EUnS

Object File

linker
Executable Object

#include0|5

template

LIVIE

constexpr(c++11

## template / linker

 $\mathsf{EUnS}$ 

January 1, 2020

LO

Object File symbol linker Executable Ob

Executable Obje Files

template

TMP

constexpr(c++11

1 linker Object File symbol

linker

Executable Object Files

- 2 #include이해
- 3 template
- 4 TMP

#### linker

Object File symbol linker

Executable Obj Files

#Include0

template

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

## Complier

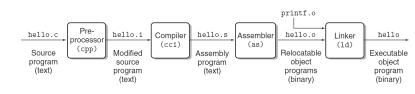


Figure: 컴파일 과정

Object File

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

EUnS

0 ELF header Object File .text .rodata .data .bss .symtab Sections .rel.text .rel.data .debug .line .strtab **Describes** object file Section header table

sections

EUnS

Object File symbol linker

linker Executable Object Files

" template

TMD

- .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

symbol linker Executable Ob

# include ∩ St

templat

IMP

- 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

- 1 symbol resolution
- 2 Relocation

#### EUnS

Object

symbol

Executable Obj

#include015

template

TMP

constexpr(c++1)

## 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 definitionss : 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

FUnS

Object Fi

Executable Object

template

. . . . . .

constexpr(c++11

## Executable Object Files

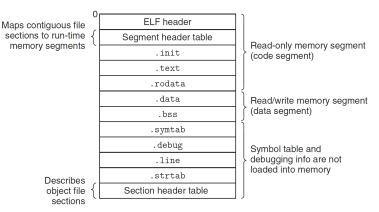


Figure: Typical ELF executable object file.

Object File

Executable Object Files

#include0|8

template

LIVIE

constexpr(c++11

## Loading

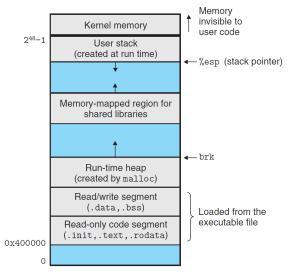


Figure: Linux x86-64 run-time memory image.

EUnS

Object Fi

Executable Obje

#include이해

template

### header file

- 그냥 텍스트파일과 같다.
- 선언을 위한것...
- 실제 구현은 어느 cpp파일에서...
- #include는 전처리 지시자.
- #include로 cpp파일에 그대로 붙여넣는다.

**FUnS** 

#### template

## template

- template을 실제로 인자를 넣어서 사용할때 전처리에서 인자를 넣은 코드를 생성함
- 타입 아무거나 넣어도 가능
- 실행 파일크기가 예상과 다르게 커질수있음

```
template / linker
     EUnS
```

#### template

```
9
```

4

6

```
{;}
int main()
     function (1,0 \times 0000,3);
     function ("s", 0x0000, "asdf");
```

char function < char > (char a , char\* b, char &c)

template < typename T >

{;}

template <>

T function (T a, T\* b, T& c)

.

Object File symbol linker

Executable Obje Files

template

template

Object File symbol linker

Executable Obje Files

#### template

TMD

Object File symbol linker

Executable Obje Files

#### template

TMD

.

Object File symbol linker

Files

### template

```
Files
```

template

TMF

```
template <int N>
struct Factorial {
    static const int result = N * Factorial <N - 1>::
    result;
};

template <>
struct Factorial <1> {
    static const int result = 1;
};

Factorial <4>::result
```

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

EUnS

Object Fil

Executable Obje

#include0|5

templat

TMP

constexpr(c++11

## TMP는 나쁘다

## 참고

- 극암의 코드 가독성
- 헬 디버깅
- 어려움
- 써보고 직접 판단 내려보자.

linker

Executable Obje

#include이해

templat

constexpr(c++11)

## constexpr

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

EUnS Executable Object Files 1 constexpr(c++11)

24 / 25

template / linker

Object Fi symbol

Executable O Files

#include0|

template

I IVII

 $\mathsf{constexpr}(\mathsf{c}{+}{+}11)$ 

과제 3-2 : 링크드리스트