CMPUT 379 Fa19 - OPERATING SYSTEM CONCEPTS Combined LEC Fa19

Mixing C and C++

When you try to mix C and C++ (more precisely, compiling with both gcc and g++ and then link the object files), you may encounter *ld: symbol(s) not found* like this:

```
gcc -c threadpool.c -o threadpool.o
g++ -c mapreduce.cc -o mapreduce.o
gcc -c distwc.c -o distwc.o
gcc *.o -pthread -o mr
Undefined symbols for architecture x86_64:
  "_MR_Emit", referenced from:
      _Map in distwc.o
  "_MR_GetNext", referenced from:
      _Reduce in distwc.o
  "_MR_Run", referenced from:
      _main in distwc.o
ld: symbol(s) not found for architecture x86_64
```

This is because C++ compilers will do "name mangling" on function names, which means functions are named differently in the objects files. Therefore, the linker may not be able to locate those functions. To get around this, you may turn off name mangling for C++ codes by surrounding the declarations in C++ header files with **extern "C" {}.** These extern "C" staff should only be visible to C++ compilers, and this is achieved by **#ifdef __cplusplus**. To wrap up, your modified **mapreduce.h** should look like

```
#ifndef MAPREDUCE_H
#define MAPREDUCE_H
#ifdef __cplusplus
extern"C" {
#endif
```

```

void MR_Emit(char*key, char*value);
// other declarations omitted
```

```

#ifdef __cplusplus
}
#endif
#endif
```

An alternate way to resolve the issue is always compile using only g++ or gcc. You should have no error using gcc. For g++, you need to modify **line 18** in **distwc.c:**

MR_Emit(token, "1"); ==> MR_Emit(token, **(char *)**"1");

Feel free to use the updated headers.
https://eclass.srv.ualberta.ca/pluginfile.php/5383401/mod_page/content/5/headers.zip

Last modified: Friday, 18 October 2019, 8:23 PM

◄ Assignment 2 — rubric Jump to... Assignment 2 — Submission ►