Translation Process

1. Macro Processor: **source.cpp** is expanded by the macro processor. All macros and #define's are replaced by C++ code.
2. C++ Language Compiler: **source.cpp**, having all macro directives removed, is translated into object representation. Only identifiers that are defined elsewhere remain in plain text form.
3. Linkage Editor **source.o** is combined with d C++ libraries and other **.o** files. External references are resolved. An executable file is created

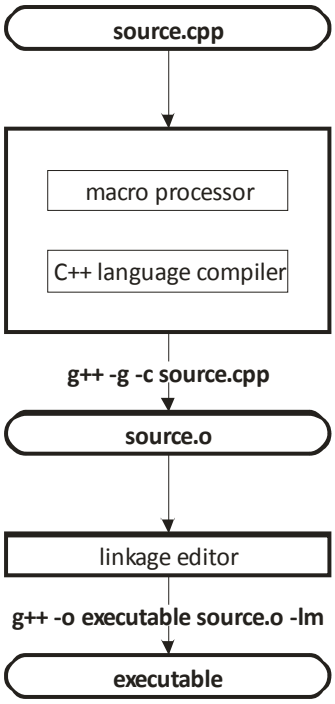

Figure 1. Translation Process

Notes:

1. **-g** option directs the compiler to include information for the source debugger, gdb
2. **-c** option directs the compiler to produce a relocatable object file. External references are not resolved
3. **-E** option directs the compiler to stop after invoking the macro processor. To view the result of the macro processor phase:
   $ **g++ -E source.cpp > source.m**
4. **-o** option directs the linker to assign the name following the option to the executable file produced. For example,
   $ **g++ -o p01 p01.o list01.o**
   directs the linker to name the executable file **p01**.
5. The linker (linkage editor) is invoked when all the input files have a .o suffice - when all the input files are relocatable objects.

Programs consisting of multiple source files

1. Compile all source files.
   a. **$g++ -c -g p01.cpp**
   b. **$g++ -c -g List01.cpp**
2. Link all objects
   a. **$g++ -o p01 p01.o List01.o -lm**

Function prototypes

1. Inform the compiler how to call a function.
2. Inform the compiler functions may be defined elsewhere
3. Validate function prototypes against actual function definitions

File organization.

1. File description comment
2. Author identification comment
3. Standard C++ Libraries
4. Application includes
5. Macro definitions
6. Class definitions
7. File global data
8. File functions

Include Files

1. The include file contains a class that defines the abstract data type.
2. The **.cpp** file contains the implementation of member functions in the class.
3. Directs compiler how to call functions, number and type of parameters and return type

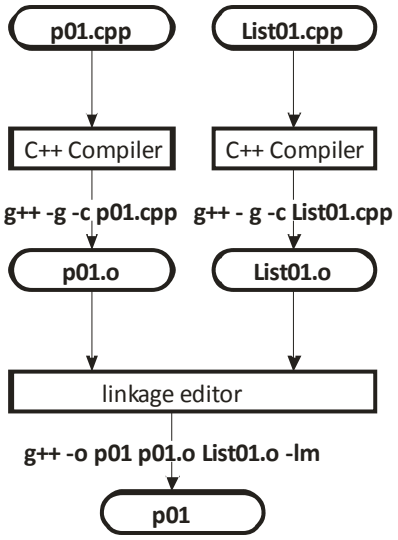

Figure 2. Project 1 Translation

Makefiles

1. Form
   target file: source files
   instructions
2. File **p01make** contains
   p01: p01.o List01.o
   g++ -o p01 p01.o List01.o -lm
   p01.o: p01.cpp List01.h
   g++ -c -g p01.cpp
   List01.o: List01.cpp List01.h
   g++ -c -g List01.cpp
3. Invoking makefiles
   $ **make -f p01make**

```
#-------------------------------------------------------------------------------------------------------
# File p01make contains instructions for creating executable file p01.  Executable file
# p01 is the program for project p01, CS2613, Programming II.
#-------------------------------------------------------------------------------------------------------
# Author:       Ms. Petunia Perfect
# Student ID:   *00000000
# E-Mail:       pperfect@uco.edu
# Course:       CS2613, Programming II
# CRN:          10847, Autumn, 2003
# Project:      p01
# Due:          September 9, 2003
# Account:      tt000
#-------------------------------------------------------------------------------------------------------
# Object files
#-------------------------------------------------------------------------------------------------------
obj     =       p01.o List01.o
#-------------------------------------------------------------------------------------------------------
# Link object files into executable file p01
#-------------------------------------------------------------------------------------------------------
p01:            ${obj}
                g++ -o p01 ${obj} –lm
#-------------------------------------------------------------------------------------------------------
# Compile p01.cpp that exercises class List
#-------------------------------------------------------------------------------------------------------
p01.o:          p01.cpp List01.h
                g++ -g –c p01.cpp
#-------------------------------------------------------------------------------------------------------
# Compile List.cpp that implements class List
#-------------------------------------------------------------------------------------------------------
List01.o:       List01.cpp List01.h
                g++ -g –c List01.cpp
```

Figure 3. File **p01make**

File **p01make** notes:

1. Lines beginning with a # sign are comments

2. **One or more UNIX tabs begin lines that are indented. There is no substitute for an UNIX tab. Code a UNIX tab using an editor on the department computer. A tab in a PC or Windows editor does not translate reliably to an UNIX tab.**