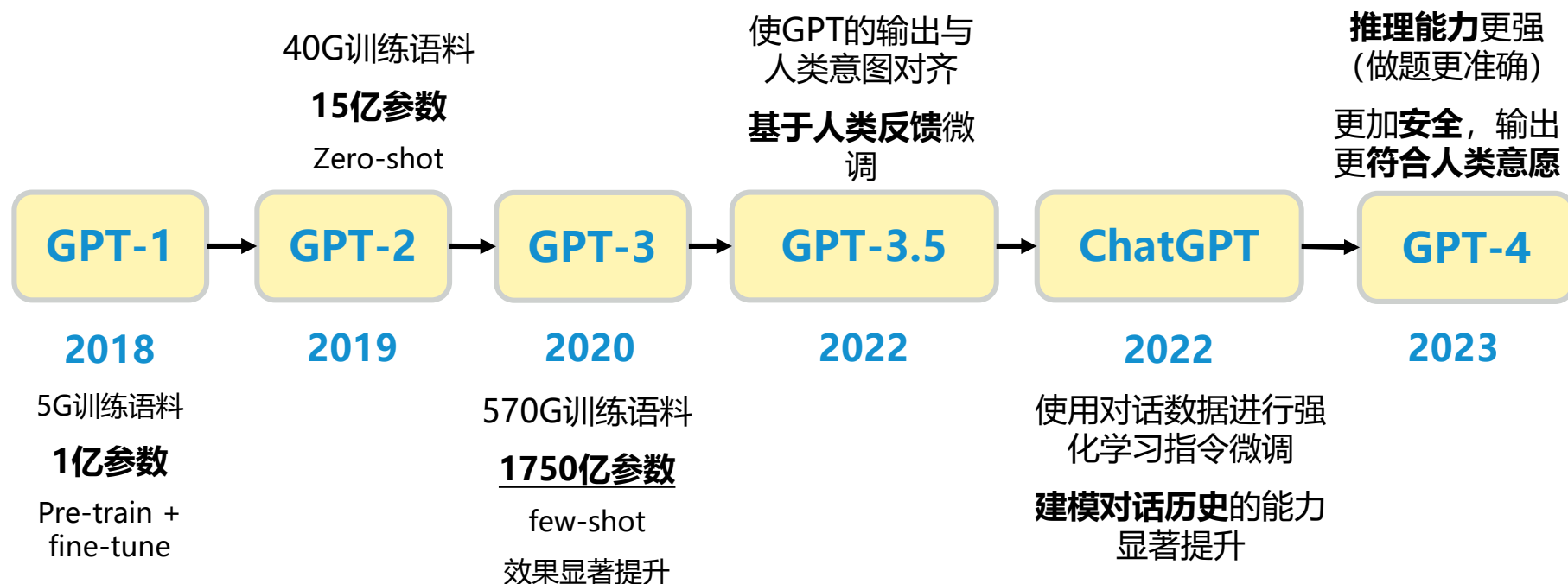
大模型 (Foundation Model)，又称基础模型，是指在大规模数据（一般指无标数据，如CLIP有4亿图像-文本对，GPT-3有570GB训练语料）上预训练后可以被应用到大量下游任务的模型，如近年来影响深远的GPT、BERT和CLIP等



大模型—自然语言处理NLP

GPT后续系列

受启发于BERT的成功，GPT后续系列也进一步将增加模型参数量和数据集规模，在GPT-3出现后，效果大幅提升，再到之后的ChatGPT和GPT-4受到广泛关注



大模型—自然语言处理NLP

大语言模型LLM

大预言模型LLM (Large Language Model) 是参数规模庞大的模型统称，旨在理解和生成人类语言，代表作包括前面提到的BERT和GPT系列，也包括近年来LLaMA和ERNIE文心大模型等

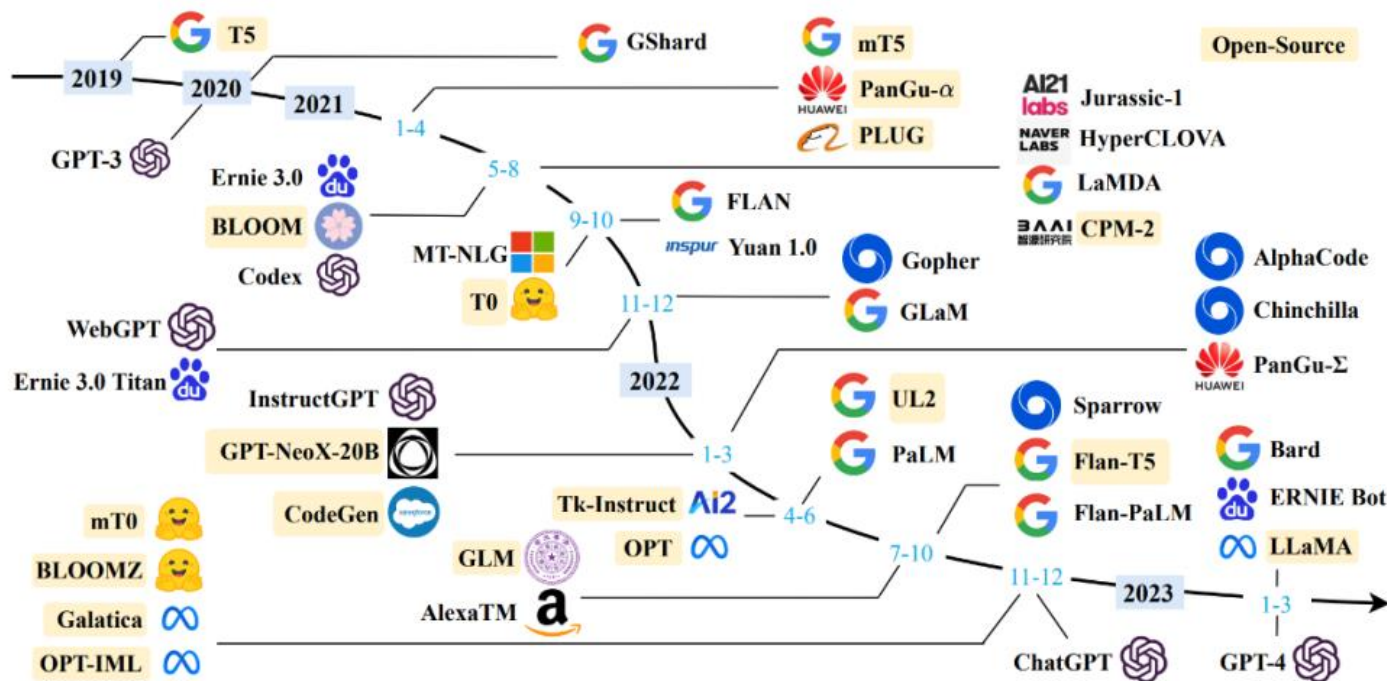


Fig. 1. A timeline of existing large language models (having a size larger than 10B) in recent years. We mark the open-source LLMs in yellow color.

大模型—计算机视觉CV

CLIP

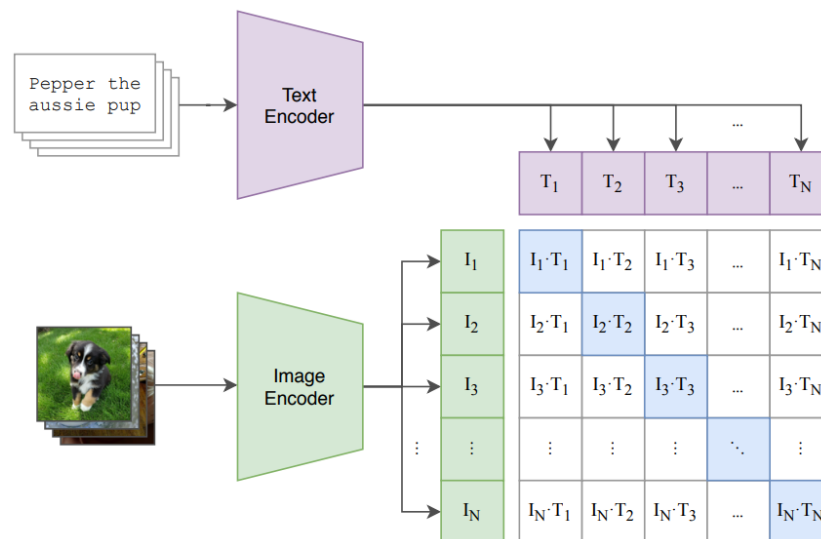
CLIP (Contrastive Language-Image Pretraining) 是一种基于**对比学习**和**图像-文本对**的**预训练**方法/模型，其目标是在自然语言的监督下学习到**可迁移**的视觉特征

□ CLIP具体在做什么？

- 找到**图像和文本的匹配关系**，如给定一张图片，从上万个文本中找到和图片对应的那一个

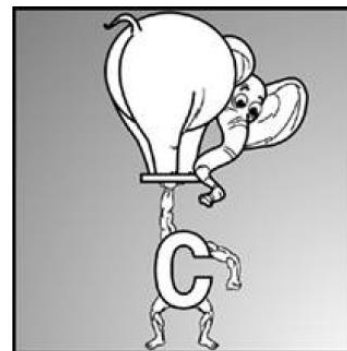
□ 为什么这样能学习到可迁移的视觉特征？

- 当数据集够大（4亿张图片），模型需要学习到**各种各样（不限类别）**的视觉概念，才可能找到其对应的文本



Features of C (特点)

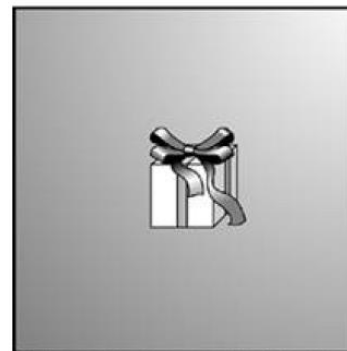
- C features:
 - Efficiency (高效)
 - Portability (可移植性好)
 - Power and Flexibility (灵活)
 - Programmer Oriented (面向程序员)
 - 数据类型丰富：字符、多种长度的整数和浮点数、指针、数组、结构和联合(unions)
 - 变化的控制结构



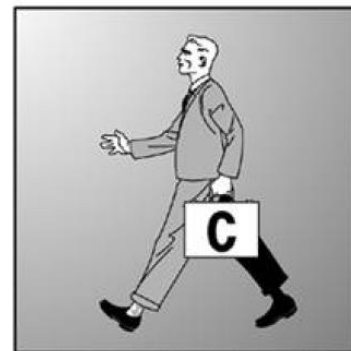
Powerful control structures



Fast



Compact code — small programs



Portable to other computers

Shortcomings of C (缺点)

- Shortcomings?
 - Exceptions (缺乏异常处理)
 - Range-checking (缺乏取值范围检查)
 - Garbage collection (缺乏垃圾收集机制)
 - Object-oriented programming (缺乏面向对象编程)
 - C语言不提供直接处理诸如字符串、集合、列表或数组等复合对象的操作;
 - C语言不直接提供多线程、并行操作、同步和协同的操作;

但上述很多局限可以通过调用一些扩展库来实现!

C vs. Related Languages (比较)

- More recent derivatives: C++, C#
- Influenced: Java, Perl, Python
- “Low-level” language → faster code (usually)
- Inherently unsafe:
 - No range checking
 - Limited type safety at compile time
 - No type checking at runtime

C语言简单、易学，运行效率高，可以了解计算机内存管理

学习一门年龄比自己还大的语言很有必要！ 😊



Compare C with C++, Java and Python ?

