Static Linking
(including dlviding your code across multiple source files)
Appropriate for 1820 onward

So far you have written many progs in a single source file. Larger progs should be written is several source files.

main.cpp

```
int main ()
{
  Foo();
}
```

other.cpp

```
void Foo()
{
}
```

When compiling main.cpp, the compiler must know Bar's signatore to know what to do with it.

A third file is used.

other.hpp

```
#pragma once
void Foo();
```

main.cpp

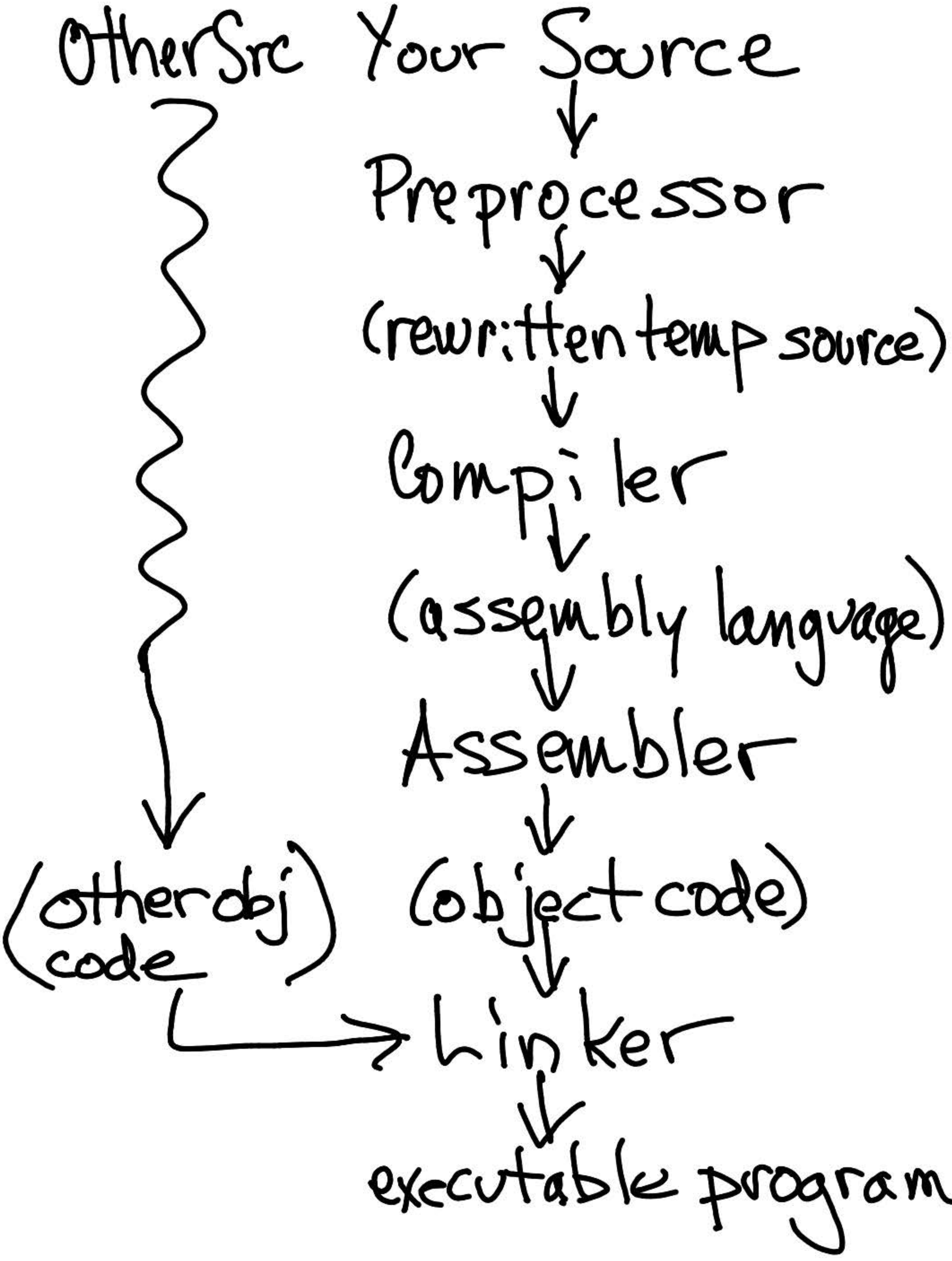
```
//includes
#include "other.hpp"

int main ()
{
    ≡
    Foo();
    ≡
}
```

① Include files provide <u>signatures</u> (and potentially other info). That is why there is no body to Bar().

② #pragma once limits the number of times the include file will be injected. This way one include including another that includes the 1st won't be a loop.

Steps in "compilation" better termed "building"



Object code From main.cpp

code: (From main.cpp)

Symbol Table:
need Foo()
have Main()

Object code from other.cpp

code: (From other.cpp)

Symbol Table:
Have Foo()

Linker

Code: from main.cpp

Code. from other.cpp

Symbol Table:*
Have Main() ✓
Have Foo() ✓

Executable*

* simplified.