

实践8小结

- 计算思维的本质是抽象和自动化！
- 重名问题：
 - 函数名也是一种标识符，函数名与变量名不要重复；
 - 全局变量与局部变量重名时，在局部域中全局变量被屏蔽，需要使用全局域运算符::访问。

实践8小结

- 函数的基本属性

- 声明
- 定义
- 调用

熟练掌握！！

- 函数的高级属性

- 重载
- 默认参数
- 内联

- 静态局部变量：

- 局部作用域
- 全局生命期

实践8小结

- 若定义重载函数

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

`float max(float x, float y);`

- 若调用时使用 `max(1.2, 3.4);` 将报错，编译器无法匹配到合适函数。
 - 可以定义 `float a=1.2, b=3.4;` 调用`max(a,b);`
 - 或者强制类型转换，调用`max((float)1.2, (float)3.4)。`
- 常量：整数默认为int型，小数默认为double型

实践8小结

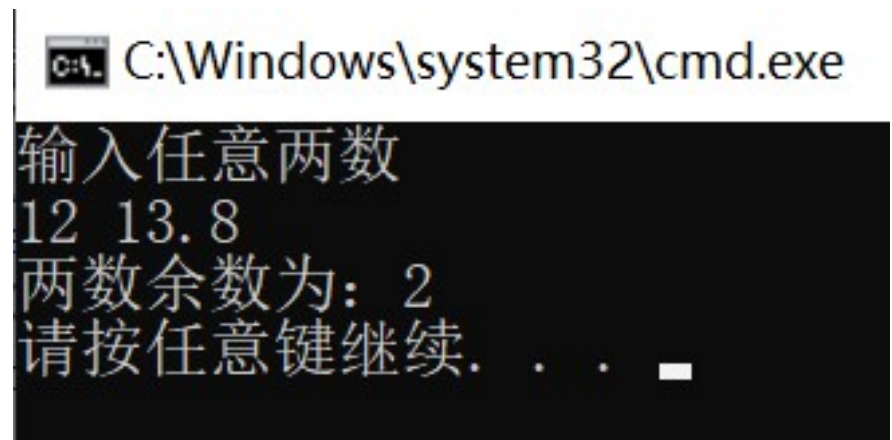
作业中有问答题：当输入参数，一个是整型，一个是实型时，产生什么情况？分析原因。

```
double a,b;
```

```
//也有定义为int a,b;
```

```
cin >> a >> b;
```

```
remainder(a, b);
```



```
C:\Windows\system32\cmd.exe
输入任意两数
12 13.8
两数余数为: 2
请按任意键继续. . .
```

注意：算术表达式 $a \% b$ 和 a / b ，都要求 b 非 0.

```

cout<<"向上取整： ";
cout<<( a>0 ? (int)a+1 : (int)a )<<endl;
cout<<"向下取整： ";
cout<<( a>0 ? (int)a : (int)a-1 )<<endl;
cout<<"四舍五入： ";
cout<<( a>0 ? (int)(a+0.5) : (int)(a-0.5) )<<endl;
cout<<"四舍五入： ";
cout<<( a>0 ? floor(a+0.5) : ceil(a-0.5) )<<endl;

```

		3.1	3.9	-3.1	-3.9
floor(x)	不大于x的最大整数	3	3	-4	-4
ceil(x)	不小于x的最大整数	4	4	-3	-3
round(x)	最邻近x的整数	3	4	-3	-4

实践8小结

- 任何对象的使用原则——“**可见性**”
 - 宏定义，默认参数（默认值在函数声明时给出）。
 - 自定义数据类型，变量，函数。
 - **头文件**包含，同一标识符的**定义有且仅可有一次**，可以声明多次。
 - 外部变量声明，如： **extern int a;**
 - 函数声明

实践8小结

- 编译器搜索头文件的顺序
 - <>从系统目录下开始搜索，然后再搜索PATH环境变量所列出的目录，不搜索当前目录。
 - “”从当前工程目录开始搜索，然后搜索系统目录和PATH环境变量所列出的目录。
 - 双引号引起字符串常量，其中windows路径分隔符\需要转义，例：`#include "D:\\MyProg\\area.h"`
 - 建议将文件放在当前工程目录下，采用默认路径包含文件，即：`#include "area.h"`
 - 简言之，include命令相当于把包含的文件的代码，放在相应的位置上。

实践8小结

- 多文件工程的意义！
- 多文件工程开发习惯
 - .h 头文件放声明，如：AreaFun.h
 - .cpp 源文件放定义（实现），如：AreaFun.cpp
 - 一个标识符（变量，函数），在整个程序中可以被声明多次，但却要且仅要被定义一次。

可利用预编译命令防止重复包含

```
#ifndef _AREAFUN_H_
#define _AREAFUN_H_
..... 头文件内容
#endif
```


