Boost and C++11

Cristian Cibils
(ccibils@stanford.edu)

Administrivia

Assignment two is out!

- We're going to write a hangman program
- Due: Tuesday May 12th, 11:59 PM
- I recommend doing it
  - You must do it if you did not do GraphViz
- There's no starter code whatsoever, so all you have to download is the dictionary file

Administrivia

Apply to section lead!

Assignment One Feedback

Overall, submissions were very good.

Let's go over some of the common problems though

Assignment One Feedback

There were four main lessons in this assignment

- File I/O
- Basic usage of data structures to transform values
- Reading basic user input
- Looping for a specified amount of time

Assignment One Feedback

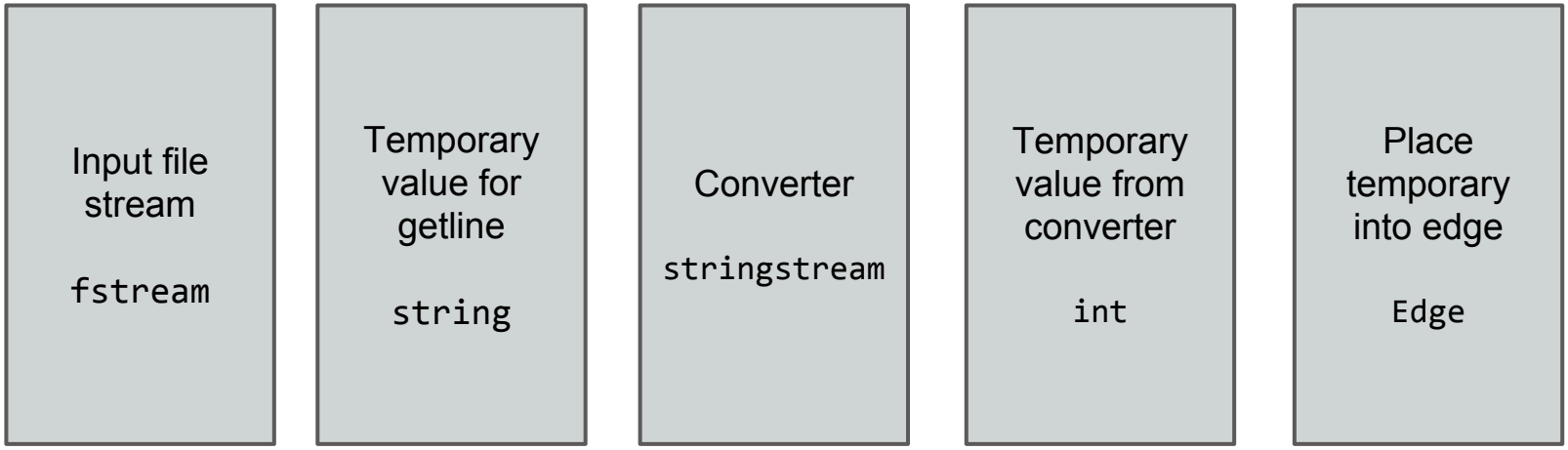
Biggest issue: File I/O
Many used getline and a stringstream to read integers from the file

Assignment One Feedback

Let's look at some correct but overly verbose file reading code.

Assignment One Feedback

Biggest issue: File I/O
Many used getline and a stringstream to read integers from the file



Assignment One Feedback

Biggest issue: File I/O

The following code is the easiest way to read in a graph:

```
size_t numNodes;
input >> numNodes;
Edge e;
while (input >> e.start >> e.end)
  graph.edges.push_back(e);
```

Assignment One Feedback

- Mixing getline and >> will result in bad things!
  - Extraneous data can be left sitting on the stream
  - In general, you should only use one or the other
  - See slides on streams for more details

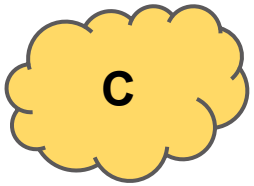
Assignment One Feedback

- Updating nodes as you iterate through them
  - If you apply attractive or repulsive forces before calculating **all** forces, then nodes will have different positions
  - This will affect the computation of later forces!

Changing C++

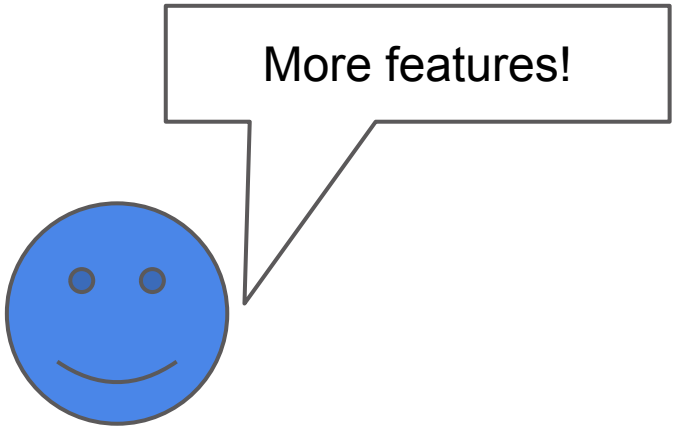
Now let’s take a quick look at the growth of C++ as a language over the years