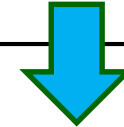
Brief Introduction to C

- Introduction to C
- **A Simple C Program**
- Build VC Programs

A Simple C Program

- 问题：打印 “hello, world”
- Create a C file: hello.cpp
- Compile and Link
- Run: hello.exe

```
/* A Simple C Program */  
#include <stdio.h>  
  
int main()  
{  
    printf("hello, world\n");  
    return 0;  
}
```



```
C:\WINDOWS\system32\cmd.exe  
hello, world  
请按任意键继续. . .
```

C Program Structure

- **Comments (注释)**
 - `/* */` 或 `//`
 - Used to describe program
- **C程序由函数和变量组合：**
 - 程序就是死规定，没有灵活性

```
/* A Simple C Program */  
#include <stdio.h>  
  
int main()  
{  
    printf("hello, world\n");  
    return 0;  
}
```

C Program Structure

- **#include <stdio.h>**
 - **#**
 - Preprocessor directive
 - **#include**
 - Tells computer to load contents of a certain file
 - **<stdio.h>**
 - The standard input/output library

```
/* A Simple C Program */
#include <stdio.h>

int main()
{
    printf("hello, world\n");
    return 0;
}
```

../VC/include/stdio.h

```
/* comments */
#ifndef _STDIO_H
#define _STDIO_H

... definitions and prototypes

#endif
```

More about header files

- **stdio.h** – part of the C Standard Library
 - Other important header files: `math.h`, `stdlib.h`, `string.h`, `time.h`
 - Included files must be on include path
- `#include "stdio.h"` 从你的当前目录 ./ 开始查找 `stdio.h` 文件, 然后再从系统目录查找
- `#include <stdio.h>` 直接从系统目录查找



`#include <stdio.h>` vs. `#include "stdio.h"`

C Program Structure

- **int main()**
 - C programs contain one or more functions, exactly one of which must be **main**
 - **()** used to indicate a function
 - **int** means that **main** "returns" an int value
 - Braces (**{** and **}**) indicate a block
 - The bodies of all functions must be contained in braces

```
/* A Simple C Program */  
#include <stdio.h>  
  
int main()  
{  
    printf("hello, world\n");  
    return 0;  
}
```


main() 函数

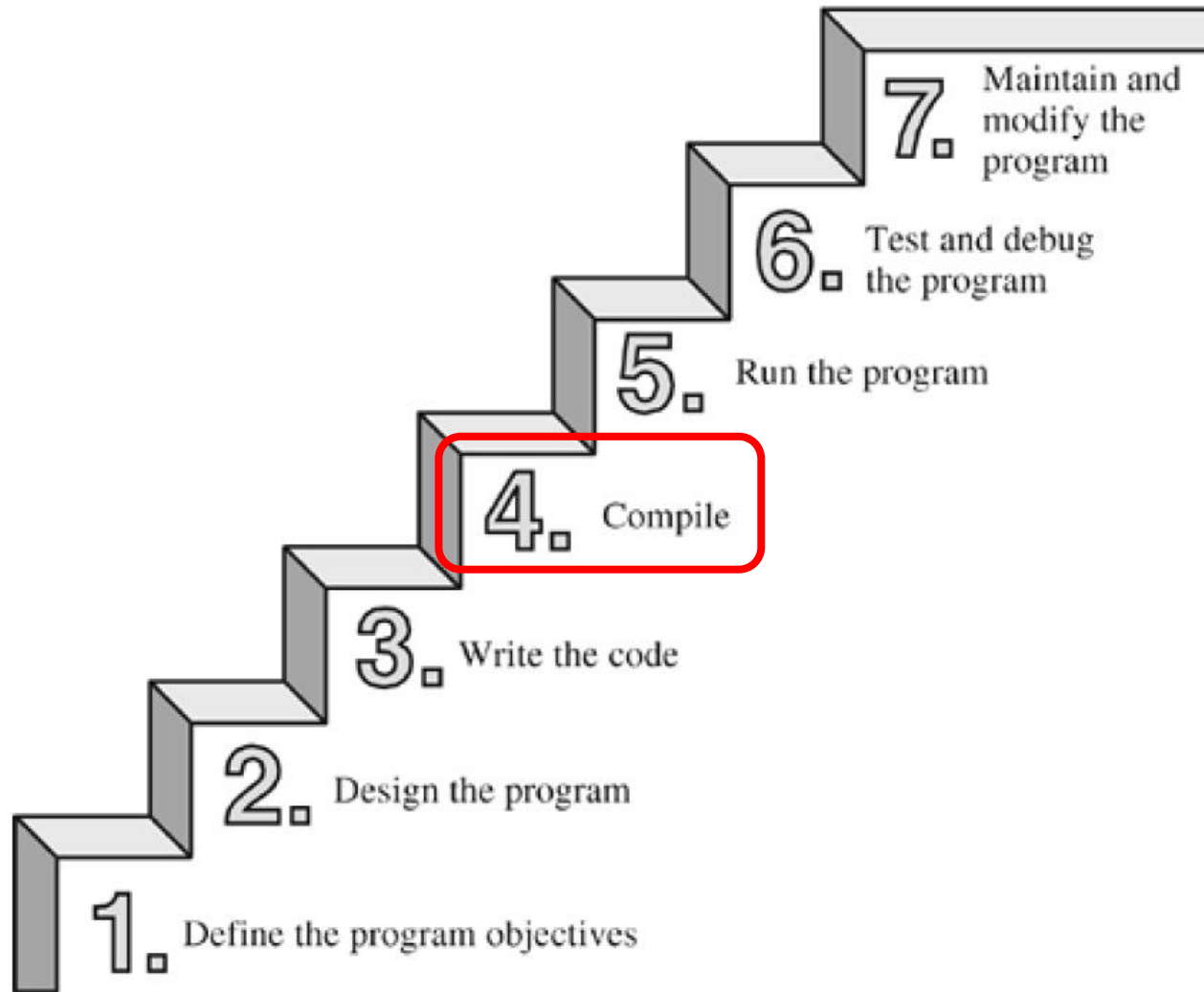
- 在C89标准中，可以接受：
 - `main()`
 - `void main()`
 - 上述两类`main`函数在最新的VS版本上可能出错
 - K&R书中使用的`main()`，新标准已经不推荐
- 在最新的C99标准中，规范化为：
 - `int main(void)` `//void可以省略`
 - `int main(int argc, char *argv[])`
 - 最好在`main`函数的最后加上`return`语句

C Program Structure

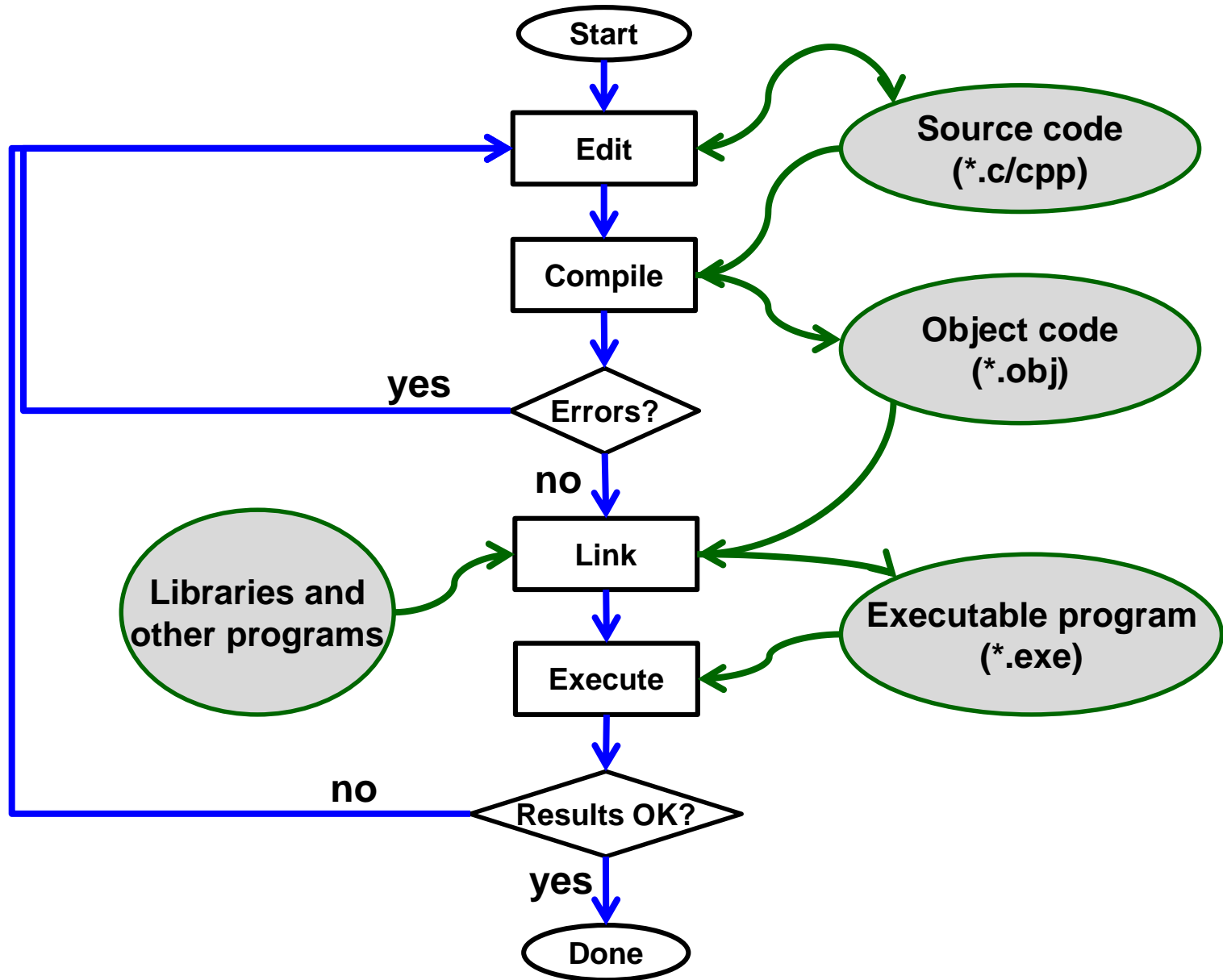
- `printf("hello, world\n");`
 - Instructs computer to perform an action
 - Specifically, prints the string of characters within quotes (" ")
 - 一条语句 statement
 - All statements must end with a semicolon (;)
 - 转义字符Escape character (\)
 - \n newline (换行符)

```
/* A Simple C Program */  
#include <stdio.h>  
  
int main()  
{  
    printf("hello, world\n");  
    return 0;  
}
```