实践9 递归函数

- 实验内容：
 - 函数的嵌套调用
 - 函数的递归调用

注意：本次实验不允许使用全局变量传参或返回。

实践9：

- 掌握函数的嵌套调用和递归调用。
 - 嵌套调用为常规的他引调用。
 - 递归调用为函数的自引调用，递归函数必须包含使递归终止的语句，即避免“死循环”。
- 掌握递归算法的实现，理解递归函数的展开。
- 实践内容：实验八，课本习题3.9~3.10。

本次作业有难度，但属于必须掌握内容。

若实现有困难，可选择自己能理解的2题，自行完成，务必自己写，自己调试，递归函数断点调试难度大，可减小测试规模，建议用cout输出中间结果，能更好地观察、体会递归函数展开过程。

实践9：提交程序清单1

实验八-3，改进求组合数的函数，函数原型为：

```
long comImprove(int m, int n);
```

要求1：求阶乘函数和改进的求组合函数都采用递归函数实现，注意：递归终止条件。

要求2：观察n取值对求阶乘函数和改进的求组合函数的影响。

提高：若递归终止条件错误，造成了“死循环”，观察堆栈空间溢出现象。

其他. 1.若有输入，设计简洁有效的输入提示。

2. main()函数中自行设计函数的测试代码。

3. 将测试结果用注释列举在文件顶部。

实践9：提交程序清单2

课本习题3.9~3.10，递归实现多分支函数。

要求：用函数实现自己的表头，程序在控制台输出表头，计算自己做的题号： $(\text{学号后二位} - 1) \% 2 + 9$

要求：继续体会递归终止条件的实现。即函数一定会采用分支结构来处理不同情况，并且在这些不同情况中，一定有一个或多个基本的或最简单的情况，用于停止递归。

其他. 1.若有输入，设计简洁有效的输入提示。

2. `main()`函数中自行设计函数的测试代码。

3. 将测试结果用注释列举在文件顶部。

课本习题3.9和3.10的测试范例:

Ackman函数

调用次数 函数值

$$1 \quad Ackman(0,0) = 1$$

$$1 \quad Ackman(0,1) = 2$$

$$2 \quad Ackman(1,0) = 2$$

$$6 \quad Ackman(1,2) = 4$$

$$14 \quad Ackman(2,1) = 5$$

$$44 \quad Ackman(2,3) = 9$$

$$541 \quad Ackman(3,2) = 29$$

$$2432 \quad Ackman(3,3) = 61$$

$$10307 \quad Ackman(3,4) = 125$$

Legendre函数

调用次数 函数值

$$1 \quad P(0,0) = 1$$

$$1 \quad P(1,0.1) = 0.1$$

$$3 \quad P(2,0.1) = -0.485$$

$$3 \quad P(2,0.5) = -0.125$$

$$3 \quad P(2,1.5) = 2.875$$

$$5 \quad P(3,1.5) = 6.1875$$

$$9 \quad P(4,1.5) = 14.0859$$

实践9：提交程序清单3

课本习题3.4，**最大公约数**通用函数的**递归**实现。

要求：体会迭代法的递归函数实现。递归的本质也是反复执行，但并非循环，无需使用循环控制结构。

要求：继续体会递归终止条件的实现。

其他. 1.若有输入，设计简洁有效的输入提示。

2. `main()`函数中自行设计函数的测试代码。

3. 将测试结果用注释列举在文件顶部。

实践9：提交程序清单4

用**迭代**、**递归**两种方法分别实现：

- 1) 十进制数转换为二进制数输出(只要输出不需用数组存储，数组的练习后续实验中会反复加强)
- 2) 斐波拉契数列(即例3.15，主要目的：体会展开过程)

要求：体会**迭代法**的**递归法**的转换。

提示：建议在递归函数内直接输出每层调用的结果，更好地体会**递归函数调用时的展开过程**。

注意体会**输出语句与递归调用函数之间的前后顺序**对输出的影响。

其他. 1.若有输入，设计简洁有效的输入提示。

2. `main()`函数中自行设计函数的测试代码。

3. 将测试结果用注释列举在文件顶部。

实践9：

1、课本习题3.15，几例递归函数。

自行读题求解后，实验验证，程序不提交。

理解递归函数展开过程！

2、课本范例实现一个整数的逆序输出。

例2.15，采用数组和循环结构，

例3.14，采用递归算法。

自行通过递归函数体中输出语句的位置对输出效果的影响，理解递归函数展开过程！范例程序在QQ群下载。

重要！！！！

提交的文件名格式：

Exp09_学号_实验名.cpp

注意:1) 下划线

2) 学号别漏掉

3) 各种名字(变量名, 文件名)的可读性!

4) 每个实验只要提交一个程序源文件, 即
.cpp文件。

附加题也用**Exp09_学号_实验名.cpp**格式提交!