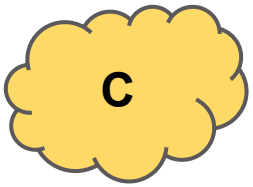
Changing C++



Changing C++

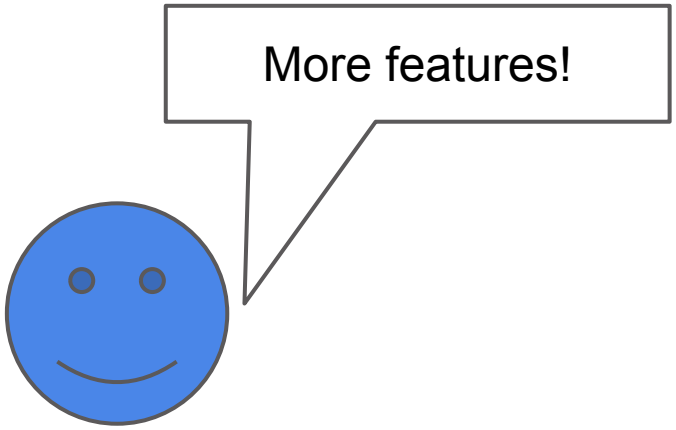
1979: 1 user

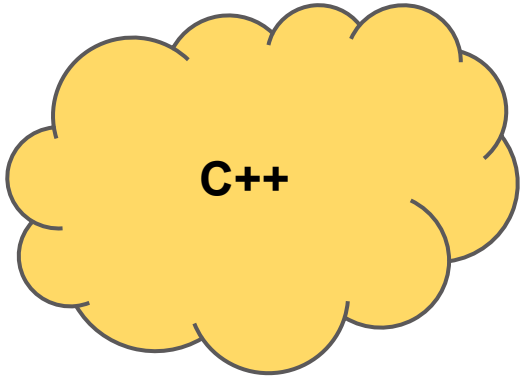




Changing C++

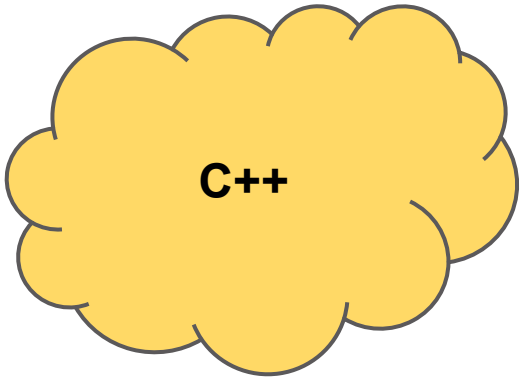
1979: 1 user





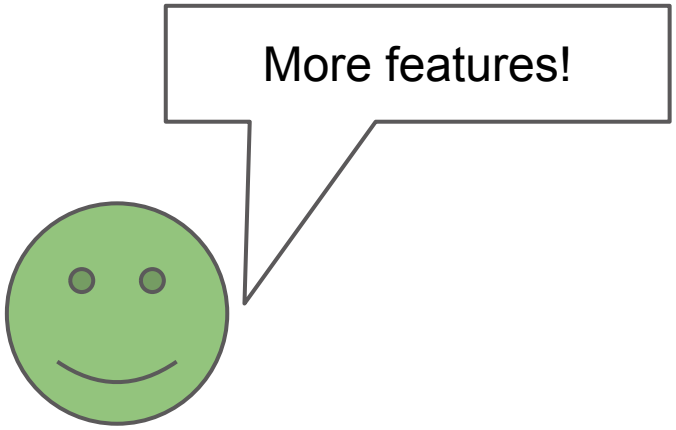
Changing C++

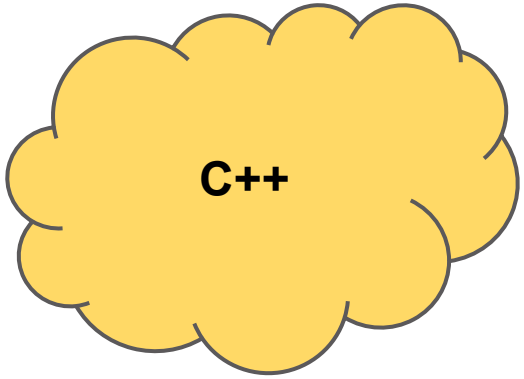
1979: 1 user



Changing C++

