# LECTURE 1 C语言和编程入门

工学院18-19学年秋季学期计算概论（邓习峰班）课后辅导

讲师：陈婉雯

日期：2018/9/28

# 课程简介

# 自我介绍

- 陈婉雯，2015级理论与应用力学专业
- 联系邮箱：chenww2015@pku.edu.cn
- 课程资料：https://github.com/aliciachenw/18-19-

# 辅导计划

- 目标：良好的编程习惯+自己解决问题的能力
- 按照老师上课进度进行概念讲解、复习加拓展
- 在提交作业之后对作业进行讲解、补充额外习题
- 编程习题讲解方法：解题思路+同学案例
- 同学可以将自己编程中遇到的问题通过邮件/微信发给我：
  - 问题描述及你的测试输入和输出（可用截图）
  - 代码文件
- 辅导计划会按照同学们的需求修改
  - 讲课方式、风格
  - 讲课内容（比如有什么难点需要加深或者重复等）

# C语言及简单程序基础

# C语言简介

- C语言的特点：
- 简洁、强大
- 面向过程的语言：分析出解决问题的步骤，然后用函数一步一步实现
- 与底层密切相关的高级语言：地址运算、二进制数位运算、硬件端口……
- 可移植性好、运行效率高
- 语法限制不严格：数组下标越界、变量类型兼容、数据类型转换……
- 应用范围广

