



linker

Object File

symbol

linker

Executable Object  
Files

#include0|해|

template

TMP

constexpr(c++11)

## ① linker

Object File

symbol

linker

Executable Object Files

## ② #include0|해|

## ③ template

## ④ TMP

## ⑤ constexpr(c++11)

- 서적 : CSAPP (7장 linker)
- 아마 시프시간에 배우는 것.

# Compiler

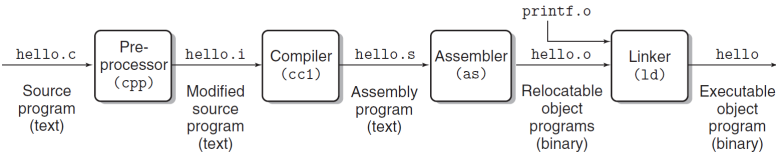


Figure: 컴파일 과정

# Object File

- Relocatable ob : compiler, assembler output
- Executable ob : linker output
- shared ob : DLL을 위한 파일 생략

# x86-linux : object 구조로 ELF(Executable and Linkable Format)사용

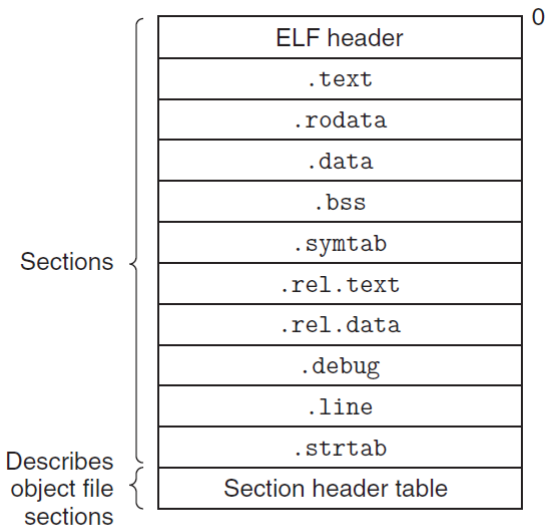


Figure: Typical ELF relocatable object file.

- **.text** The machine code of the compiled program.
- **.rodata** Read-only data such as the format strings in printf statements, and jump tables for switch statements.
- **.data** Initialized global and static C variables.
- **.bss** Uninitialized global and static C variables, along with any global or static variables that are initialized to zero. block satrted by symbol 의 약어
- **.symtab** A symbol table with information about functions and global variables that are defined and referenced in the program.
- **.rel.text** A list of locations in the .text section that will need to be modified when the linker combines this object file with others.
- **.rel.data** Relocation information for any global variables that are referenced or defined by the module.
- **.debug** A debugging symbol table with entries for local variables and typedefs defined in the program, global variables defined and referenced in the program, and the original C source file.
- **.line** A mapping between line numbers in the original C source program and machine code instructions in the .text section.
- **.strtab** A string table for the symbol tables in the .symtab and .debug sections and for the section names in the section headers.

# symbol

- Global symbols : 모듈 m에 정의되거나 다른 모듈에 참조될 수 있는 심볼 / 전역변수와 static이아닌 함수
- Global symbols : 다른 모듈에 정의된 전역변수 /다른곳에 정의된 (extern)전역변수와 static이아닌 함수
- Local symbols : static으로 선언된 함수,전역변수 참조 x



# linker 단계

- 1 symbol resolution
- 2 Relocation

# Symbol Resolution

- strong symbol : 초기화된 전역변수
- weak symbol : 초기화x 전역변수

## Rule

- ① Multiple strong symbols with the same name are not allowed.
- ② Given a strong symbol and multiple weak symbols with the same name, choose the strong symbol.
- ③ Given multiple weak symbols with the same name, choose any of the weak symbols.