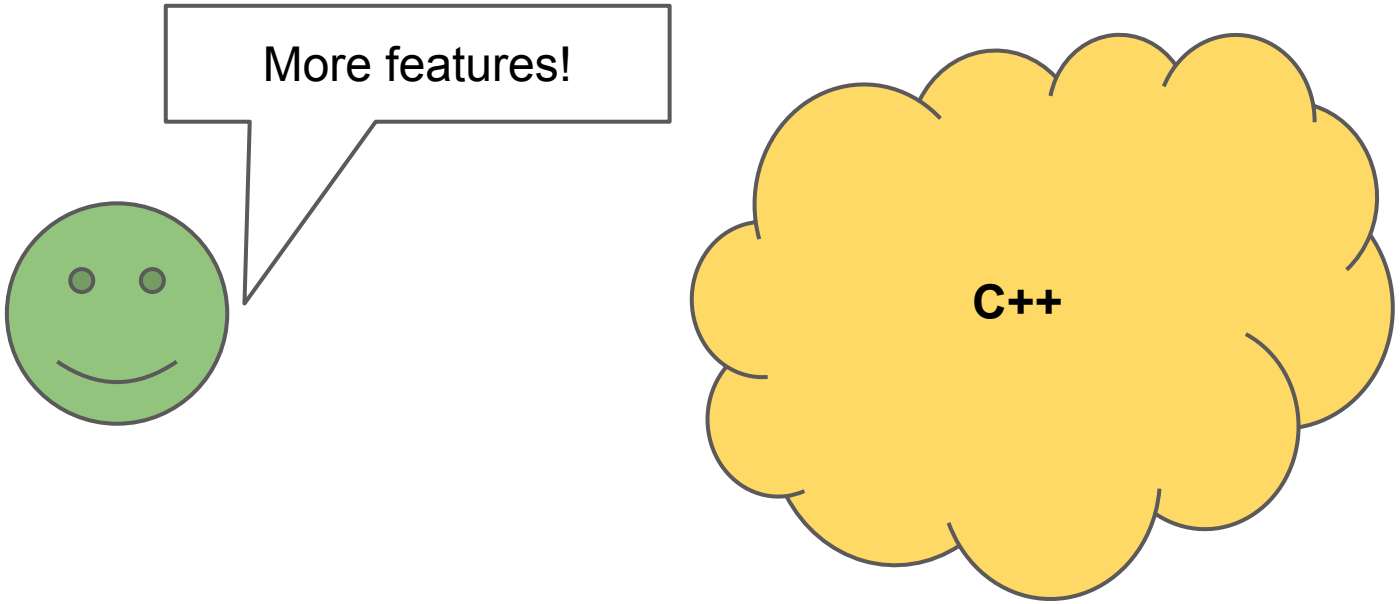
1985: 500 users





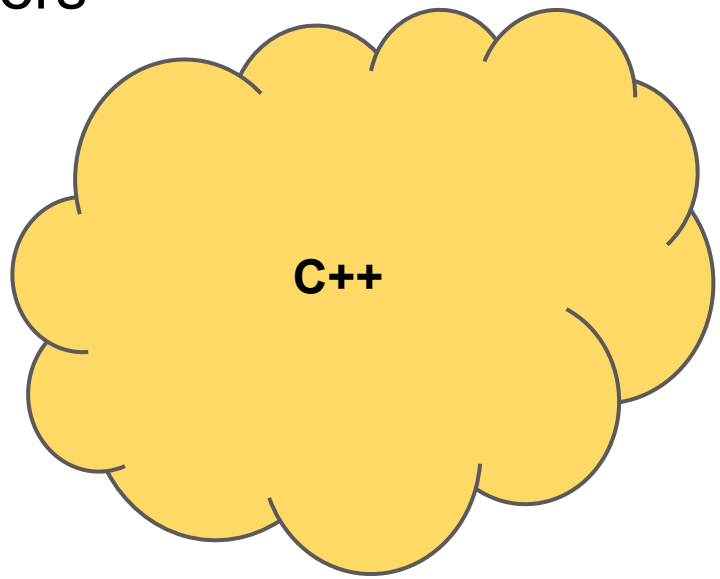
Changing C++

1985: 500 users



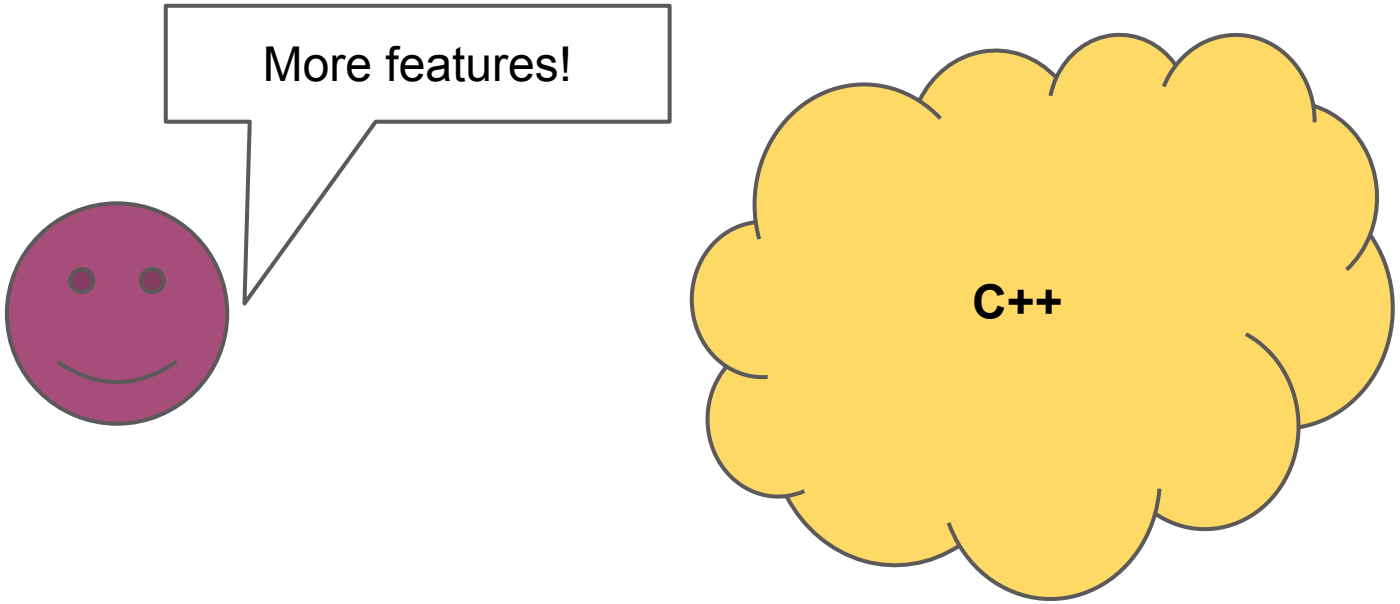
Changing C++

1985: 500 users



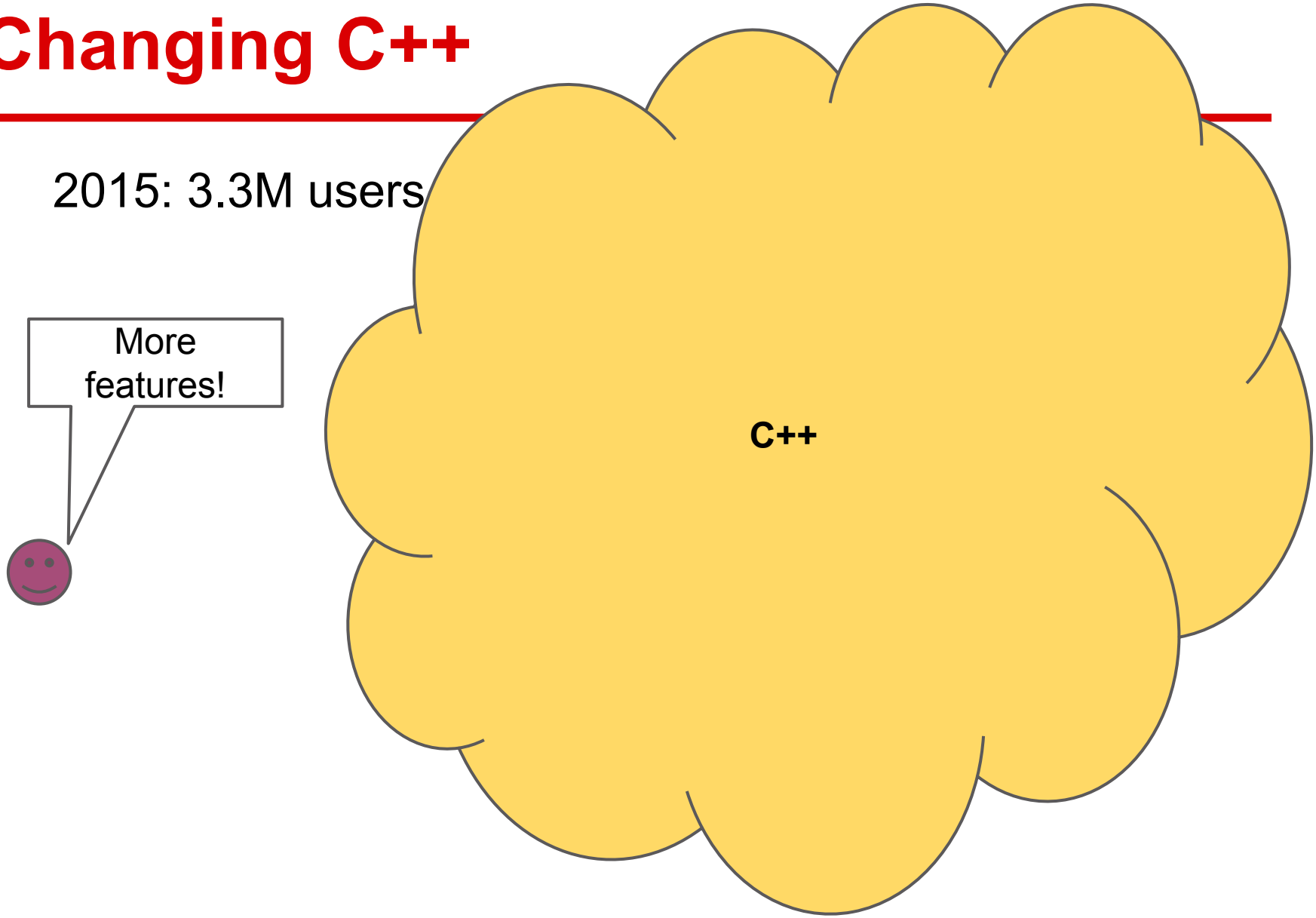
Changing C++

2015: 3.3M users



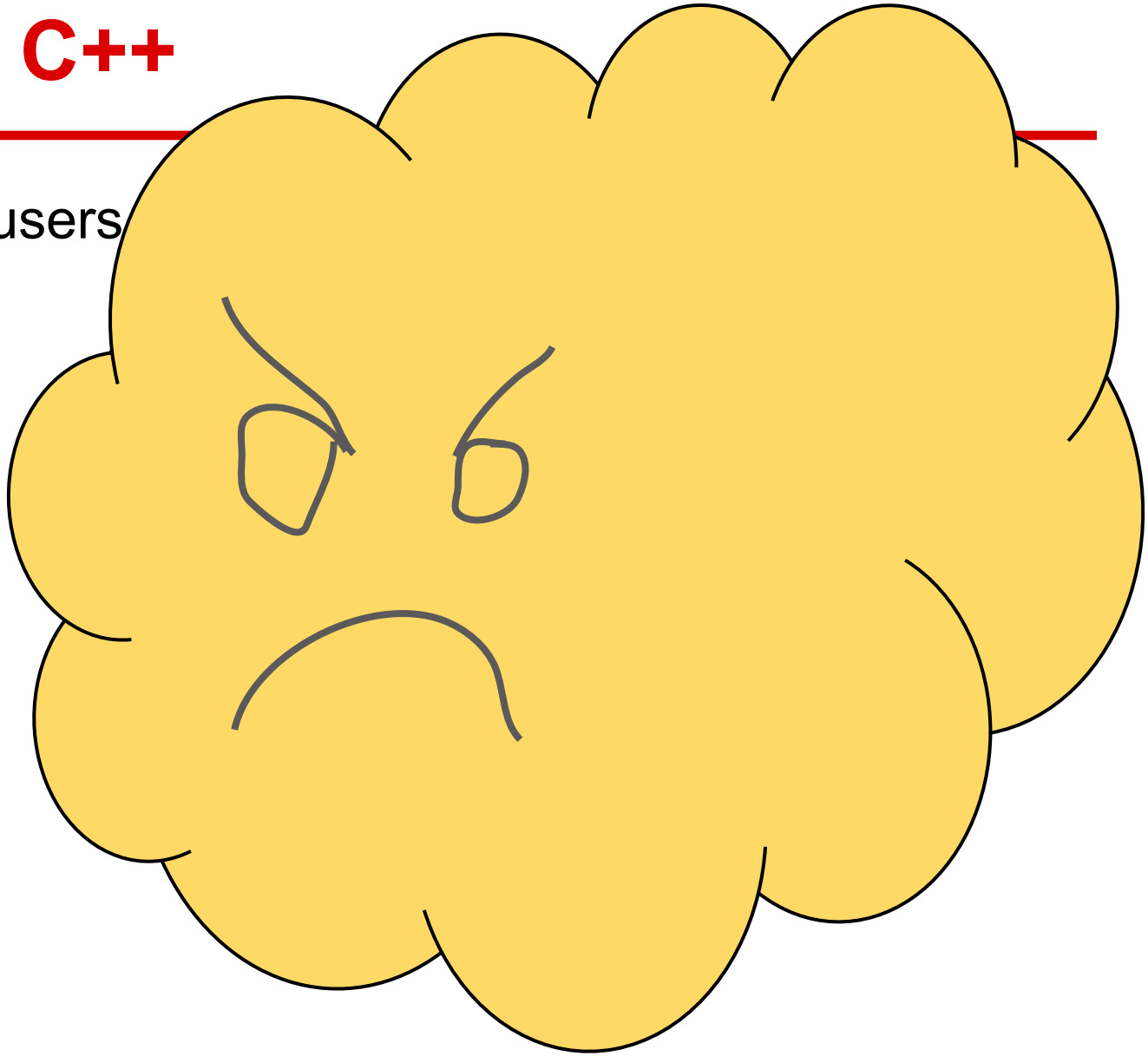
Changing C++

2015: 3.3M users



Changing C++

2015: 3.3M users



Solution: Extensible Language

Extensibility

Let C++ users add features to the language without changing the definition of the language

Boost

- Huge (19 million lines of code) collection of third party C++ code
- High quality
- Many libraries incorporated into C++11

Boost

How much C++ you need to know to understand...