# 简单编程指导

# 编程七步曲

- 定义程序目标
- 设计程序

思考、在纸上记录

- 编写代码
- 编译
- 运行程序
- 测试和调试
- 维护和修改

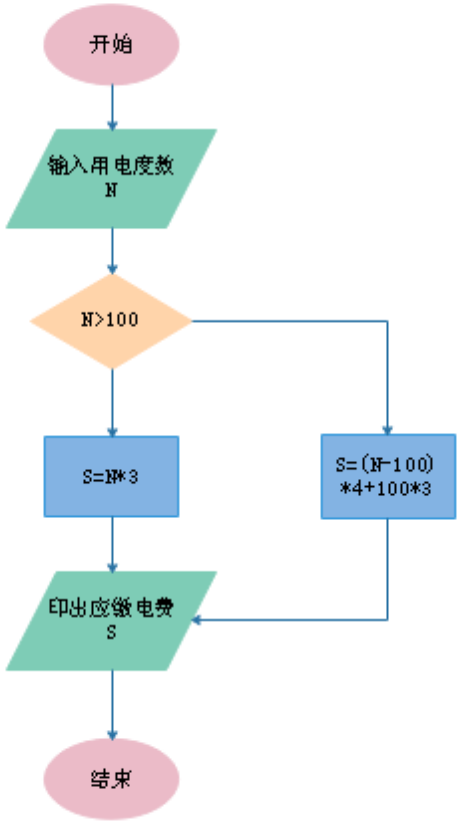
在IDE内完成

# 如何设计程序?

- 过程：把问题划分成子问题；按步骤解决问题
- 数据：表示数据、存储数据、处理数据
- 伪代码、算法流程图

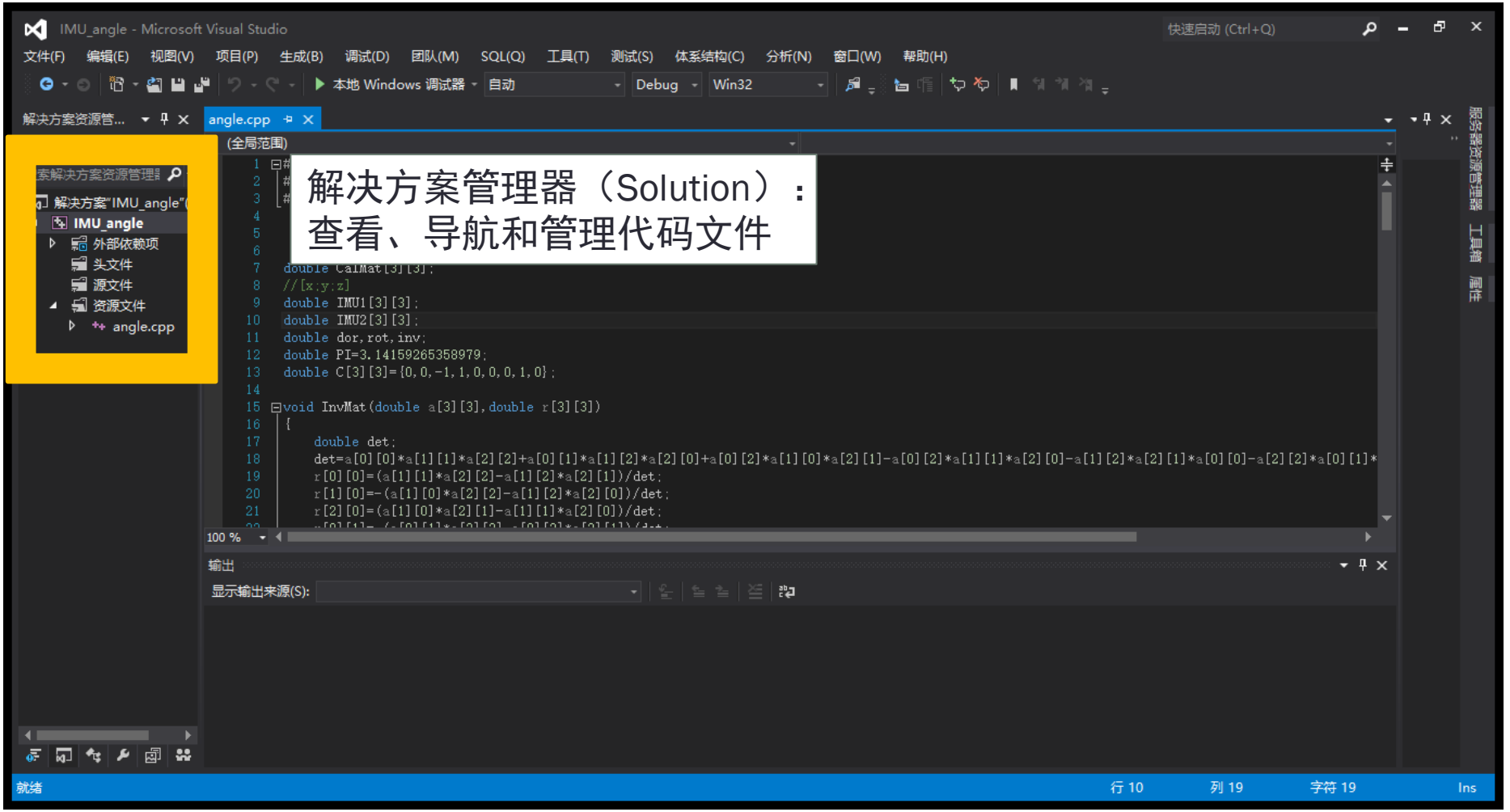
**Algorithm 1** Euclids algorithm

```
1: procedure EUCLID(a, b)                          ▷ The g.c.d. of a and b
2:     r ← a mod b
3:     while r ≠ 0 do                              ▷ We have the answer if r is 0
4:         a ← Σ_{i=1}^{n} x_i                     ▷ Nonsense to show that tall lines might work
5:         a ← b
6:         b ← r
7:         r ← a mod b
8:     end while
9:     return b                                    ▷ The gcd is b
10: end procedure
```



