Introduction to C

Preprocessor

Instructor: Yin Lou

02/09/2011

Preprocessor

- Commands to the compiler
- Include files, shortcuts, conditional compilation
- Command must start at beginning of line

Preprocessor

- Commands to the compiler
- Include files, shortcuts, conditional compilation
- Command must start at beginning of line

Common Preprocessor Commands

```
#include
#define
#ifdef / #ifndef
```

Running just the preprocessor

```
$ gcc -E main.c -o preprocessed.c
```

#include: Header Files

- Includes files: Literally copy-paste

#include: Header Files

- Includes files: Literally copy-paste
- Typically header files

#include: Header Files

- Includes files: Literally copy-paste
- Typically header files

Header Files

Declares

- External functions
- Variable types
- External global variables

Typically named *.h

#include: Header Files

mylib.h

```
int max(int a, int b);
```

mylib.c

```
#include "mylib.h"
int max(int a, int b)
{
    return (a > b ? a : b);
}
```

#include: Header Files

project.c

```
#include "mylib.h"

void foo()
{
    ...
    m = max(p, q);
    ...
}
```

#include: Header Files

project.c

```
#include "mylib.h"

void foo()
{
    ...
    m = max(p, q);
    ...
}
```

gcc -o project project.c mylib.c

#define: Macros

Blind substitution inside file

```
#define malloc mymalloc
#define maxsize 100

p = malloc(maxsize);
printf("Allocated %d bytes", maxsize);
```

#define: Macros

Blind substitution inside file

```
#define malloc mymalloc
#define maxsize 100

p = malloc(maxsize);
printf("Allocated %d bytes", maxsize);
```

is exactly the same as

```
p = mymalloc(100);
printf("Allocated %d bytes", 100);
```

#ifdef: Conditional compilation

project.c

```
#ifdef DEBUG
#include "mylib.h"
#define malloc mymalloc
#endif
...
p = malloc(100);
```

#ifdef: Conditional compilation

project.c

```
#ifdef DEBUG
#include "mylib.h"
#define malloc mymalloc
#endif
...
p = malloc(100);
```

For debugging: gcc -DDEBUG -o project project.c mylib.c
For release: gcc -o project project.c mylib.c

#ifdef: Conditional compilation

mylib.h

```
void *mymalloc(int size);
void myfree(void *ptr);
```

#ifdef: Conditional compilation

mylib.c

```
#include <stdio.h>
#include <stdlib.h>
#include "mylib.h"

void *mymalloc(int size)
{
    void *ret = malloc(size);
    fprintf(stderr, "Allocating: %d at %p\n", size, ret);
    return ret;
}

void myfree(void *ptr)
{
    fprintf(stderr, "Freeing: %p\n", ptr);
    free(ptr);
}
```

#include: Problems

mylib1.h

```
#include "mylib2.h"
```

mylib2.h

```
#include "mylib1.h"
```

#include: Solution

mylib1.h

```
#ifndef __MYLIB1_H
#define __MYLIB1_H
#include "mylib2.h"
#endif
```

mylib2.h

```
#ifndef __MYLIB2_H
#define __MYLIB2_H
#include "mylib1.h"
#endif
```

#define: More useage

project.c

```
#define prod(a, b) prod2(a, b * 10)

prod(5, 6) => prod2(5, 6 * 10)
```

#define: More useage

project.c

```
#define prod(a, b) prod2(a, b * 10)

prod(5, 6) => prod2(5, 6 * 10)
```

prod(5, 6 + 7) => prod2(5, 6 + 7 * 10) BUG!!

#define: Solution

project.c

```
#define prod(a, b) (prod2((a), (b) * 10))

prod(5, 6 + 7) => (prod2((5), (6 + 7) * 10))
```

#define: More useage

project.c

```
#define oldfunc(a, b) newfunc1(a); newfunc2(b);

oldfunc(5, 6) => newfunc1(5); newfunc2(6);
```

#define: More useage

project.c

```
#define oldfunc(a, b) newfunc1(a); newfunc2(b);

oldfunc(5, 6) => newfunc1(5); newfunc2(6);
```

for (i = 0; i < 5; i++) oldfunc(5, 6); =>
for (i = 0; i < 5; i++) newfunc1(5); newfunc2(6);

#define: Solution

project.c

```
#define oldfunc(a, b) do { \
    newfunc1((a)); newfunc((b)); \
} while (0)
```

#define: More problems

project.c

```
#define max(a, b) ((a) > (b) ? (a) : (b))

max(p, q) => ((p) > (q) ? (p) : (q))
```

#define: More problems

project.c

```
#define max(a, b) ((a) > (b) ? (a) : (b))

max(p, q) => ((p) > (q) ? (p) : (q))
```

max(f1(), f2()) =>
((f1()) > (f2()) ? (f1()) : (f2()))

#define: More problems

project.c

```
#define max(a, b) ((a) > (b) ? (a) : (b))

max(p, q) => ((p) > (q) ? (p) : (q))
```

max(f1(), f2()) =>
((f1()) > (f2()) ? (f1()) : (f2()))
Solution: Be extra careful when calling a function inside code that could be a #define. Always use uppercase for macros to serve as reminder.