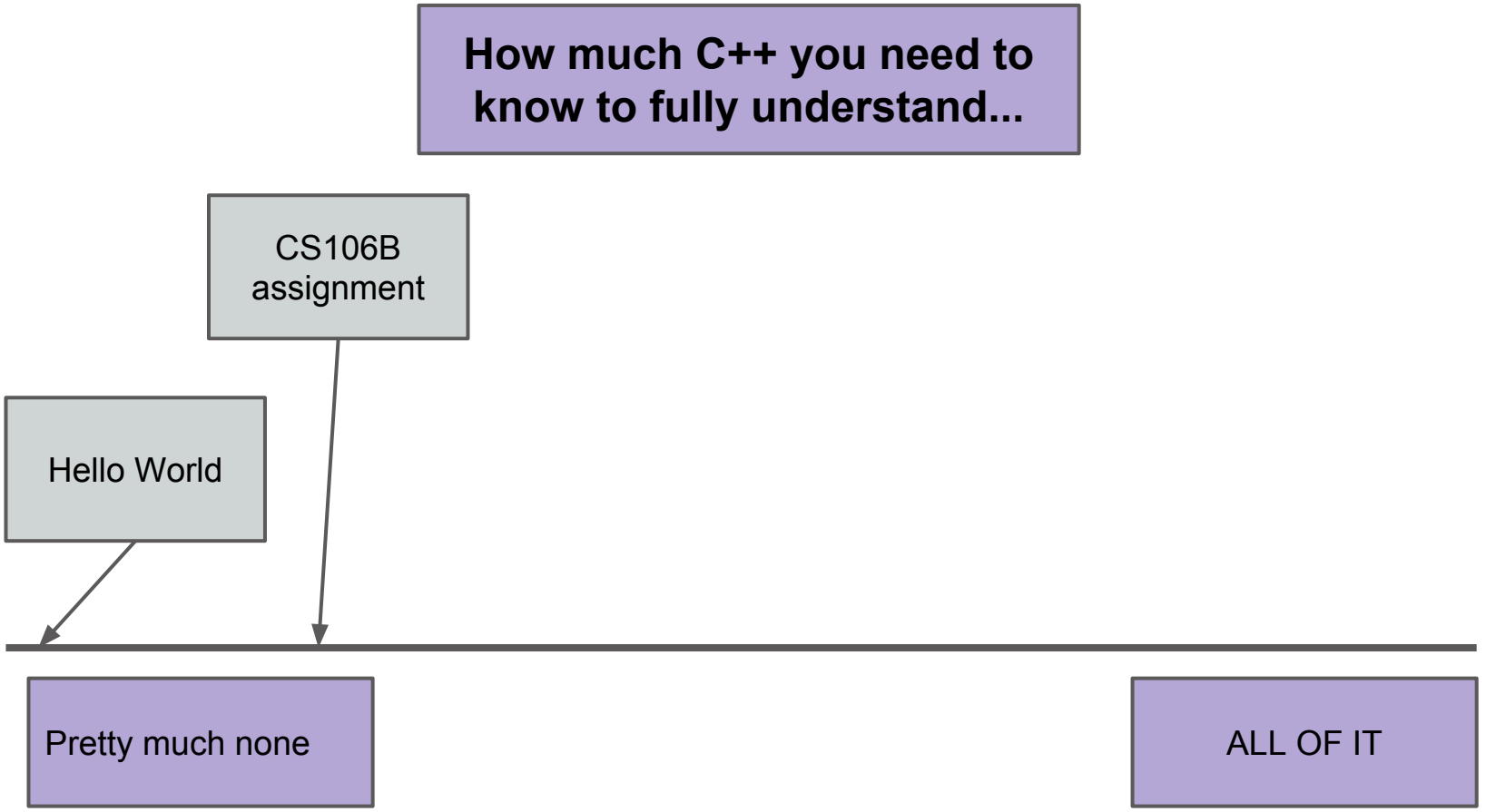
Pretty much none

ALL OF IT

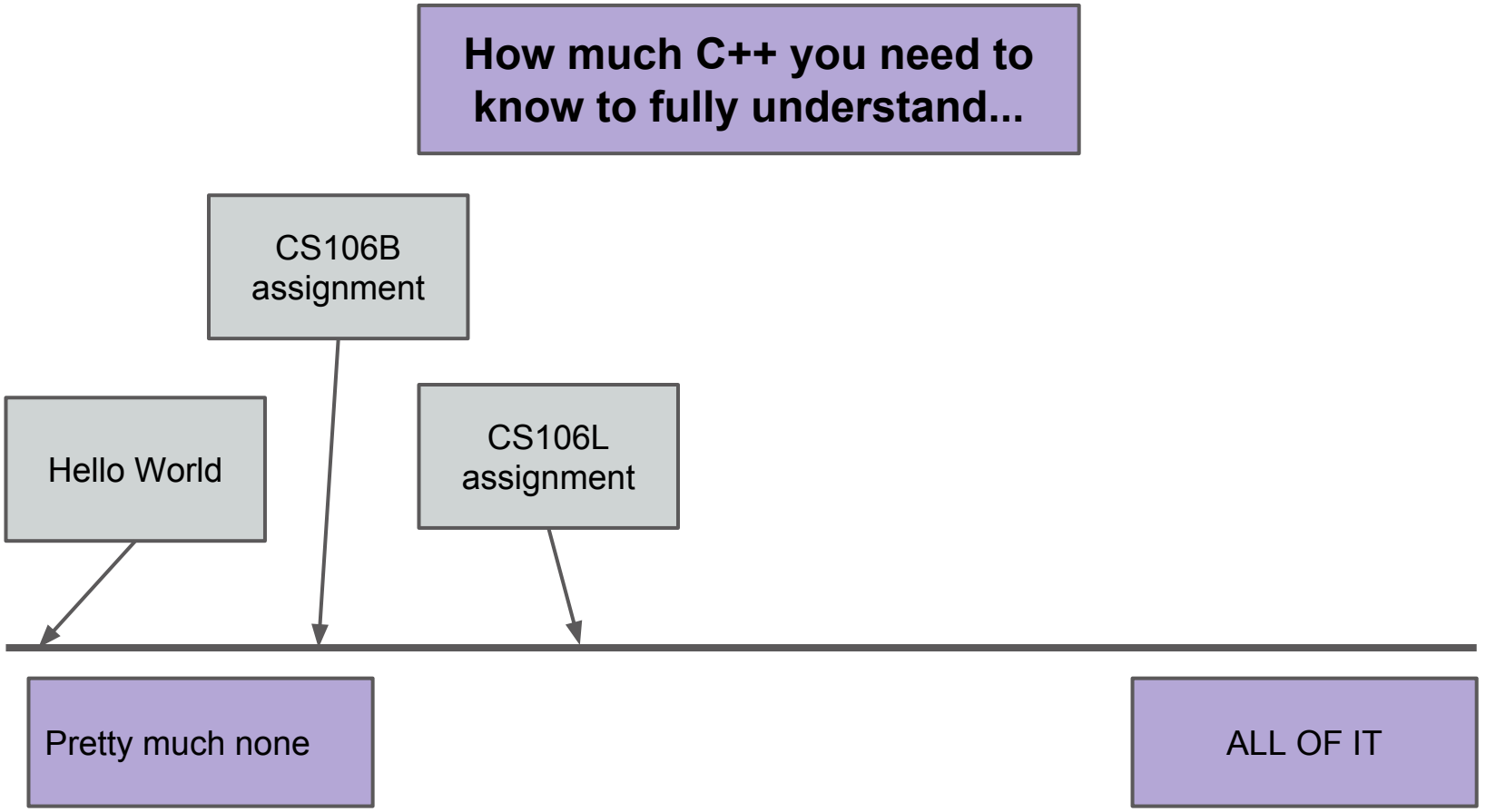
Boost

How much C++ you need to know to fully understand...