# 编写程序：什么是IDE

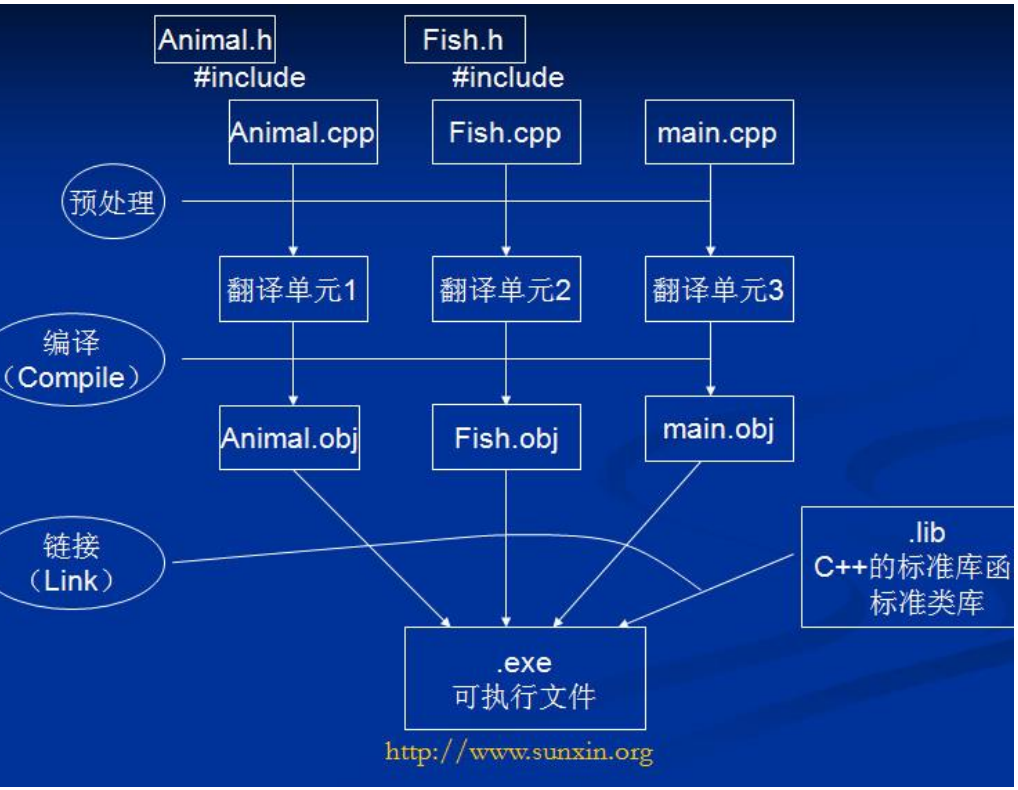
- IDE：集成开发环境（Integrated Development Environment）
- 用于提供程序开发环境的应用程序，一般包括代码编辑器、编译器、调试器和图形用户界面等工具。集成了代码编写功能、分析功能、编译功能、调试功能等一体化的开发软件服务套。
- 常用的IDE：Visual Studio、Visual C++ Express、Dev C++

