PREPROCESSING : commands

PREPROCESSOR COMMANDS

#define Define a preprocessor macro.

Usage: #define MAX 256
#define ERROR_STR "\pError, try again."
#define MAX(A,B) ((A)>(B) ? (A) : (B))

#undef Remove a preprocessor macro definition.

Usage: #undef MAX

#include Insert text from another source file.

Usage: #include <file> ->searches 'standard' locations
#include "file" ->searches 'local' locations

#if Conditionally include some text, based on the value of a constant expression.

Usage: #if MAX > 256
#include "altSize.h"
#endif

#ifdef Conditionally include some text, based on whether a macro name is defined.

#ifndef Conditionally include some text, based on whether a macro name is not defined.

Usage: #ifndef MAX
#define MAX 256
#endif

#else Alternatively include some text, if the previous #if, #ifdef, #ifndef, or #elif test failed.

#endif Terminate conditional text.

#line Supply a line number for compiler messages.

#elif Alternatively include some text based on the value of another constant expression, if the previous #if, #ifdef, #ifndef, or #elif test failed.

Usage: #ifndef OS
#define OS MAC
#endif

#if OS == MAC
#define SYS_HEADER "MacOS.h"
#elif OS == WIN95
#define SYS_HEADER "Win95.h"
#elif OS == UNIX
#define SYS_HEADER "Unix.h"
#endif

defined Preprocessor function that yields 1 if a name is defined

as a preprocessor macro and 0 otherwise; used in #if and #elif statements.

Usage: #if defined TOKEN -or- #if defined(TOKEN)

unary # Operator to replace macro parameter with a string constant containing the parameter's value.

Usage: #define PRINT(a) printf("value = " #a "\n")

'PRINT(5);' becomes 'printf("value = 5\n");'

binary ## Operator to create a single token out of two adjacent tokens.

Usage: #define INPUT(i) input ## i

'INPUT(1) = INPUT(2);' becomes 'temp1 = temp2;'

#pragma Specify implementation-dependent information to the compiler.

#error Produce a compile-time error with a designated message.

Usage: #error "Opps! MAX not Defined."