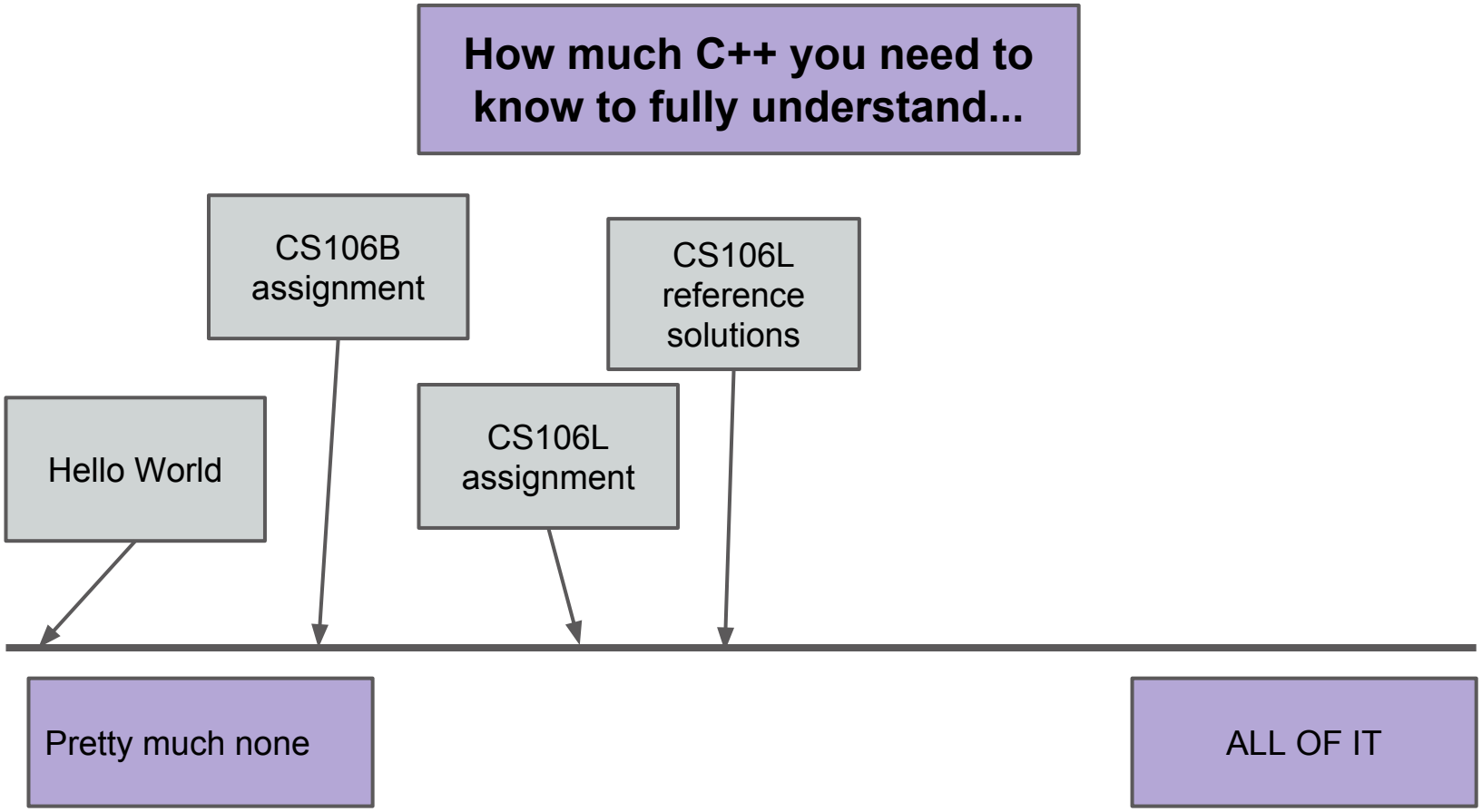
Boost


How much C++ you need to know to fully understand...
CS106B assignment
Hello World
Pretty much none
ALL OF IT

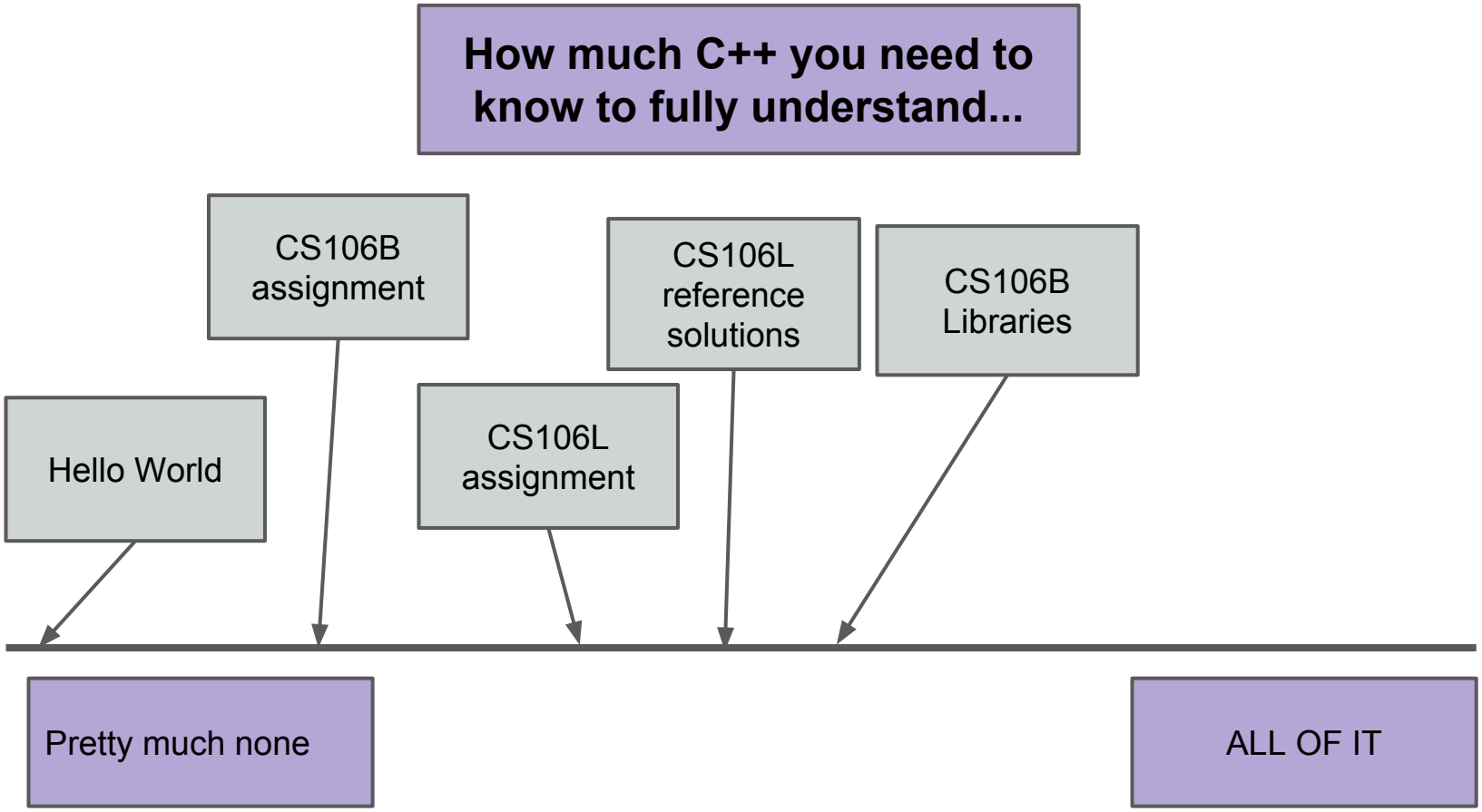
Boost


How much C++ you need to know to fully understand...
CS106B assignment
Hello World
CS106L assignment
Pretty much none
ALL OF IT