Seven Steps of Programming



Typical Steps for Programming in C

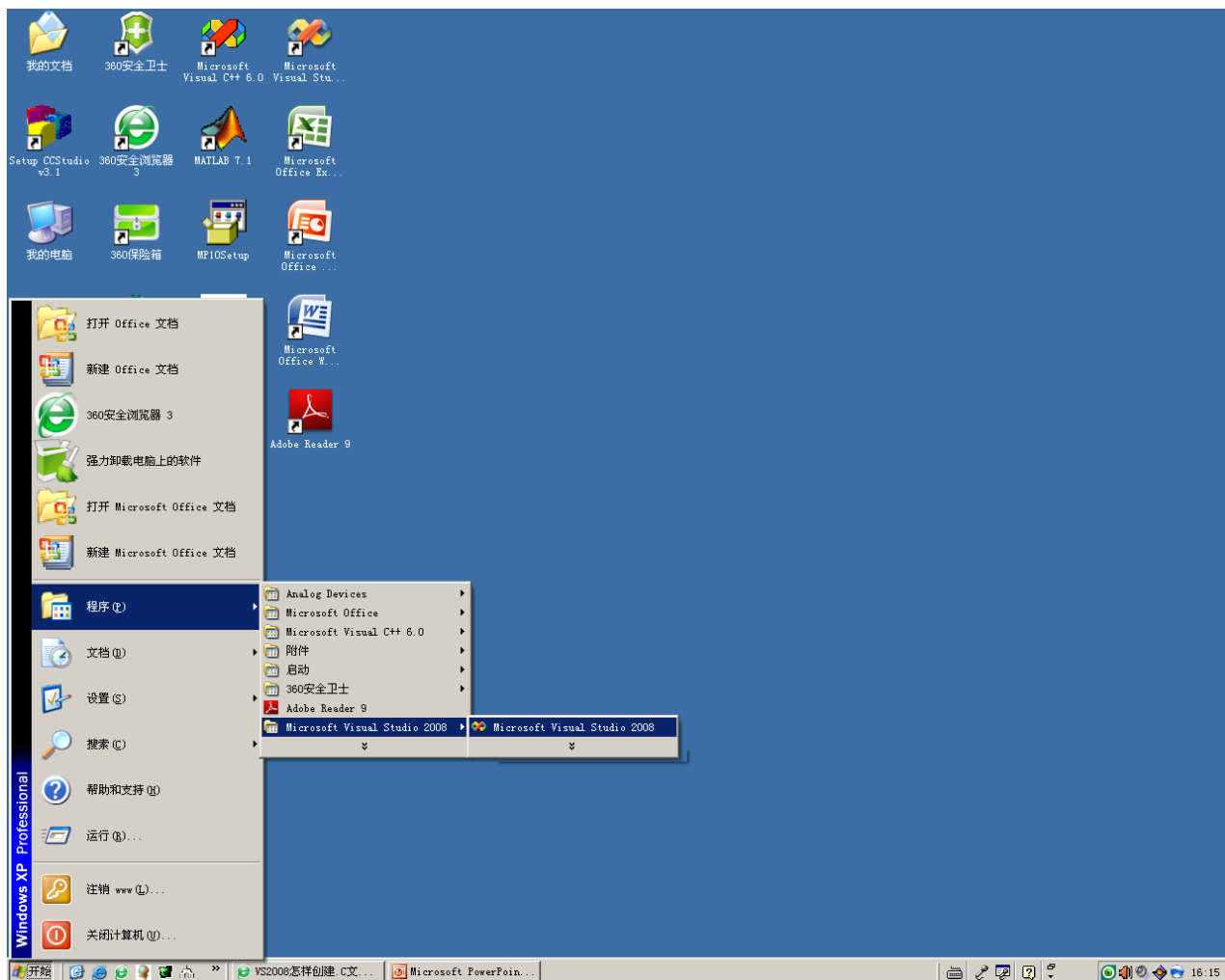


Brief Introduction to C

- Introduction to C
- A Simple C Program
- **Build VC Programs**

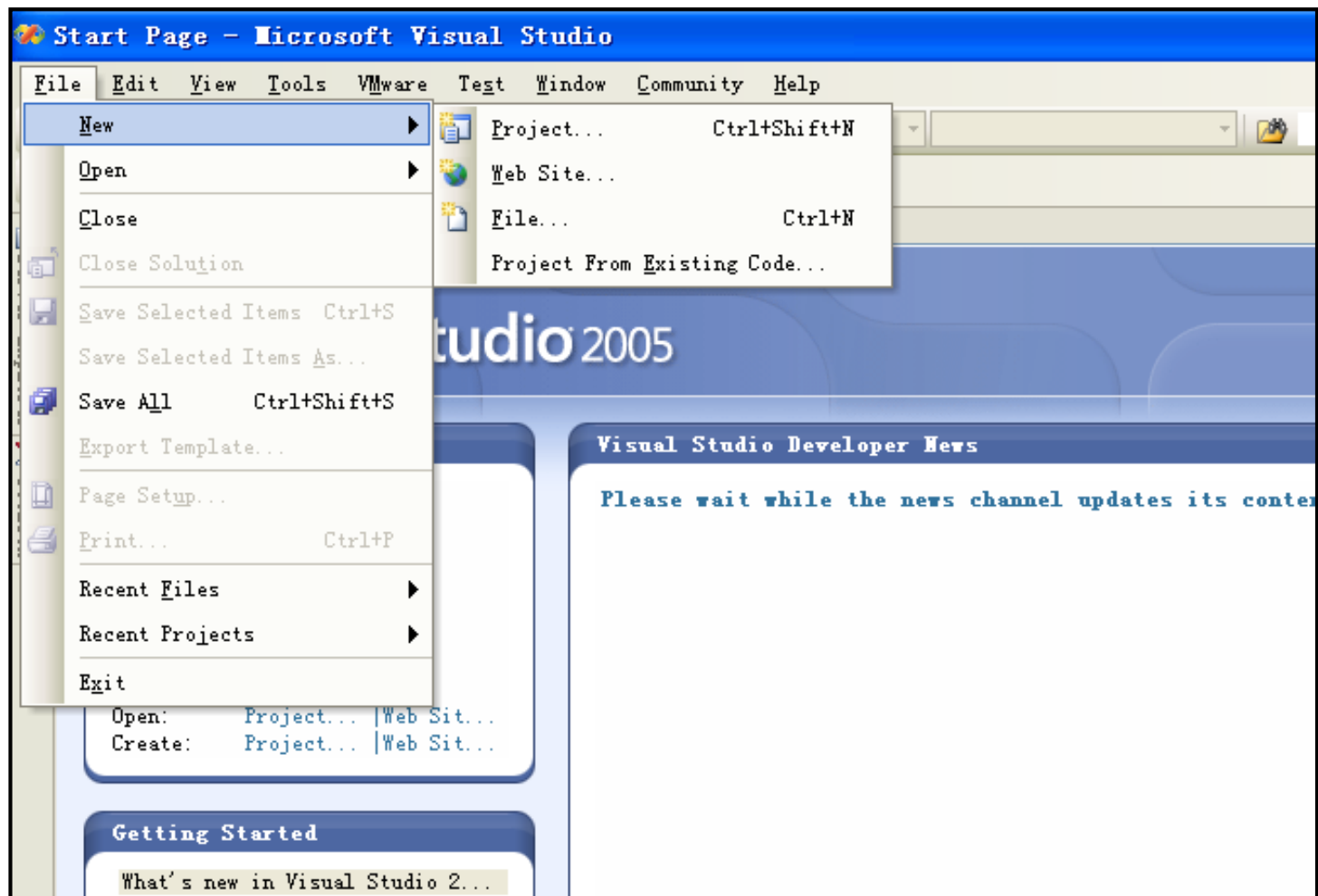
创建一个VC工程

1. 打开Microsoft Visual Studio 2008



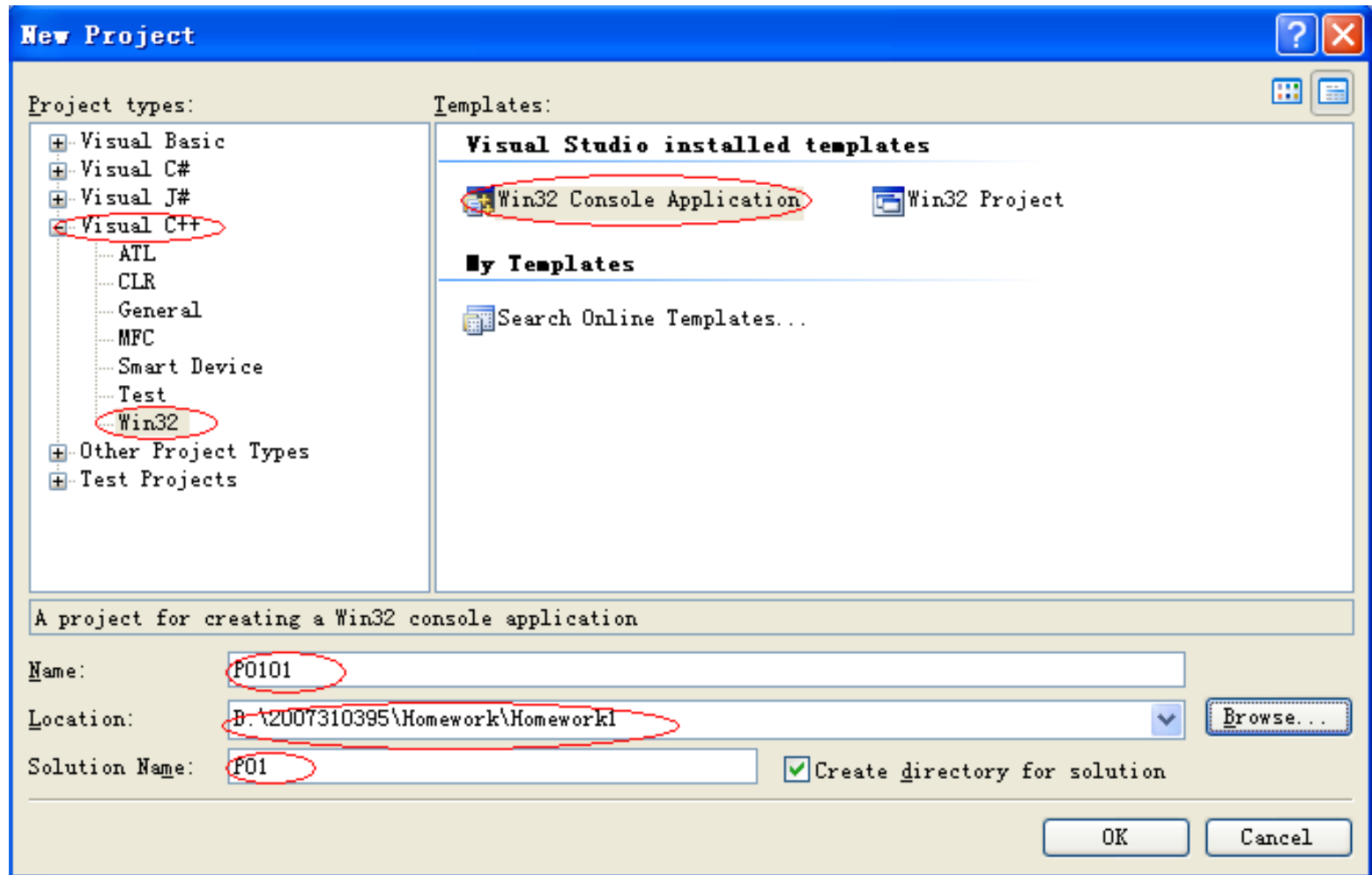
创建一个VC工程

2. 选择File->New->Project



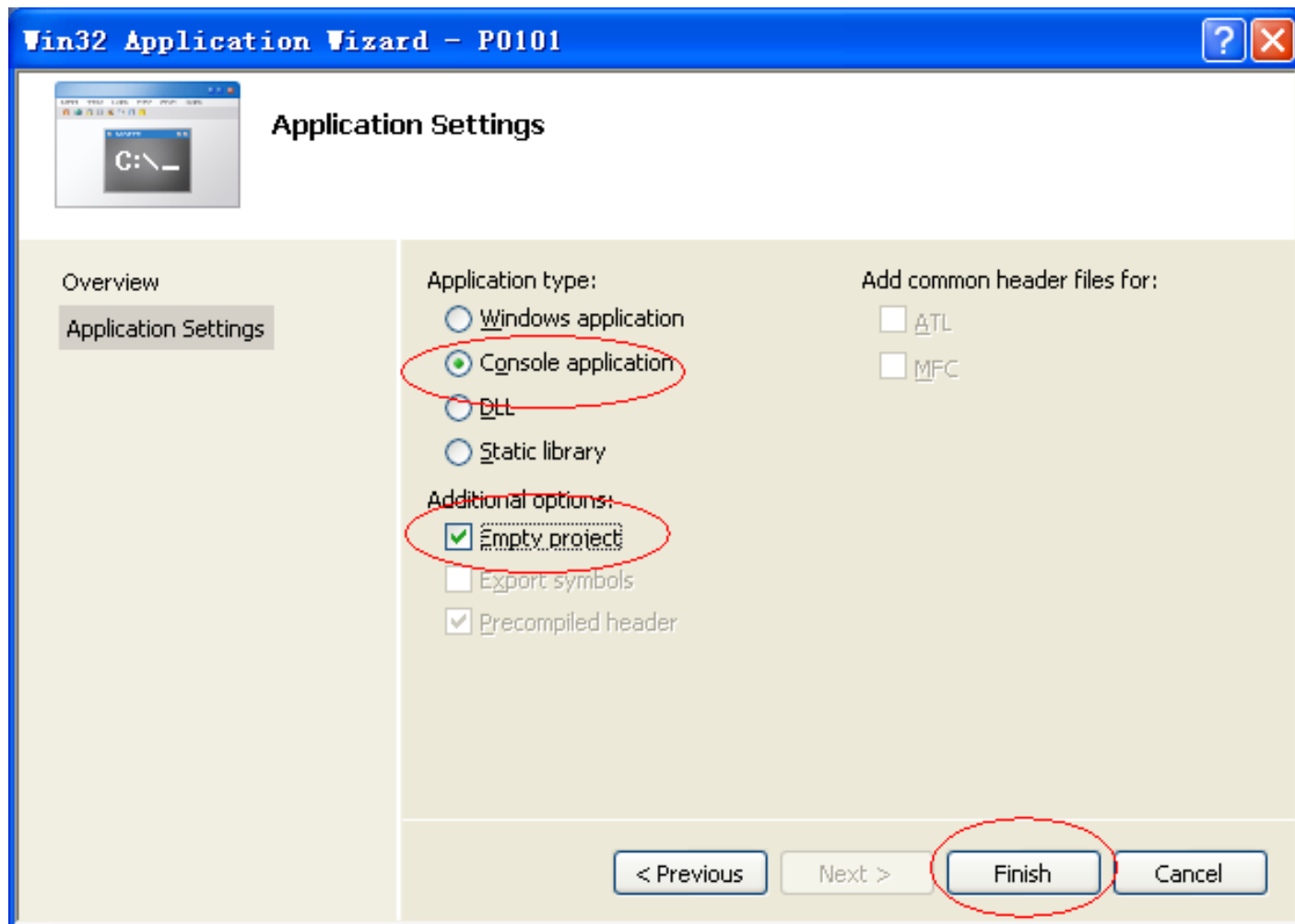
创建一个VC工程

3. 创建Visual C++ 下Win32 Console Application



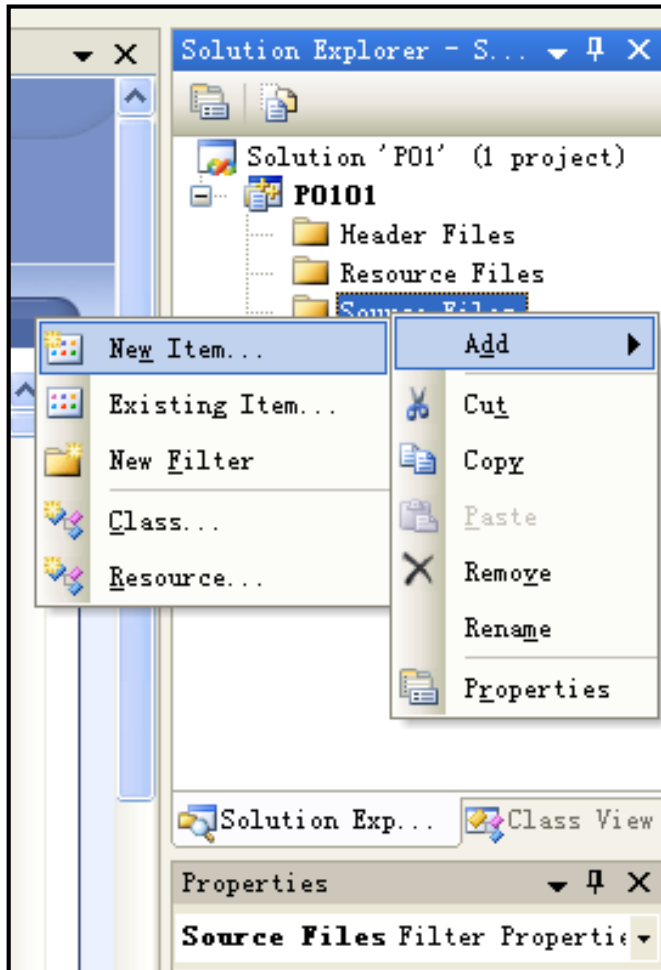
创建一个VC工程

4. 选择Console application以及empty project