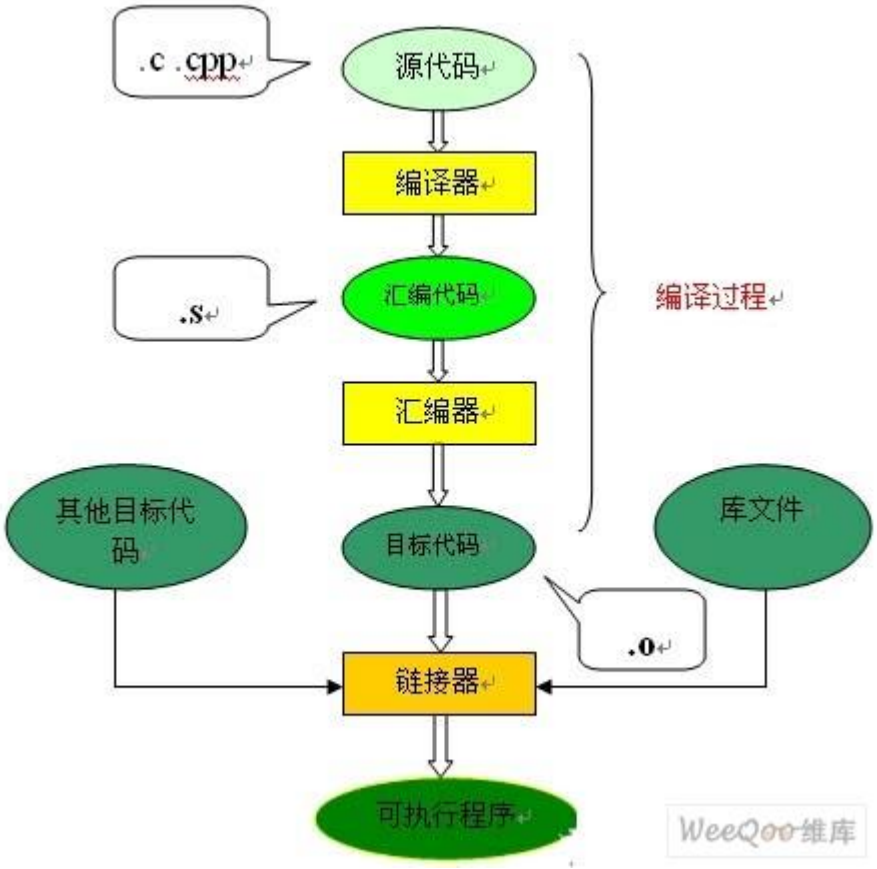

IMU_angle - Microsoft Visual Studio
解决方案管理器（Solution）：
查看、导航和管理代码文件
IMU_angle
外部依赖项
头文件
源文件
资源文件
angle.cpp
double CalMat[3][3];
//[x;y;z]
double IMU1[3][3];
double IMU2[3][3];
double dor,rot,inv;
double PI=3.14159265358979;
double C[3][3]={0,0,-1,1,0,0,0,1,0};
void InvMat(double a[3][3],double r[3][3])
{
double det;
det=a[0][0]*a[1][1]*a[2][2]+a[0][1]*a[1][2]*a[2][0]+a[0][2]*a[1][0]*a[2][1]-a[0][2]*a[1][1]*a[2][0]-a[1][2]*a[2][1]*a[0][0]-a[2][2]*a[0][1]*
r[0][0]=(a[1][1]*a[2][2]-a[1][2]*a[2][1])/det;
r[1][0]=-(a[1][0]*a[2][2]-a[1][2]*a[2][0])/det;
r[2][0]=(a[1][0]*a[2][1]-a[1][1]*a[2][0])/det;
就绪
行 10
列 19
字符 19
Ins

# 头文件(header)与源文件(source)

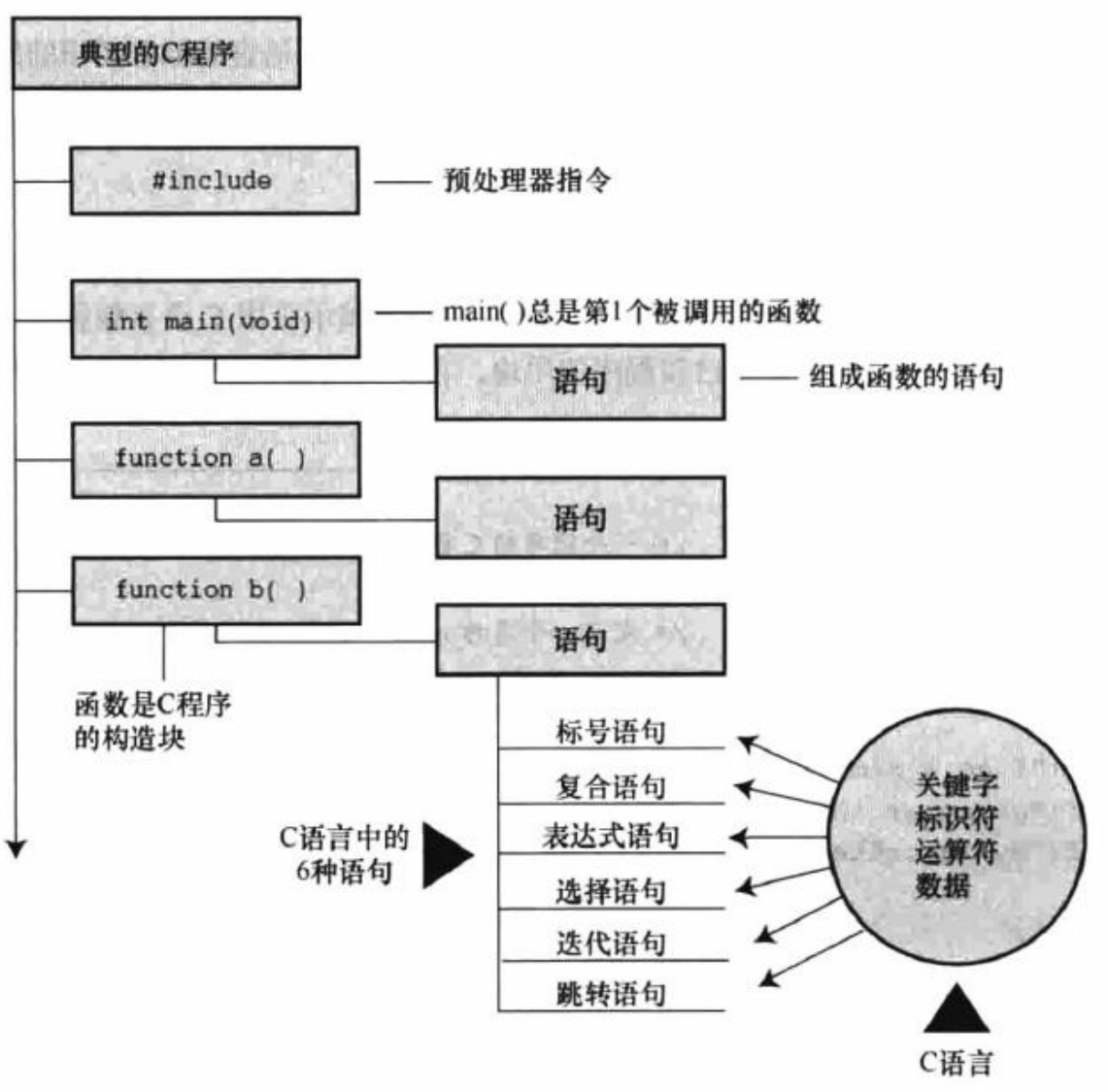
- 头文件：扩展名为 .h 的文件
  - 两种类型：程序员编写的头文件和编译器自带的头文件
  - 引用头文件相当于复制头文件的内容
  - 建议把所有的常量、系统全局变量和函数原型写在头文件中，在需要的时候随时引用这些头文件。
  - 引用语法：
    - #include <file> ：引用编译器自带的头文件
    - #include “file“：引用自己编写的头文件
- 源文件：拓展名为.c的文件
  - 存储函数的定义