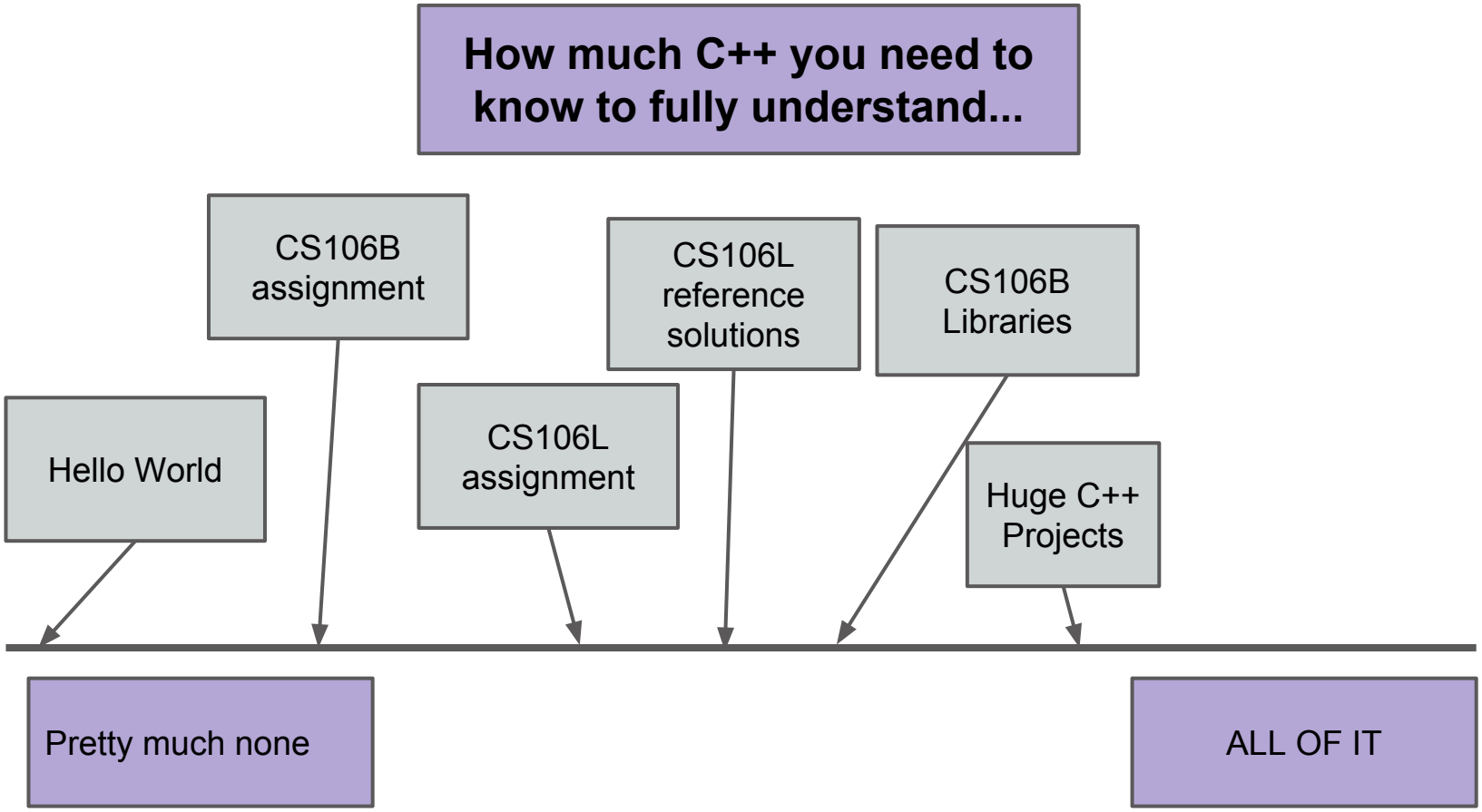
Boost


How much C++ you need to know to fully understand...
CS106B assignment
CS106L reference solutions
Hello World
CS106L assignment
Pretty much none
ALL OF IT

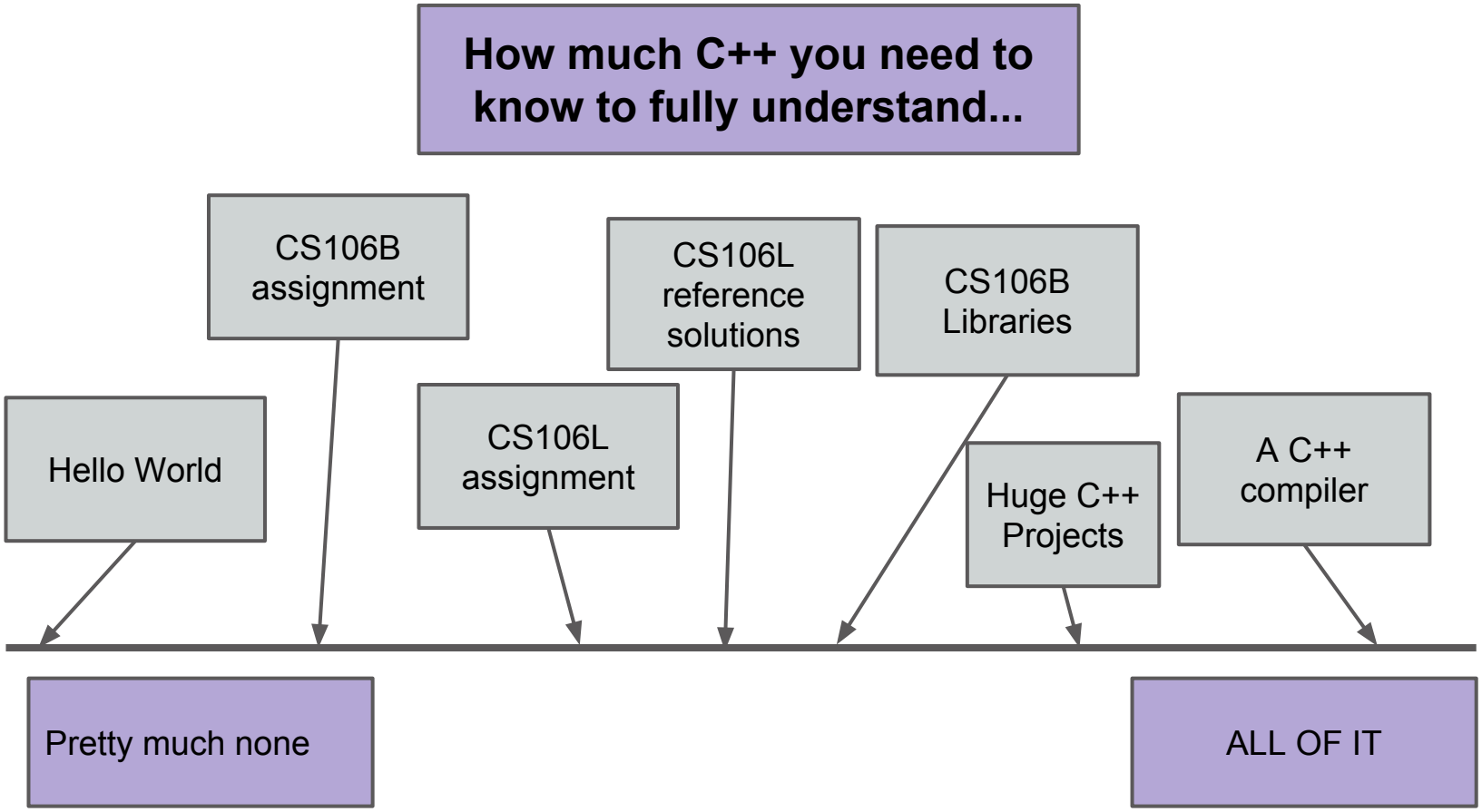
Boost


How much C++ you need to know to fully understand...
CS106B assignment
CS106L reference solutions
CS106B Libraries
Hello World
CS106L assignment
Pretty much none
ALL OF IT

