Variable:

int this\_is\_a\_variable\_exampe = 8;

Note: Never use a variable name such as i, x, n unless it is used for a loop

Macro and Constant:

#define THIS\_IS\_A\_MACRO\_EXAMPLE(\_a, \_b, \_c) do { \

\_a = 0; \

\_b = 0; \

}while(0)

#define THIS\_IS\_ANOTHER\_MACRO\_EXAMPLE 0XFF //less preferred

static const int LOCAL\_PORT = 15501; //preferred

Note: Never use a magic number directly unless it is used as bit operation (but still recommended to use MACRO)

Enum:

typedef enum {

THIS\_IS\_A\_ENUM\_CASE,

THIS\_IS\_ANOTHER\_ENUM\_CASE

} this\_is\_a\_enum\_example\_t;

Note: Use enum instead of macro if more than one macro is defined in a series

Structure:

typedef struct {

int int\_example;

} this\_is\_a\_structure\_example\_t;

Function or Method:

//Description: this is the function to

//Arguments:

//Return Values:

int this\_is\_a\_function\_example(int a, int b)

Note: At least the description for each function is required, explanation for arguments and return values are highly recommended.

Class and Object:

// Description: this is the class to

class ThisIsClassExample {

….

private:

int \_this\_is\_a\_internal\_variable\_example;

};

ThisIsClassExmaple thisIsClassExample;

Note: The description for each class is required

File:

Each cpp file contains ONLY one class and has the same name as the class.

Each h file, if it is corresponding to a cpp file, must use the same name as the cpp file.

ThisIsHFileExample.h

#ifndef \_THIS\_IS\_H\_FILE\_EXAMPLE\_H\_

#define THIS\_IS\_H\_FILE\_EXAMPLE\_H\_

….

#endif

Space and New line:

if( result == True ){ //Note: One space after (, around == and before ), new line after {

printf…. //Note: Every four spaces for one shift

}

while( result == True ){ //Note: One space after (, around == and before ), new line after {

printf…. //Note: Every four spaces for one shift

}

switch( result ){ //Note: One space after ( and before ), new line after {

case 1: { //Note: One space after :, new line after {

…..

break;

} //Note: New line after }

default: { //Note: One space after :, new line after {

….

break; //Note: New line after }

}

}

Note: One space around all operations such as +, -, \* and /

Note: No trailing spaces for any time

Vector and Array:

Note: Vector is always the first try

Comments:

Note: Always use // instead of /\*\*/