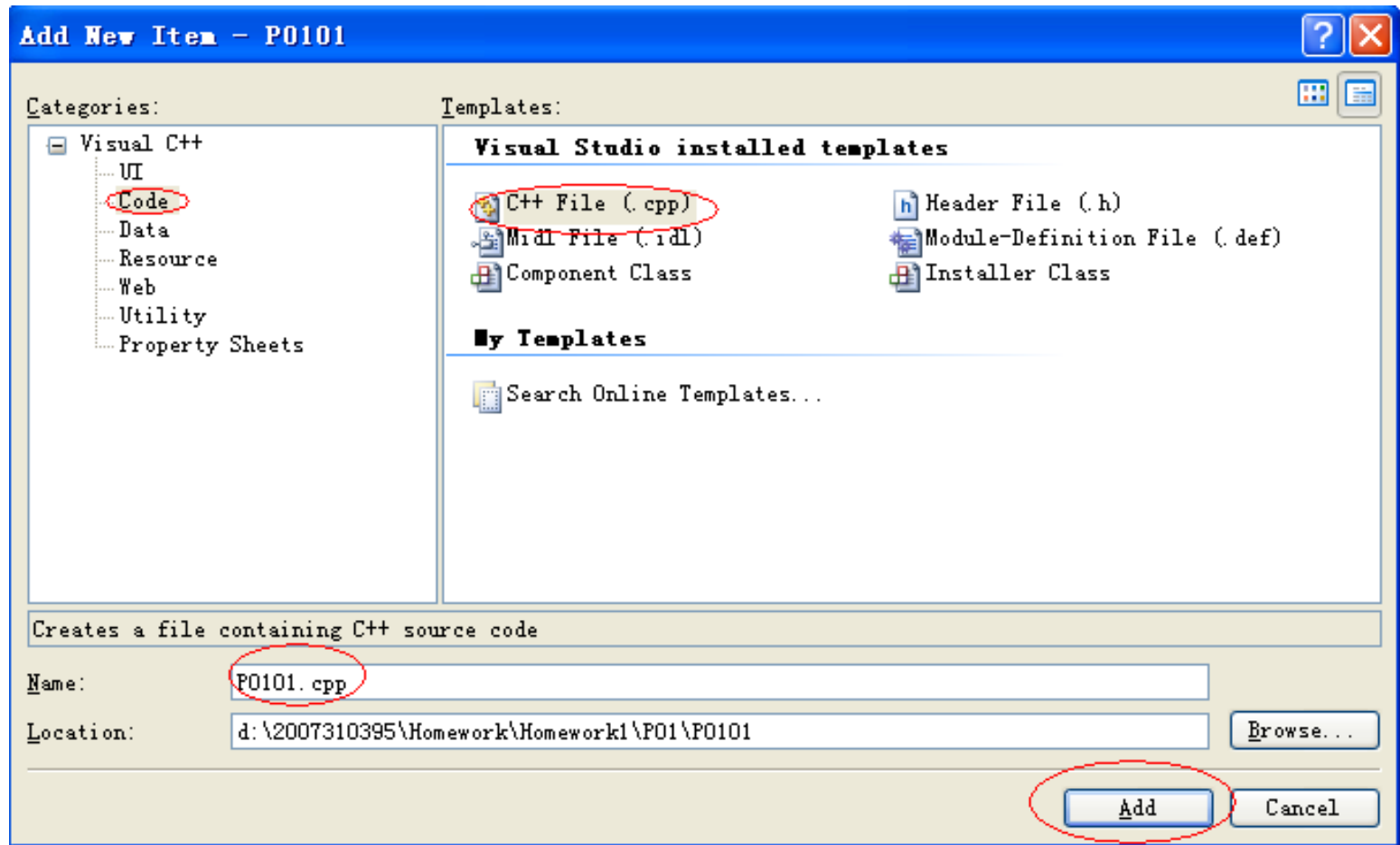
创建一个VC工程

5. 在新建好的工程中的Solution Explore中，右键点击Source Files文件夹，选择Add->New Item



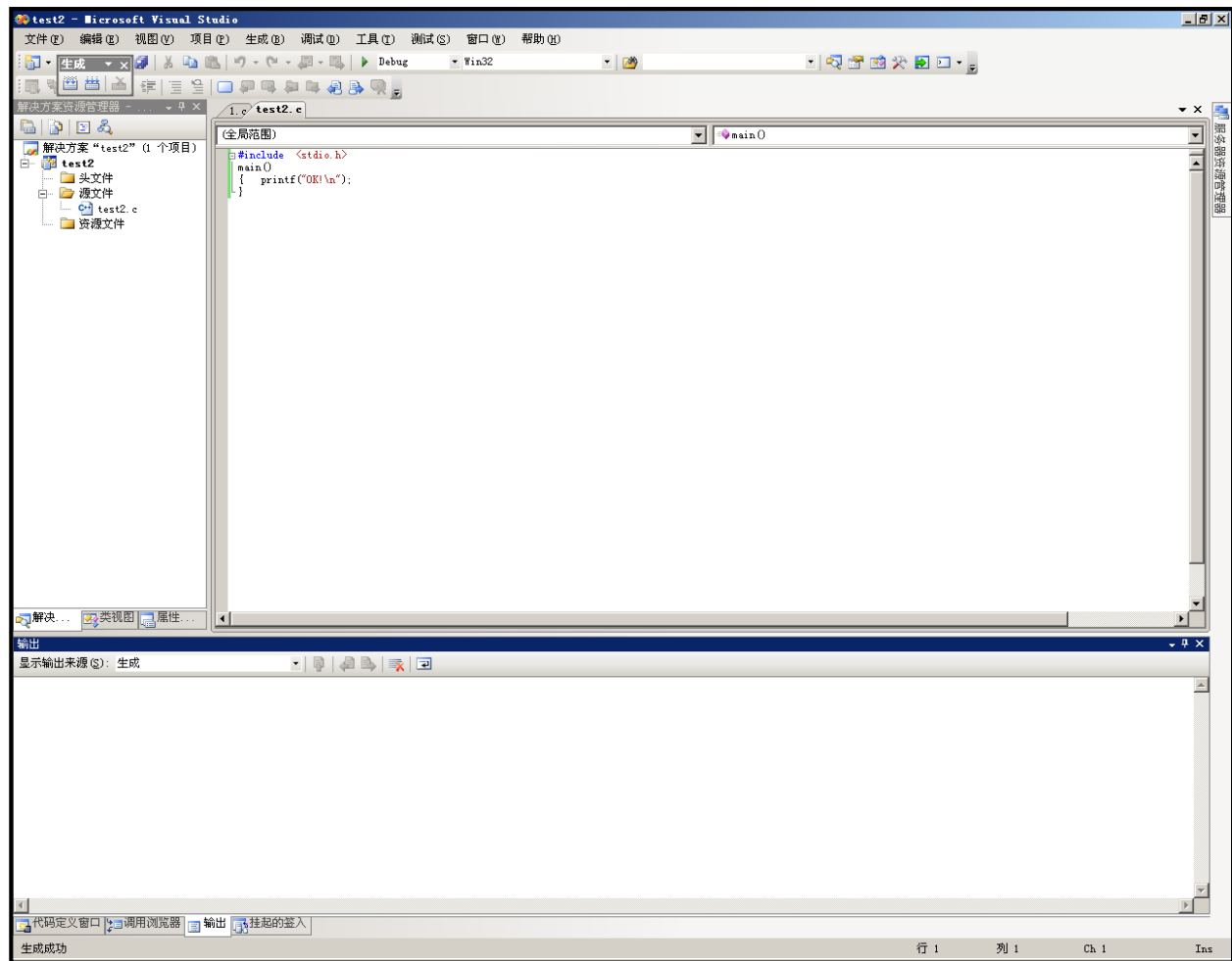
创建一个VC工程

6. 在出现的对话框中选择C++ File，并根据题目中要求的工程名命名源程序。如P0101.cpp



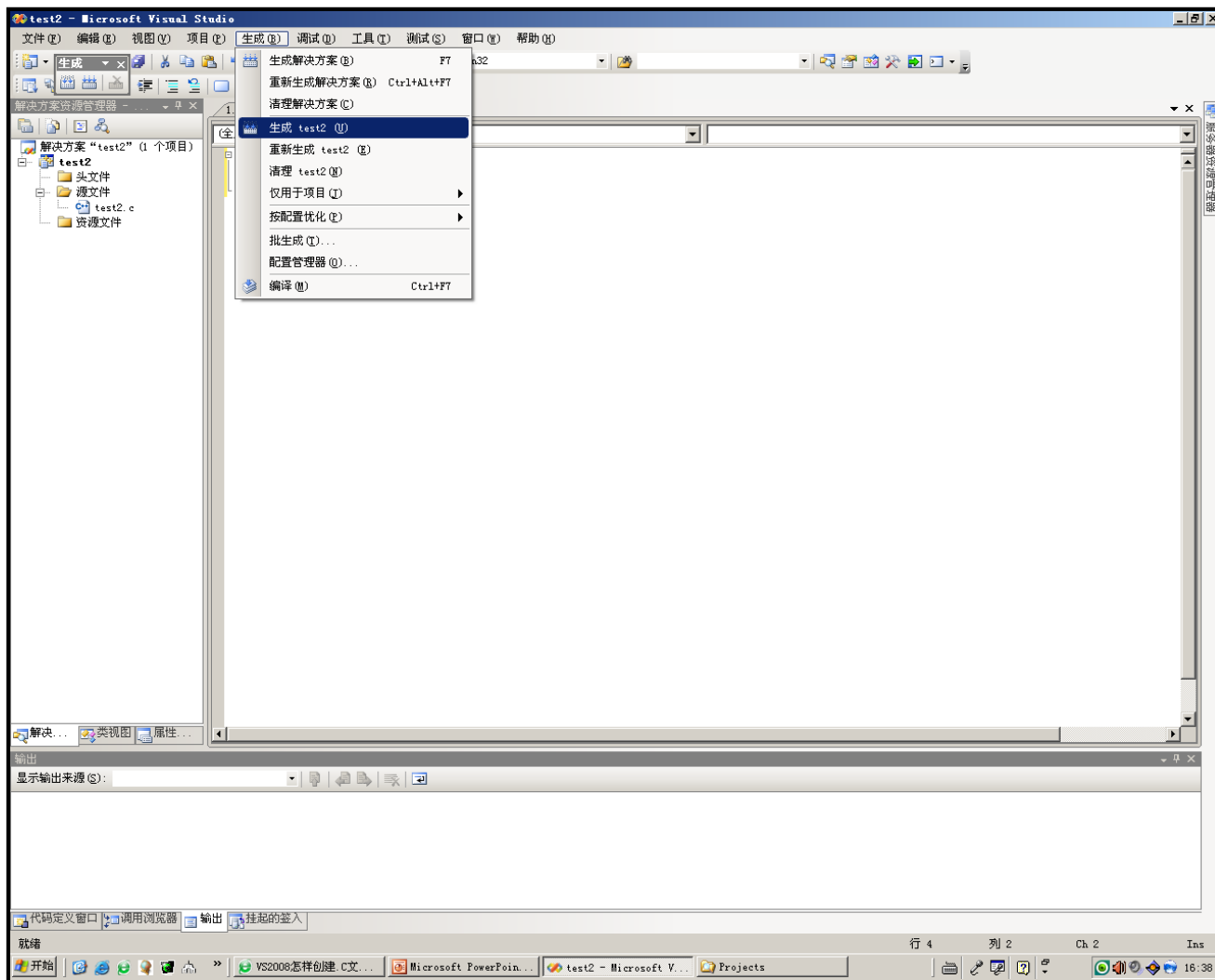
创建一个VC工程

7. 在新建的cpp文件中编写你的程序。



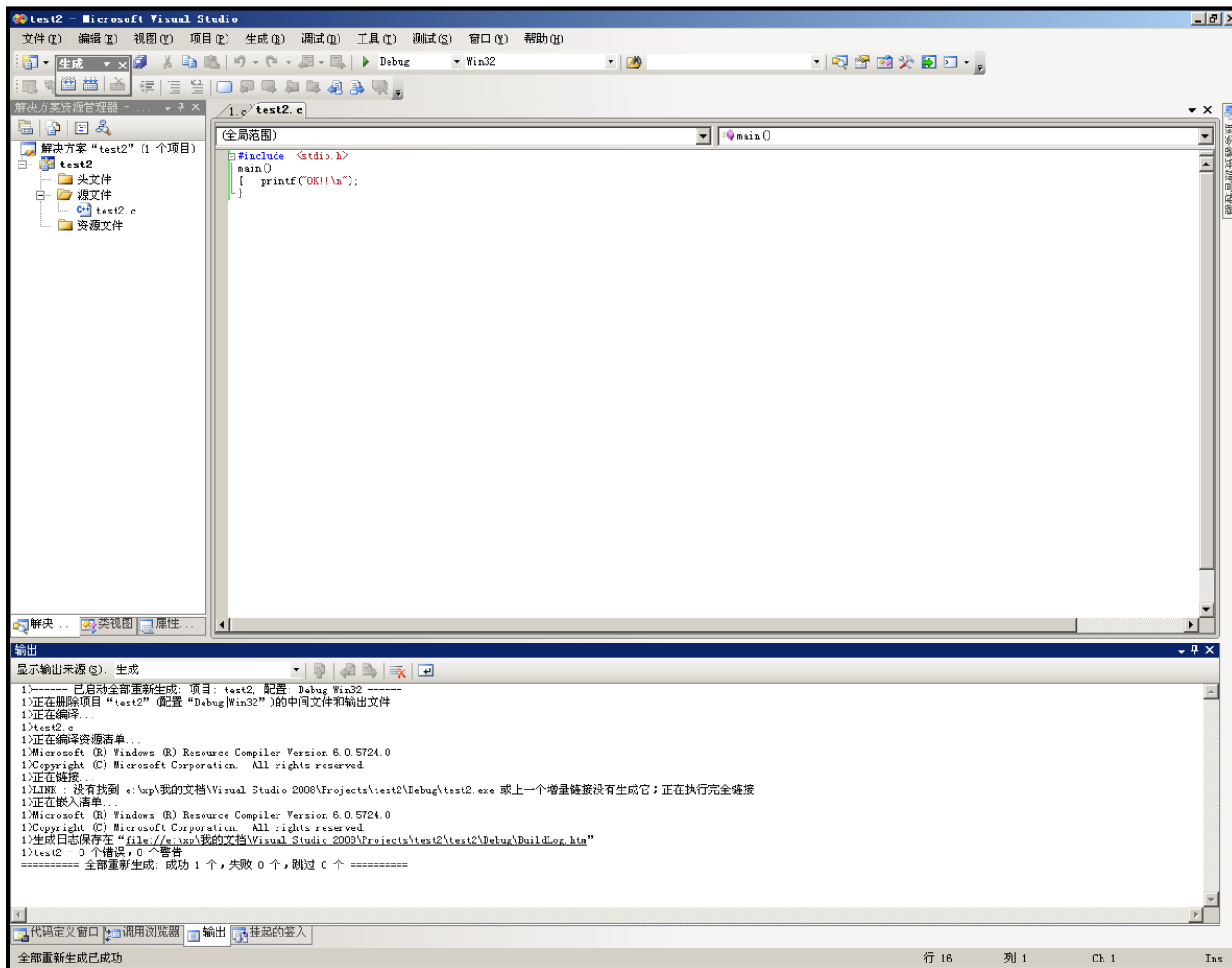
创建一个VC工程

8. 生成项目



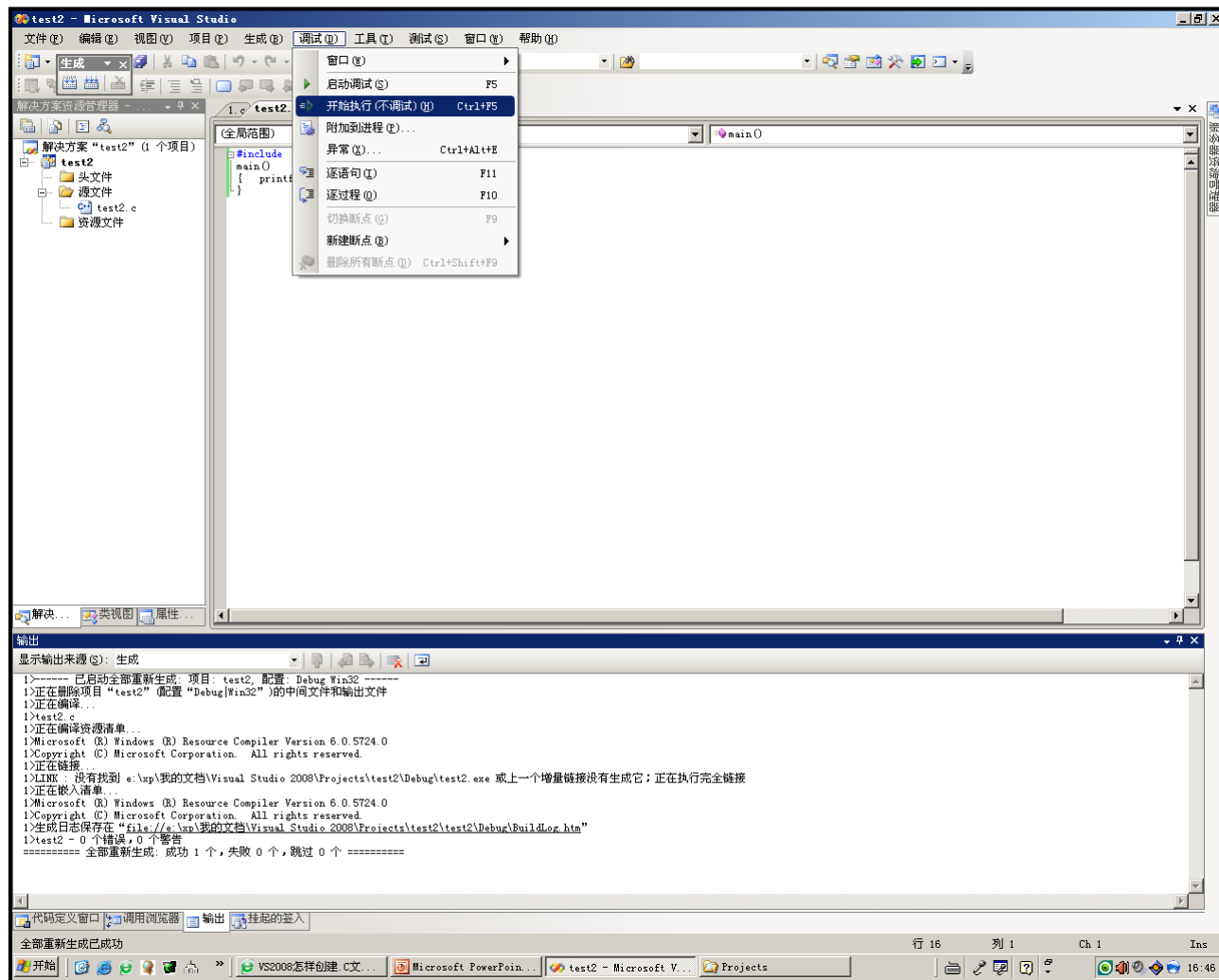
创建一个VC工程

9. 生成项目成功



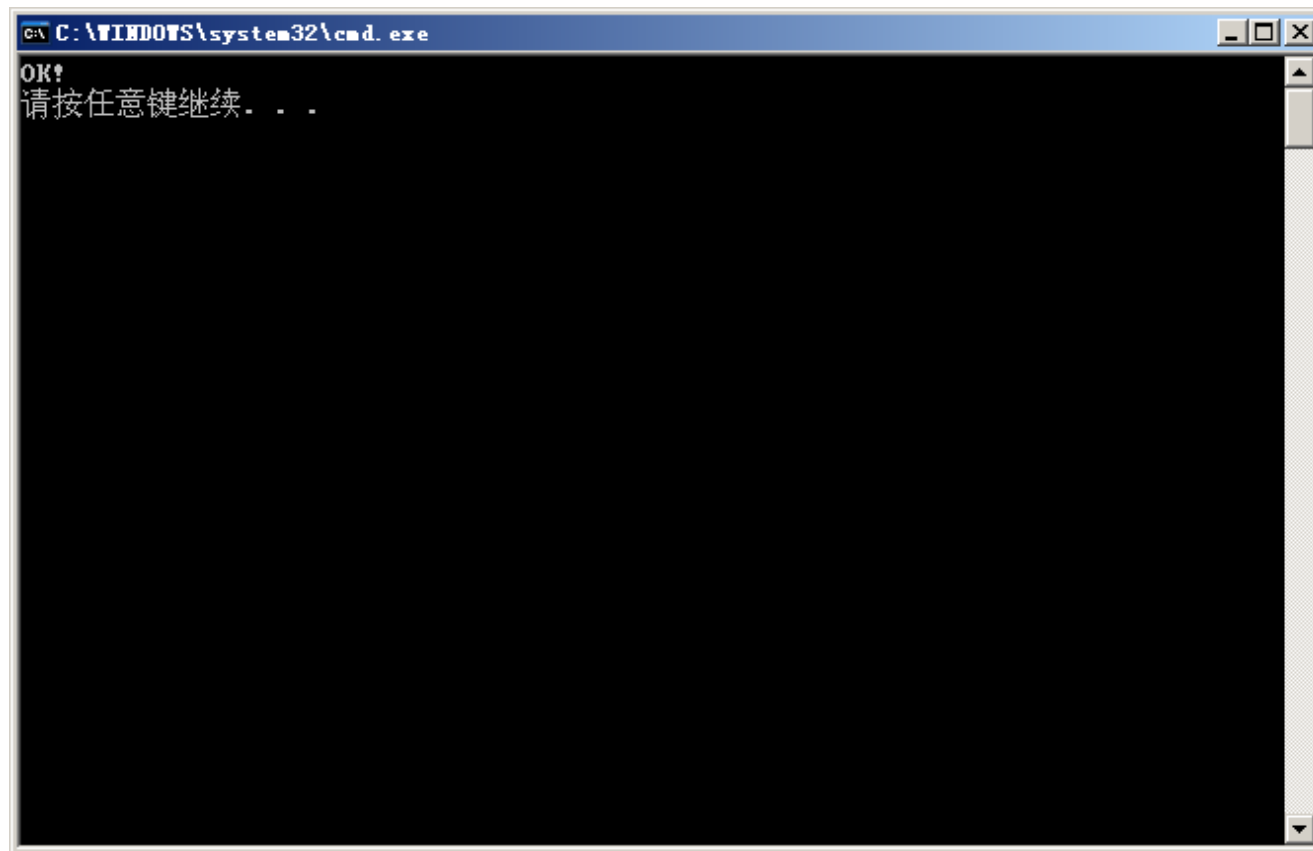
创建一个VC工程

10. 程序运行，通过Debug菜单或者F5（或者Ctrl+F5）编译运行程序。