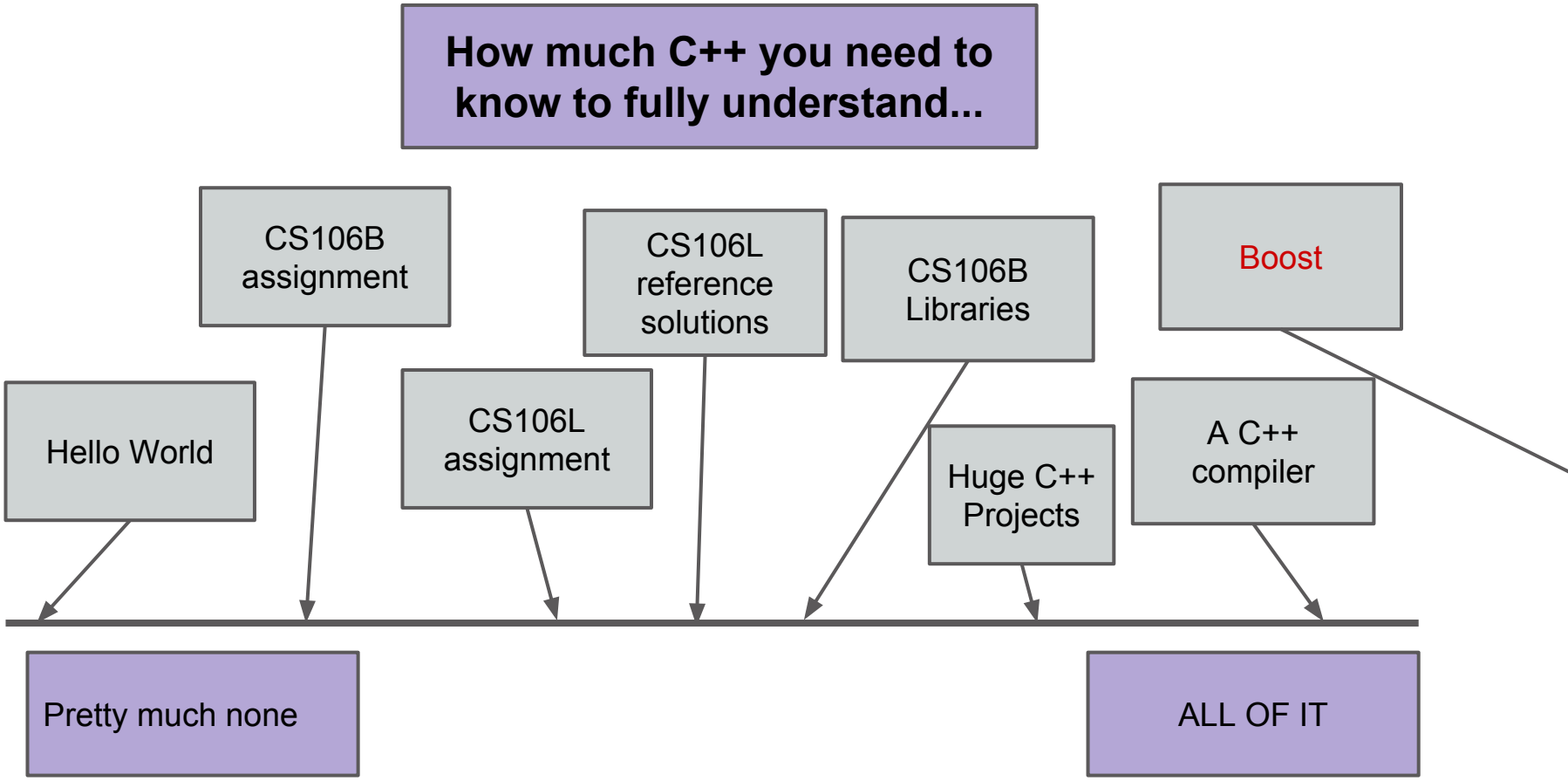
Boost


How much C++ you need to know to fully understand...
CS106B assignment
CS106L reference solutions
CS106B Libraries
Hello World
CS106L assignment
Huge C++ Projects
Pretty much none
ALL OF IT

Boost


How much C++ you need to know to fully understand...
CS106B assignment
CS106L reference solutions
CS106B Libraries
Hello World
CS106L assignment
Huge C++ Projects
A C++ compiler
Pretty much none
ALL OF IT

Boost


How much C++ you need to know to fully understand...
CS106B assignment
CS106L reference solutions
CS106B Libraries
Boost
Hello World
CS106L assignment
Huge C++ Projects
A C++ compiler
Pretty much none
ALL OF IT

Boost

Boost uses the extensibility of C++ to add features to the language without requiring a new version of the language itself

Boost

- We often talk about converting between integers and strings
- What if there was an easier way?
- Like, **much** easier

Boost

See code in LexicalCast.pro

Boost and C++11

- Boost uses the extensibility of C++ to add features to the language without requiring a new version of the language itself
- Sometimes, Boost's changes will be incorporated into C++ itself
- Let's take a look at some examples.

C++11

- Let's take a look at how foreach came to be included in the language
- `BOOST_FOREACH` first introduced in 2004
- Stanford's foreach loop does similar stuff
- This was standardized to the "range-based for" loop we know and love in C++11

```
set<int> s;
for (int x : s)
   cout << x << endl;
```

C++11

See code in ForEach.pro

Auto

- Now let's look at a C++11 feature, `auto`
- The new keyword `auto` allows a programmer to simplify the declaration of a variable
- Variables declared `auto` have their type inferred from the right of the = sign

```
auto x = 0;
auto y = 0.0;
```

Auto

See code in Auto.pro

Lambdas

- Let’s look at another cool feature: lambdas
- Lambdas allow us to define a function just like we define a variable
- It’s tricky to explain why this is useful, but let’s take a look at a few examples to understand why

Lambdas

Let’s take a look at a simple use of lambdas (BasicLambda.pro)

Lambdas

Lambda Syntax:

```
[captured variables](paramaters) {
   //function stuff
};
```

Lambdas

Lambda Syntax:

```
int x;
[x](paramaters) {
   //function stuff
};
```

Captures x by value

Lambdas

Lambda Syntax:

```
int x;
[&x](paramaters) {
   //function stuff
};
```

Captures x by reference

Lambdas

Lambda Syntax:

```
[&=](paramaters) {
   //function stuff
};
```

Captures all variables by reference

Comparator Function

- Many algorithms and containers have a concept of order
- Ascending order → n1 < n2 < …< n100
- A **Comparator Function** in C++ is a function that determines whether one element is less than another
- Equivalent to implementing the ‘<’ operator

Comparator Function

```
bool compareInt(int x, int y) {
   return x < y;
}

bool reverseCompareInt(int x, int y) {
   return y < x;
}
```

Lambdas

Let’s take a look at some more advanced uses of lambdas (BasicLambda.pro)

C++11

- C++11 added much more than just range based for, type inference, and lambdas
  - Whole new set of algorithms including all_of, is_sorted, and many more
  - Better random number library
  - Better functions to measure time
  - Multithreading
  - much, much more