# Relocation

- 1 Relocating sections and symbol definitions : 여러개의 .data section을 합친다. 이후 인스트럭션과 전역변수들이 런타임 메모리 주소를 가진다.
- 2 Relocating symbol references within sections. : 모든 심볼 참조를 수정한다. 이후 모든 심볼들이 런타임 메모리 주소를 가진다. 어셈블러가 만든 .rel.data section에 있는 Relocation Entries 자료구조를 가지고 수행한다
  - R\_X86\_64\_PC32 : 상대주소
  - R\_X86\_64\_32 : 절대주소

# Executable Object Files

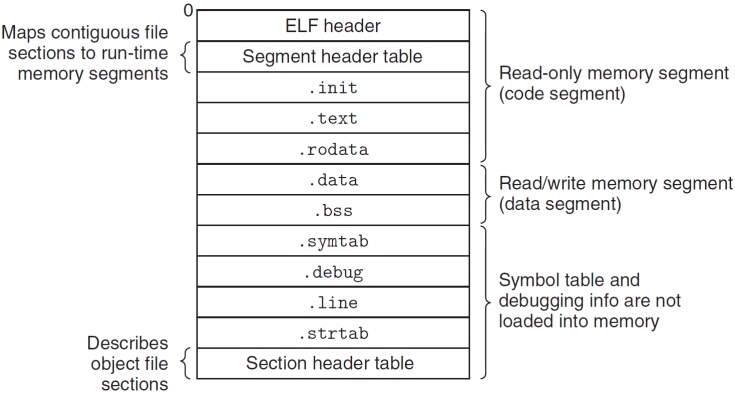


Figure: Typical ELF executable object file.

# Loading

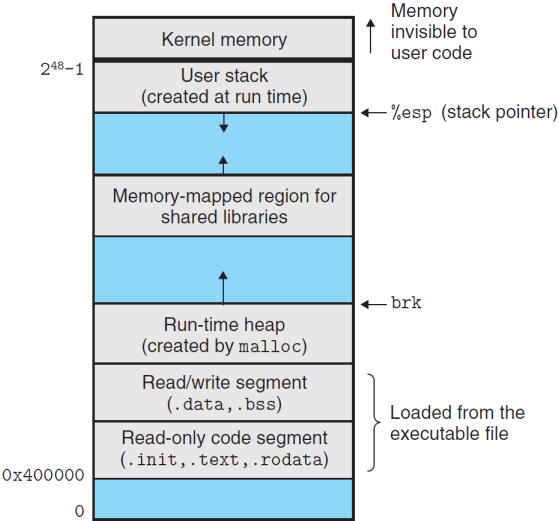


Figure: Linux x86-64 run-time memory image.

# header file

- 그냥 텍스트파일과 같다.
- 선언을 위한것...
- 실제 구현은 어느 cpp파일에서...
- `#include`는 전처리 지시자.
- `#include`로 cpp파일에 그대로 붙여넣는다.



```
1      template<typename T>
2      T function(T a , T* b, T& c)
3      {;}
4      template<
5      char function<char>(char a , char* b, char &c)
6      {;}
7
8      int main()
9      {
10         function(1,0x0000,3);
11         function("s", 0x0000,"asdf");
12     }
13
```



EUnS

linker

Object File

symbol

linker

Executable Object  
Files

`#include0|8|`

template

TMP

`constexpr(c++11)`

EUnS

linker

Object File

symbol

linker

Executable Object  
Files

`#include0|8|`

template

TMP

`constexpr(c++11)`

EUnS

linker

Object File

symbol

linker

Executable Object  
Files

`#include0|8|`

template

TMP

`constexpr(c++11)`

EUnS

linker

Object File

symbol

linker

Executable Object  
Files

`#include0|8|`

template

TMP

`constexpr(c++11)`

```
1  template <int N>
2  struct Factorial {
3      static const int result = N * Factorial<N - 1>::
   result;
4  };
5
6  template <>
7  struct Factorial<1> {
8      static const int result = 1;
9  };
10  Factorial<4>::result
11
```

- template의 특성을 이용해서 반복되는 계산을 컴파일타임에 계산을 해놓은다음 그 값을  $O(1)$ 에 부르는 흑마법
- 이런것도 가능



## constexpr

- 변수에 사용할때
  - `#define` 대체가능
  - `const` 상수 대체가능
  - 컴파일타임에 상수로 대체
- 함수에 사용할때
  - 어느부분 TMP 대체가능
  - 컴파일타임에 계산할수도있고 안할수도있음

EUnS

- linker
  - Object File
  - symbol
  - linker
  - Executable Object Files

`#include0|8|`

template

TMP

`constexpr(c++11)`

1  
2