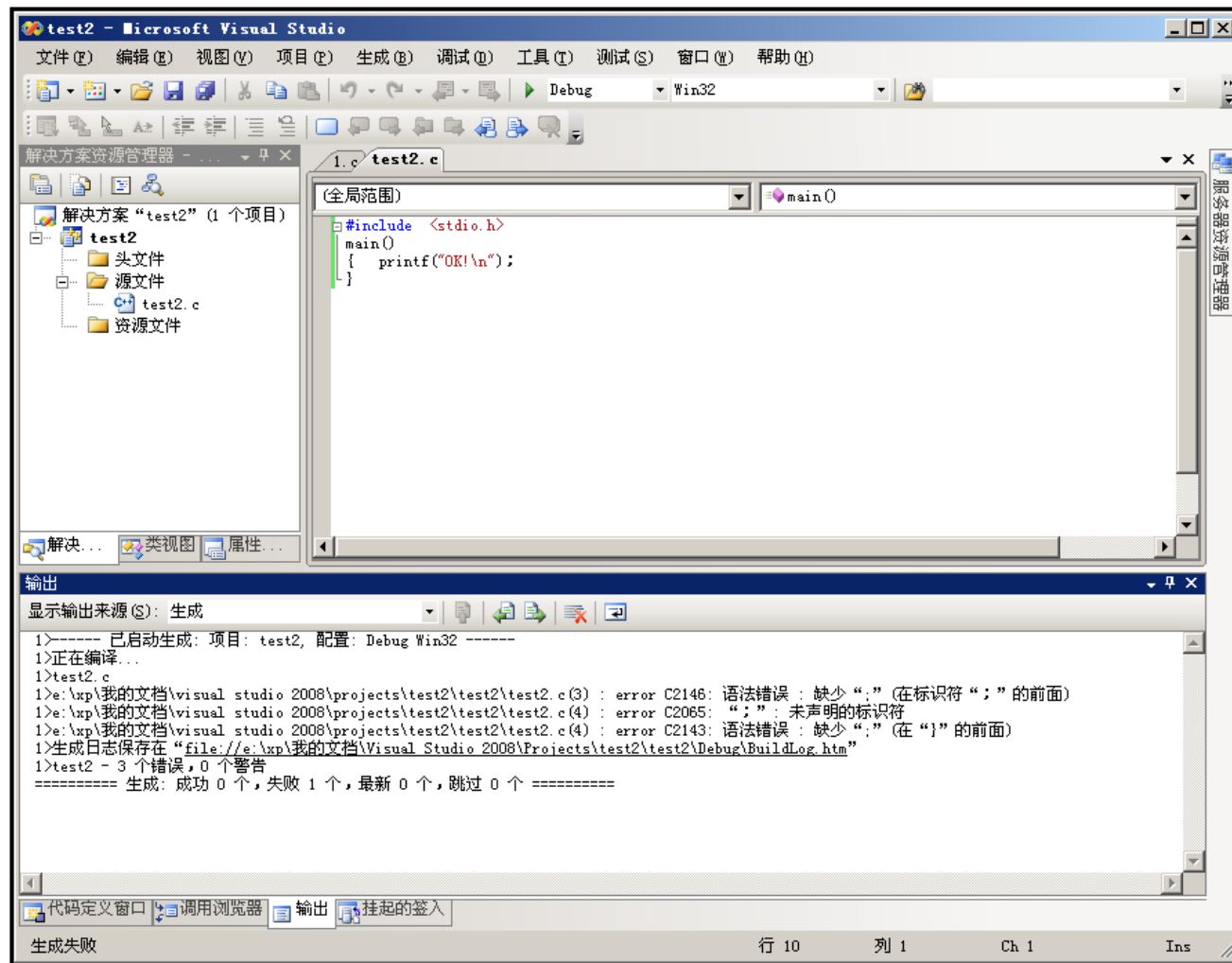
创建一个VC工程

11. 运行结果



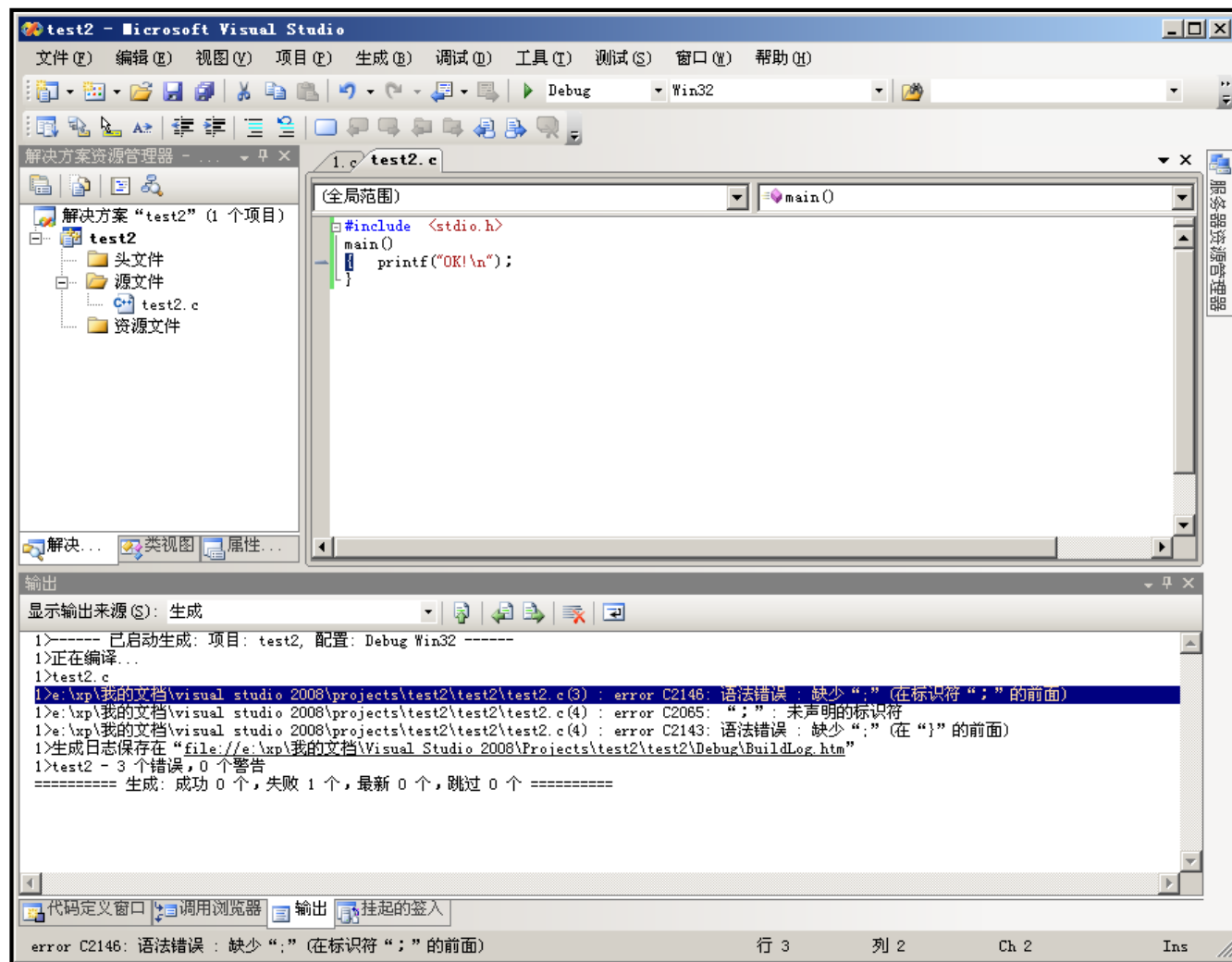
创建一个VC工程

12. 生成项目失败



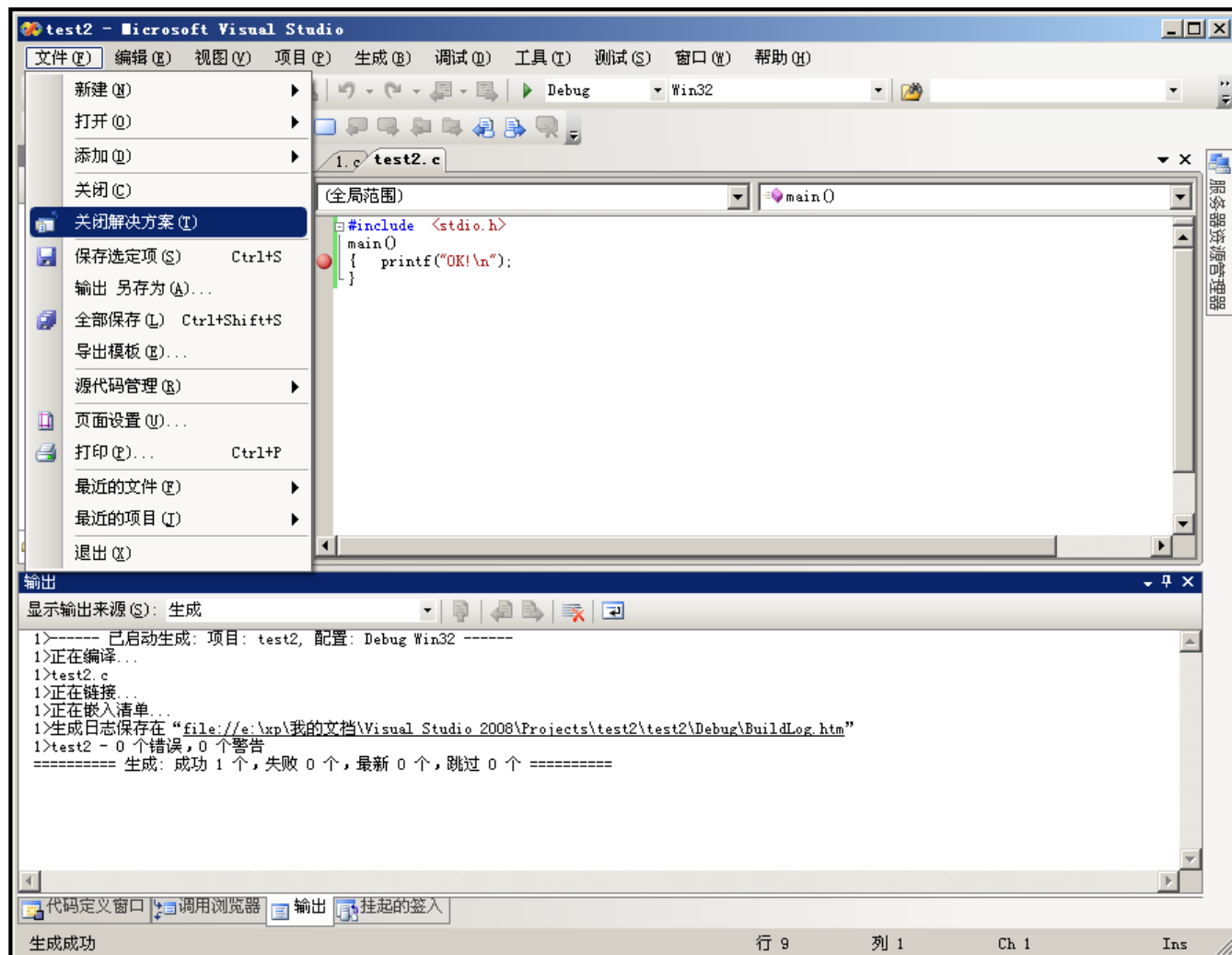
创建一个VC工程

13. 显示出错行



创建一个VC工程

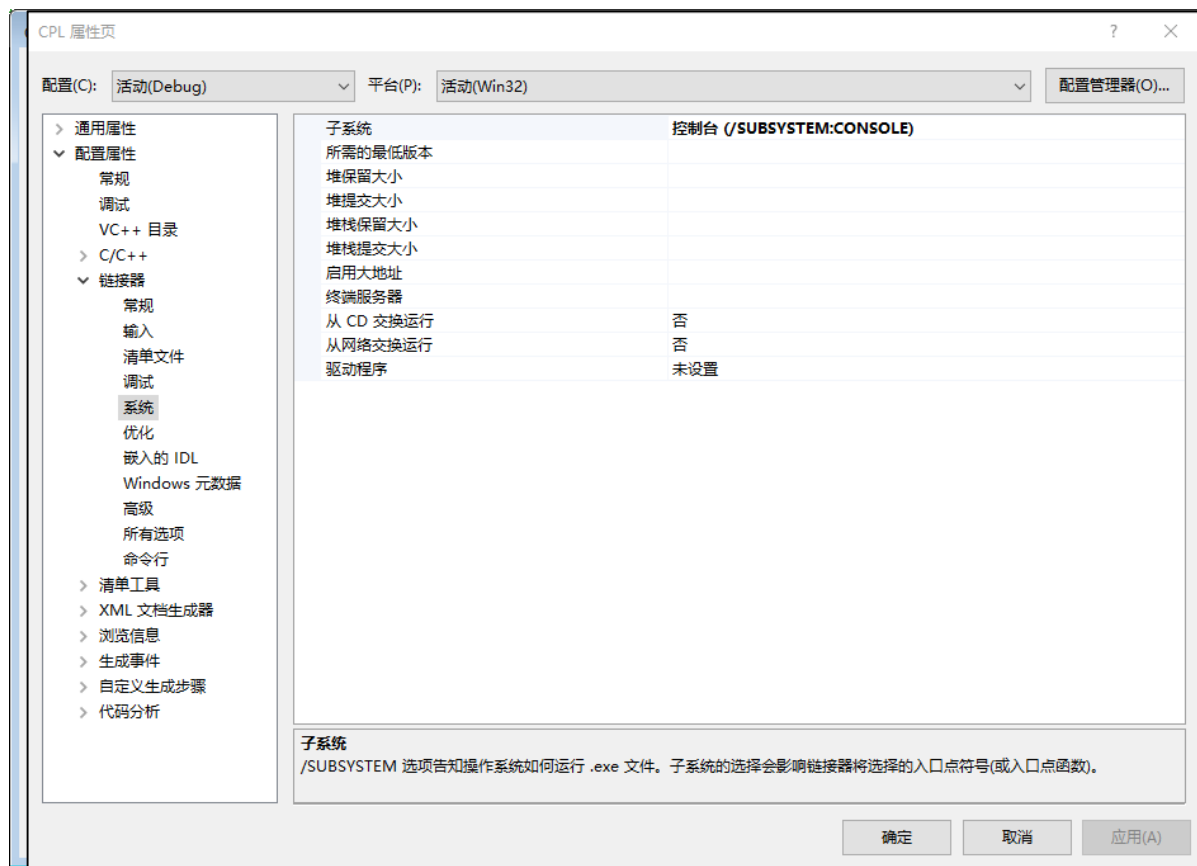
14. 关闭解决方案



VS控制台程序输出窗口闪退问题

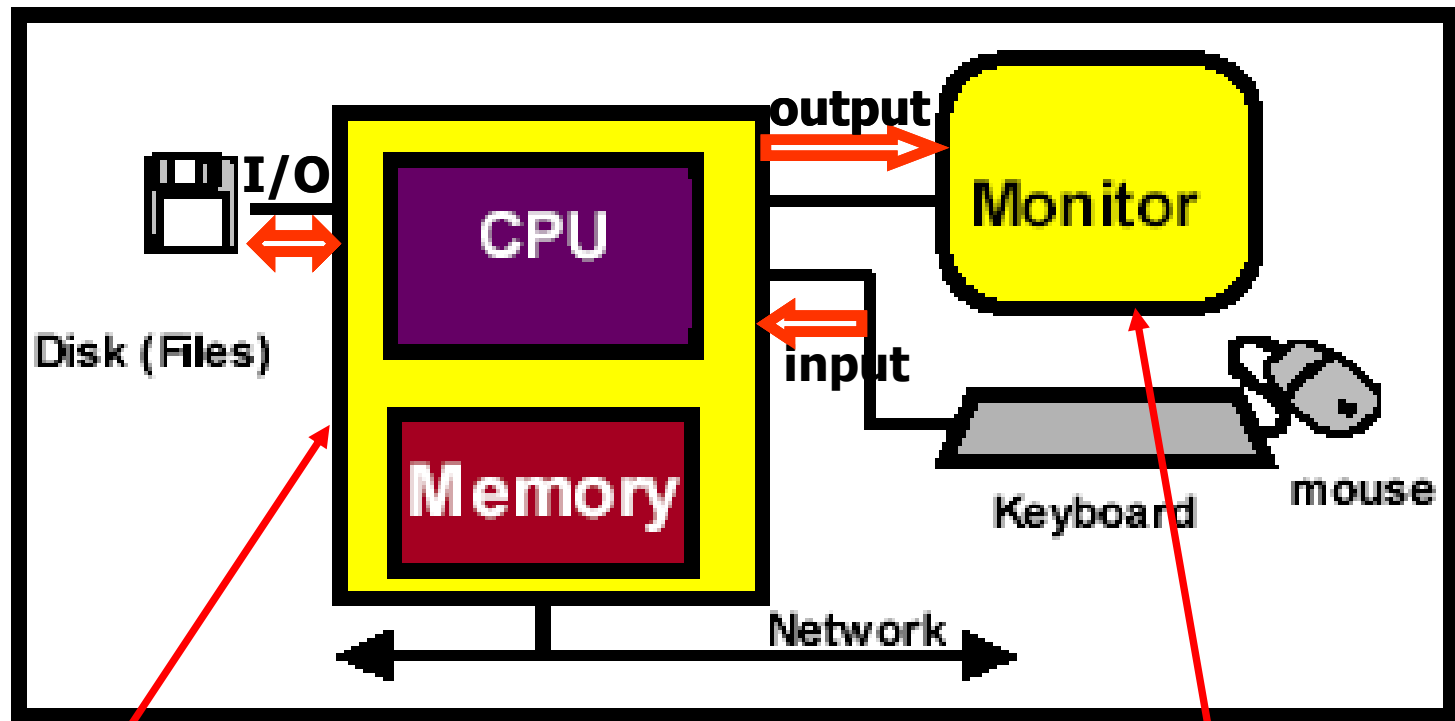
解决办法:

1. Ctrl+F5
2. `system("pause");`
3. C中, 使用 `getchar();`
4. C++中, 使用 `cin.get()`
5. F5在Debugging模式, 加断点调试



在解决方案资源管理器中找到工程, 在工程上右键--->属性--->配置属性-->链接器--->系统--->子系统 (在窗口右边) --->下拉框选择控制台 (/SUBSYSTEM:CONSOLE)

Formatted Output and Input 输入与输出