# 编写程序：编程机制


Animal.h
#include
Fish.h
#include
Animal.cpp
Fish.cpp
main.cpp
预处理
翻译单元1
翻译单元2
翻译单元3
编译
（Compile）
Animal.obj
Fish.obj
main.obj
链接
（Link）
.lib
C++的标准库函
标准类库
.exe
可执行文件
http://www.sunxin.org


.c .cpp
源代码
编译器
.s
汇编代码
汇编器
编译过程
其他目标代码
目标代码
库文件
.o
链接器
可执行程序
WeeQoo维库

# 编写程序： C程序结构


典型的C程序
#include
预处理器指令
int main(void)
main( )总是第1个被调用的函数
语句
组成函数的语句
function a( )
语句
function b( )
语句
函数是C程序
的构造块
C语言中的
6种语句
标号语句
复合语句
表达式语句
选择语句
迭代语句
跳转语句
关键字
标识符
运算符
数据
C语言

# 程序示例

```
#include <stdio.h>
int main(void)                          /* 一个简单的C程序 */
{
    int num;                            /* 定义一个名为num的变量 */
    num = 1;                            /* 为num赋一个值 */

    printf("I am a simple ");           /* 使用printf()函数 */
    printf("computer.\n");
    printf("My favorite number is %d because it is first.\n",num);

    return 0;
}
```

# 程序细节

- #include <stdio.h>
  - 预处理指令，相当于拷贝“stdio.h”里的内容
- int main ()
  - 函数
  - C程序一定从main()函数（主函数）开始执行
  - int: main函数会返回一个整数给操作系统（程序最后return 0;）
  - 你也可以写成 void main()，这样就不需要写return 0；
- {}
  - 表示函数体/块
- int num;
  - 声明，函数中有一个叫做num的变量；这个变量为int类型。
  - 编译器会在内存中预留可以存放一个int的空间

# 程序细节

- num = 1;
  - 赋值：把变量1赋值给num
  - 计算机把“1”存放在了num的位置
- printf(“I am a simple “);
  - printf(): C语言中的标准函数，在stdio库内
  - #include <stdio.h>: 让编译器能够找到printf()
  - “I am a simple “: 字符串，printf函数的参数

