C语言编程规范总则

1. **文件：**

文件（源文件.c和头文件.h）命名要符合简洁、通俗易懂的规则，尽量使用能表达出该文件的作用的英文来命名，不得使用汉语拼音，采用小驼峰法，即第一个单词首字母小写，其余单词首字母大写。

1. **头文件：**

(1)引用头文件时，标准库中的头文件使用<>，而自定义的头文件使用“”；

Eg. #include <math.h>

#include “hello.h”

(2)为了防止头文件被重复引用，应当采用#ifndef/#define/#endif结构产生预处理。

Eg. #ifndef \_\_HEADER\_H\_ /\* 文件名前面加两个下划线“\_”,后面加“\_H\_” \*/

#define \_\_HEADER\_H\_

//……

#endif

1. **函数命名:**

函数命名要规范，符合简洁、通俗易懂的原则，尽量使用能表达该函数功能的英文来命名函数，不得使用汉语拼音，一般使用动词或者动词与名词混合的形式，一个单词时首字母需要大写；当超过两个单词时，采用帕斯卡命名法，即每个单词的首字母大写。

Eg. void Sum();

int ImageProcess();

1. **常量与变量命名规范：**

常量和变量的定义要简洁、通俗易懂，尽量使用能表达该量所代表的含义的英文来命名，不得使用汉语拼音，一般使用名词，通常来说，常量使用全大写字母来表达，如果该常量的名称中有两个英文单词，则使用下划线进行隔开；变量使用首首字母小写来表达，当超过两个单词时，采用小驼峰命名法，第一个单词的首字母小写，其余单词的首字母需要大写。

Eg. #define PI 3.1415926

#define GRAVITY­\_ACCELERATION 9.8

int speed;

double imageSize;

一些常用的缩写例子：

argument 可缩写为 arg

buffer 可缩写为 buff

clock 可缩写为 clk

command 可缩写为 cmd

compare 可缩写为 cmp

configuration 可缩写为 cfg

device 可缩写为 dev

error 可缩写为 err

hexadecimal 可缩写为 hex

increment 可缩写为 inc、

initialize 可缩写为 init

message 可缩写为 msg

parameter 可缩写为 para

previous 可缩写为 prev

register 可缩写为 reg

semaphore 可缩写为 sem

statistic 可缩写为 stat

synchronize 可缩写为 sync

temp 可缩写为 tmp

1. **作用域：**

在程序中尽量使用常量，如果要使用变量，则其作用域要尽量小，而且尽量使用局部变量，而不使用全局变量。

1. **书写规范：**

要遵循美观舒适的原则，运算符左右要使用空格，花括号内的语句要使用Tab键进行缩进，花括号要尽量缩进相同的距离，以便查找上下是否相对应。

Eg. double Sum(double addendA, double addendB)

{

double sumC;

sumC = addendA + addendB;

return sumC;

}

1. **排版：**

为了增加程序的可读性，需要使用语义空行，使程序有效分割。

1. **长度：**

每个函数的长度尽量不要太长，从而增加可读性（一般控制在30行左右），每一行的代码尽量不要太长，可以适当换行，从而使整个函数尽量呈矩形，以增加美感。

1. **耦合度：**

函数与函数之间要尽量降低耦合度，最好是去耦合，要做到低耦合高内聚，即函数的功能单一化。

1. **复用度：**

当多个函数共用同一段代码时可以将这一段代码单独写成一个函数，从而实现代码复用。

1. **注释：**

为了增加程序的可读性，注释必不可少，函数头和每个功能函数的注释参考我已写好的模板，在每个函数中尽量做到每一行都有注释。

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*文件名称：加法器 \*

\*作 者：WC \*

\*版 本：1.0.0 \*

\*日 期：2018-04-03 \*

\*功能描述：本程序为计算两个数的和。 \*

\*输 入：该程序为多输入，输入为两个数。 \*

\*输 出：该程序为单输出，输出为两个数的和。 \*

\*函数列表：1.Sum()——用于计算两个数的和。 \*

\*历 史： \*

\* Copyright (C) IIPCV Lab. All Rights Reserved \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <math.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*描述：Sum()函数用于计算两数和，输入为加数addendA和被加数addendB，输出为两数和sumC。 \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

double Sum(double addendA, double addendB)

{

/\*定义两个数之和sumC\*/

double sumC;

/\*进行加法计算\*/

sumC = addendA + addendB;

/\*返回结果\*/

return sumC;

}

1. **异常处理机制：**

要考虑异常处理机制，定义状态值，使用if-else进行判断。