核心部件

I/O: 相对于内存

I/O设备

- 输入/输出的实现：函数调用

- `int printf(const char *, ...);`
- `int scanf(const char *, ...);`
- `int putchar(int c);`
- `int getchar(void);`

printf 函数

功能：向终端显示器输出若干个各种设定类型的数据

格式：printf(“格式控制字符串”，表达式列表)

```
int m;
```

```
double pi;
```

```
pi = 3.14;
```

```
m = 2;
```

```
printf("%d 乘以 %f 等于 %f. \n",  
        m, pi, (double)m * pi);
```

格式说明

普通字符

2 乘以 3.140000 等于 6.280000。

格式字符

格式字符	含义
d、i	输出十进制整数
o	以八进制数形式输出整数
x、X	以十六进制数形式输出整数（不带符号）
u	以十进制形式输出输出 unsigned 型数据
c	输出一个字符
s	输出一个字符串
f	以小数的形式输出一个实数
e、E	以指数形式输出实数
g、G	输出实数，根据数值大小自动选 f 或 e 格式

scanf 函数

功能：从键盘输入若干个各种类型的数据。

格式：scanf(“格式控制字符串”，地址列表);

```
int    i;  
float  fp;  
  
scanf("%d, %f", &i, &fp);  
printf("%d %f", i, fp);
```

15, 3.1415

15 3.141500

1. 取地址运算符("&")不能少
2. 普通字符(如逗号“,”)须输入

C++输入输出流 (cout, cin)

输入流 `cin` 语句: `cin>>变量1>>变量2>>.....>>变量n;`

输出流 `cout` 语句: `cout<<表达式1<<表达式2<<.....<<表达式n;`

- 包含头文件: `#include<iostream>`
- 使用名字空间: `using namespace std;`

```
#include<iostream>
using namespace std;
int main()
{
    cout << "hello world!" << endl;
    return 0;
}
```

hello world!