# 编写程序：良好的编程习惯

- 什么是代码规范：
  - 命名：体现变量、函数的含义
    - 下划线命名法： print_employee_paychecks()
    - 驼峰命名法：printEmployeePaychecks()
    - 帕斯卡命名法：PrintEmployeePaychecks()
    - 约定：常量全部大写，变量小写，单字符一般用在局部变量/循环
    - 可以变量用驼峰命名法，函数用帕斯卡命名法
  - 换行与缩进：看清楚循环、选择等程序结构；括号要对齐！

因代码不规范，码农枪击4名同事，一人情况危急

代码规范很重要

```
void TimeMat(double a[3][3],double b[3][3],double r[3][3])
{
    for(int i=0;i<3;i++)
        for(int j=0;j<3;j++)
        {
            r[i][j]=0;
            for(int p=0;p<3;p++)
                r[i][j]+=a[i][p]*b[p][j];
        }
}
```

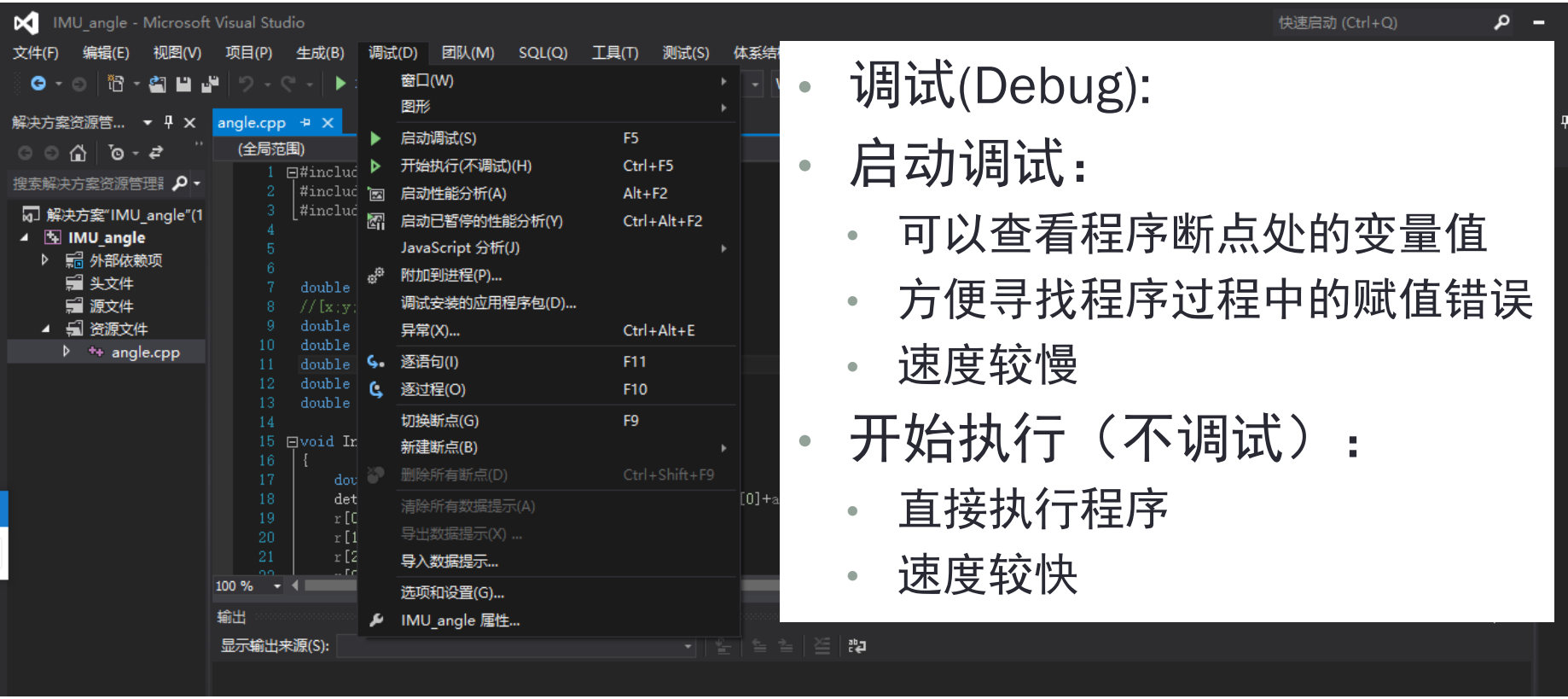
# 要写注释：
# 写代码，不是自己开心就可以的！

- 开头：
  - 作者姓名、单位、程序描述、日期
- 中间：
  - 函数作用及输入输出格式
  - 重要步骤的说明
  - 变量含义
- 进阶玩法：技术文档
  - 设计思路、参考文献、模块作用.....