请按任意键继续...

A Tutorial Introduction

- **Variables and Arithmetic Expressions**
- **The Looping Statement**
- **Symbolic Constants**
- **Character Input and Output**
- **Arrays**
- **Functions**

A Tutorial Introduction

- **Variables and Arithmetic Expressions**
- The Looping Statement
- Symbolic Constants
- Character Input and Output
- Arrays
- Functions

Variables and Arithmetic Expressions

- 变量和算术表达式
- Example 1: 温度转换计
- 华氏温度→摄氏温度
 $^{\circ}C = (5/9) (^{\circ}F - 32)$

0	-17
20	-6
40	4
60	15
80	26
100	37
120	48
140	60
160	71
180	82
200	93
220	104
240	115
260	126
280	137
300	148

Variables

- 使用了新概念：注释、声明变量、算术表达式、循环、格式化输出等。
- 注释在程序编译时会被忽略，但它可以使程序更易于理解

```
#include <stdio.h>
/* print Fahrenheit-Celsius table
   for fahr = 0, 20, ..., 300 */
int main()
{
    int fahr, celsius;
    int lower, upper, step;

    lower = 0;           // lower limit
    upper = 300;         // upper limit
    step = 20;           // step size

    fahr = lower;
    while (fahr <= upper){
        celsius = 5 * (fahr-32) / 9;
        printf("%d\t%d\n", fahr, celsius);
        fahr = fahr + step;
    }
    return 0;
}
```

Variables (变量)

- **Variables**
 - declaration (声明)
 - assignment (赋值)
- **Basic data types:**
 - char, short, int, long, float, double
- **C程序中所有变量必须先声明后使用**

```
#include <stdio.h>
/* print Fahrenheit-Celsius table
   for fahr = 0, 20, ..., 300 */
int main()
{
    int fahr, celsius;
    int lower, upper, step;

    lower = 0;           // lower limit
    upper = 300;         // upper limit
    step = 20;           // step size

    fahr = lower;
    while (fahr <= upper){
        celsius = 5 * (fahr-32) / 9;
        printf("%d\t%d\n", fahr, celsius);
        fahr = fahr + step;
    }
    return 0;
}
```

Arithmetic Expressions (算术表达式)

- 算术运算符: $+$ 、 $-$ 、 $*$ 、 $/$ 、 $\%$
- 关系运算符: $>$ 、 $<$ 、 $==$ 、 $>=$ 、 $<=$ 、 $!=$
- 赋值运算符: $=$



Why not $(5/9)*(fahr-32)$?

- 整数除以整数结果取整后为0, 即: $5/9=0$, 可改为: $5.0/9.0$

```
#include <stdio.h>
/* print Fahrenheit-Celsius table
   for fahr = 0, 20, ..., 300 */
int main()
{
    int fahr, celsius;
    int lower, upper, step;

    lower = 0;           // lower limit
    upper = 300;          // upper limit
    step = 20;            // step size

    fahr = lower;
    while (fahr <= upper){
        celsius = 5 * (fahr-32) / 9;
        printf("%d\t%d\n", fahr, celsius);
        fahr = fahr + step;
    }
    return 0;
}
```


A Tutorial Introduction

- Variables and Arithmetic Expressions
- **The Looping Statement**
- Symbolic Constants
- Character Input and Output
- Arrays
- Functions

The Looping Statement

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

```
/* print Fahrenheit-Celsius table */
```

```
int main()
```

```
{
```

```
    int fahr;
```

```
    for (fahr = 0; fahr <= 300; fahr = fahr + 20)
```

```
        printf("%3d %6.1f\n", fahr, (5.0/9.0)*(fahr-32));
```

```
    return 0;
```

```
}
```

0	-17.8
20	-6.7
40	4.4
60	15.6
80	26.7
100	37.8
120	48.9

.....

- 在 **for** 循环中，只用一个 **int** 类型的变量 **fahr**
- 温度的上限、下限和步长都是常量
- 而计算摄氏温度的表达式成为 **printf** 函数的第三个参数（表达式）

The For Statement

```
#include <stdio.h>

/* print Fahrenheit-Celsius table */
int main()
{
    int fahr;
    for (fahr = 0; fahr <= 300; fahr = fahr + 20)
        printf("%3d %6.1f\n", fahr, (5.0/9.0)*(fahr-32));
    return 0;
}
```

```
for (init; condition; update)
{
    statements;
}
```

- 事前须初始化、事后要更新
- 先条件测试再执行
- 计数驱动

A Tutorial Introduction

- Variables and Arithmetic Expressions
- The Looping Statement
- **Symbolic Constants**
- Character Input and Output
- Arrays
- Functions

Symbolic Constants

```
#include <stdio.h>
#define LOWER 0      // lower limit
#define UPPER 300    // upper limit
#define STEP 20      // step size
/* print Fahrenheit-Celsius table */
int main()
{
    int fahr;
    for (fahr = LOWER; fahr <= UPPER; fahr = fahr + STEP)
        printf("%3d %6.1f\n", fahr, (5.0/9.0)*(fahr-32));
    return 0;
}
```

Symbolic Constants

```
#include <stdio.h>
#define LOWER 0      // lower limit
#define UPPER 300    // upper limit
#define STEP 20      // step size
/* print Fahrenheit-Celsius table */
int main()
{
    int fahr;
    for (fahr = LOWER; fahr <= UPPER; fahr = fahr + STEP)
        printf("%3d %6.1f\n", fahr, (5.0/9.0)*(fahr-32));
    return 0;
}
```

- Symbolic Constants 符号常量
 - **#define** name replacement-text
 - They are not variables, so not appear in declarations
 - 通常用 **UPPER CASE (大写字母)** 来与变量区分

A Tutorial Introduction

- Variables and Arithmetic Expressions
- The Looping Statement
- Symbolic Constants
- **Character Input and Output**
- Arrays
- Functions

Character Input and Output

- **Example 2:**
 - 屏幕打印字符串并输出
 - `getchar()`: read a character
 - `putchar()`: print a character

```
hello world!  
hello world!
```

```
#include <stdio.h>  
/* copy input to output */  
int main()  
{  
    int c;  
    c = getchar();  
    while (c != '\n'){  
        putchar(c);  
        c = getchar();  
    }  
    return 0;  
}
```

不等于

A Tutorial Introduction

- Variables and Arithmetic Expressions
- The Looping Statement
- Symbolic Constants
- Character Input and Output
- **Arrays**
- Functions

Arrays (数组)

- **Example 3: Print days for each month**

```
Month 1 has 31 days.  
Month 2 has 28 days.  
Month 3 has 31 days.  
Month 4 has 30 days.  
Month 5 has 31 days.  
Month 6 has 30 days.  
Month 7 has 31 days.  
Month 8 has 31 days.  
Month 9 has 30 days.  
Month 10 has 31 days.  
Month 11 has 30 days.  
Month 12 has 31 days.
```

```
#include <stdio.h>  
/* print the days for each month */  
int main()  
{  
    int days[12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};  
    int i;  
    for (i = 0; i < 12; i++)  
        printf("Month %d has %d days.\n", i+1, days[i]);  
    return 0;  
}
```

Arrays (数组)

- Array subscripts always start at **zero** in C
 - e.g. `days[0]`, `days[1]`, ..., `days[11]`

```
#include <stdio.h>
/* print the days for each month */
int main()
{
    int days[12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    int i;
    for (i = 0; i<12; i++)
        printf("Month %d has %d days.\n", i+1, days[i]);
    return 0;
}
```

A Tutorial Introduction

- Variables and Arithmetic Expressions
- The Looping Statement
- Symbolic Constants
- Character Input and Output
- Arrays
- **Functions**

Functions (函数)

- 一次编程可多次调用

- Example 4:
power function
幂函数: m^n

2	0	1
2	1	2
2	2	4
2	3	8
2	4	16
2	5	32
2	6	64
2	7	128
2	8	256
2	9	512

```
#include <stdio.h>
int power(int m, int n);
/* test power function */
int main()
{
    int i, x=2;

    for (i=0; i < 10; ++i)
        printf("%d %d %d\n", x, i, power(x,i));
    return 0;
}

/* power: raise base to n-th power */
int power(int m, int n)
{
    int i, p=1;
    for (i = 1; i<=n; ++i)
        p = p * m;
    return p;
}
```

函数的定义

函数定义的一般形式:

```
<返回值类型> <函数名> (<0个或多个形参声明>)  
{  
    <数据声明部分>  
    <程序语句序列>  
}
```

Functions (函数)

actual argument
实参

formal argument
形参

```
#include <stdio.h>
int power(int m, int n);
/* test power function */
int main()
{
    int i, x=2;

    for (i=0; i < 10; ++i)
        printf("%d %d %d\n", x, i, power(x,i));
    return 0;
}
/* power: raise base to n-th power */
int power(int m, int n)
{
    int i, p=1;
    for (i = 1; i<=n; ++i)
        p = p * m;
    return p;
}
```

The diagram illustrates the flow of arguments between the `main` function and the `power` function. A solid green arrow originates from the `power(x,i)` call in the `main` function and points to the `int m` parameter in the `power` function definition. Another solid green arrow originates from the `power(x,i)` call and points to the `int n` parameter in the `power` function definition. A red dotted arrow originates from the `power(x,i)` call and points to the `int n` parameter in the `power` function definition, indicating the return value of the `power` function being passed back to the caller.

Lecture 1 - Summary

- **Topics covered:**

- **Syllabus**

- Schedule, Textbooks, Learning objectives

- **Brief introduction to C**

- Some Computing Fundamentals and History
 - How to edit, compile, and debug your first C programs?
 - Create a VC project

- **C programming fundamentals**

- Comments, `#include`, `main()`, `printf()`

- **A tutorial introduction to C**

Exercise

1. 使用VC编写一个程序，并编译、调试、运行、输出结果
2. 编写一个程序，输入一个摄氏温度，输出相应的华氏温度。在输出时，保留小数点后面两位。尝试修改本节课的示例程序完成该习题
3. **第一次上机时间提醒**
 - **上机时间：9月21日(周四), 下午13:00-16:00**
 - **上机地点：东配楼11区402, 406**