- 建议：用英语写注释方便移植/交流

```
import ...

# update 7/26/2018:
# intent 22 (finding) is discarded; end of each intend is ignored;
# add grid search
# update 8/3/2018:
# new features & visualization (withlabel_v2, glove_viz_v2)
# update 8/6/2018:
# save model


num_sensors = 17
num_features = 20


# load data from all the trials
def load_trials(ID, hand, feature=True):...

# pca
def get_pca(data, n_features, feature=True):
    exp_info = data[:, 0:2]
    trial_info = data[:, data.shape[1]-2:data.shape[1]-1]
    # info[:, 0]: obj number  (0: cylinder, 1: box, 2: disk, 3: ball)
    # info[:, 1]: intend number (20: explore, 21: excavate, 22: find, 23: retrieve)
```

# 测试、调试程序




- 调试(Debug):
- 启动调试：
  - 可以查看程序断点处的变量值
  - 方便寻找程序过程中的赋值错误
  - 速度较慢
- 开始执行（不调试）：
  - 直接执行程序
  - 速度较快

断点：
让程序在需要的地方中断

# 测试、调试程序

- 假如编译不通过：
  - 按顺序查看错误说明（会显示在哪一行出现什么错误）并修改
  - 建议改一条就再编译一次
  - 遇到看不懂的说明就去百度/谷歌，总有人会和你错的一样的
- 假如编译通过了但结果不对：
  - 用断点调试（推荐）或print重要变量等方法检查程序运行过程中变量的值是否有误
  - 通常是输入、赋值的问题
  - 如果是程序结构的问题，可以用手在草稿纸上算一算

# 作业预备知识及提示

# 常量：不能改变的量

- 整型常量：0,100,1000
- 实型常量：
  - 十进制小数：0.123
  - 指数：12.34e3（代表12.34*10^3）
  - e或E代表以10为底的指数；e或E前必须有数字，后面必须为整数。
- 字符常量：用单引号表示
  - ‘A’, ‘o’,...
  - 转义字符：’\n’, ‘\t’, ...
- 字符串常量：用双引号表示
  - “Hello, world!\n”
- 符号常量：#define PI 3.14156

# 变量：可以改变的量

- 先声明，后使用！
- 变量声明要在函数块的顶部
- 常用数据类型：
  - 整型：int, long, short, unsigned
  - 字符型：char
  - 浮点型：float, double

- 2.0是整型还是浮点型？

# 简单输入输出

- printf(格式控制，输出表列);
  - printf("I want %d apples and %f liter of water. \n", num, volume);
- 格式控制：格式声明+普通字符

| 转换说明 | 输出类型 |
| --- | --- |
| %d | 十进制整数 |
| %s | 字符串 |
| %c | 字符 |
| %f | 浮点数（float类型） |
| %lf | 浮点数（double类型） |

%m.nf:
m：数据宽度
n：小数位数
%-m.nf:
-：左对齐

# 简单输入输出

- scanf(格式控制，地址表列)
  - scanf(“I want %d apples and %f liter of water.”, &num, &volume)

- &：取地址符
- 一定注意scanf后面要用变量地址而不是变量名！
- 输入格式要和格式控制严格相同！
  - scanf(“a=%f,b=%f,c=%f”,&a,&b,&c)
  - 输入a=1,b=3,c=2（回车）√
  - 输入1,2,3（回车）×
  - 输入a=1 b=2 c=3（回车）×

# 常见编程问题

- 为什么程序读不了我输入的数？
  - 检查你输入格式和你预设的格式是否完全相同
  - 尤其注意中英文标点符号、换行和空格
- 为什么调试总是失败？
  - 检查括号数量是不是对上了
  - 检查分号有没有打对（不要打成中文符号，不要打成冒号）
  - 检查你前后变量名有没有打错
- 其他常见问题：
  - 赋值（=）和判断等于（==）
  - 循环
  - 选择