CSE110A: Compilers: Introduction Part 2.

- *Compiler Overview*
  - What is a compiler
  - What are the different stages of a compiler
    - Frontend
    - Intermediate
    - Backend

back

Topics:

- Introduction to compilers
- Compiler architecture

Topics:

- **Introduction to compilers**

- Compiler architecture

What is a compiler?

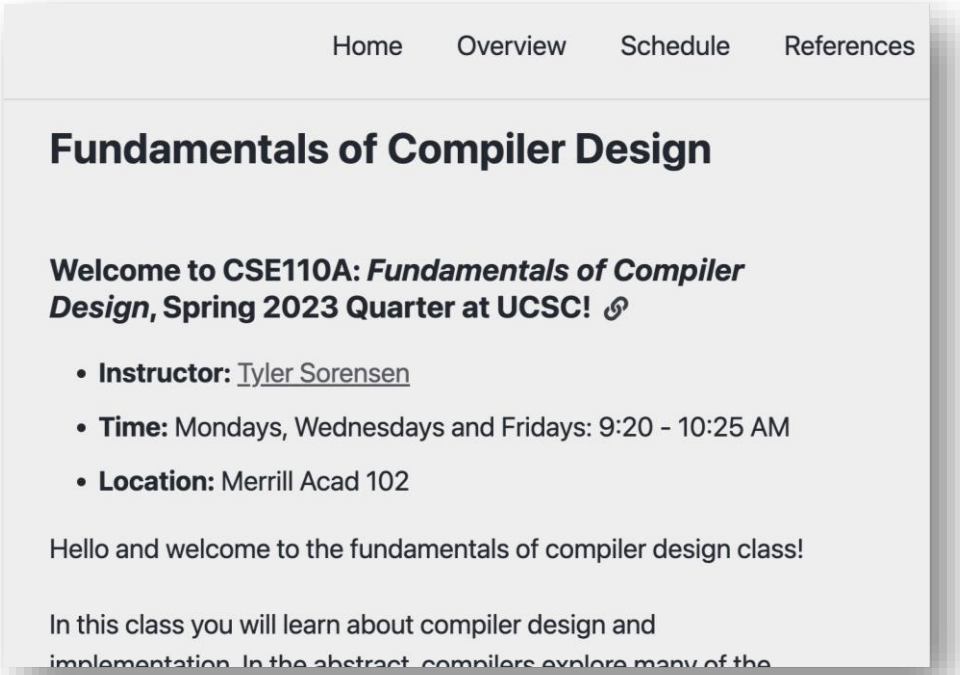
What is a compiler?

This is way too general to be useful
Any program fits this description.

What is a compiler?

A theoretical answer

```
---
title: "Fundamentals of Compiler Design"
layout: single
---


### Welcome to **CSE110A:** _Fundamentals of Compiler Design_, Spring 2022 Quarter at UCSC!

- **Instructor:** [Tyler Sorensen](https://users.soe.ucsc.edu/~tsorensen/)
- **Time:** Mondays, Wednesdays and Fridays: 4:00 - 5:05 pm
- **Location:** Porter 144

Hello and welcome to the fundamentals of compiler design class!

In this class you will learn about compiler design and implementation. In the abstract, compilers explore many of the [foundational problems in computer science](https://en.wikipedia.org/wiki/Halting_problem). In practice, compilers are [massive pieces of well-oiled software](https://www.phoronix.com/scan.php?page=news_item&px=MTg3OTQ), and are some of the engineering marvels of the modern world.

_COVID Note_ : The last few years have been difficult due to the COVID pandemic. Public health concerns and policies remain volatile. The first priority in this class in your health and well-being. We will approach any challenges that arise with compassion and understanding. I expect that you will do the same, both to the teaching staff and to your classmates. We will follow university guidelines and work together to have a productive and fun quarter.
```



Building this website started with:

- Markdown to describe the page
- compiled with Jekyll to a static webpage
- static webpage is in HTML and javascript

This would be a compiler

What is a compiler?

A more traditional description
What are some examples here?

Strings belonging to language L

A series of statements in programming language L

Strings belonging to language L’

An executable binary file in an ISA language

What is a compiler?

A classic example

Strings belonging to language L		Strings belonging to language L’
A series of statements in programming language L		An executable binary file in an ISA language
A program written in C	gcc/clang	An x86 Binary executable

GCC and Clang

- Two mainstream compiler frameworks

- Similarities and differences?

How about this answer?

What is a compiler?

```
int main() {
    printf("hello world\n");
}
```

`gcc main.c`

Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

What is a compiler?

What is wrong with this picture?

```
int main() {
    printf("hello world\n");
}
```

```
$ ./a.out
hello CSE 110A
```

`gcc main.c`

Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

What is a compiler?

A valid input must have a equivalent valid output.
Semantic equivalence

Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

What is a compiler?

What is wrong with this picture?

```
int main() {
  printf("hello world\n");
}
```

```
$ ./a.out
hello CSE 110A
```

`gcc main.c`

Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

What is a compiler?

```c
int main() {
  printf("hello world\n");
}
```

```
$ ./a.out
hello world
```

`gcc main.c`

Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

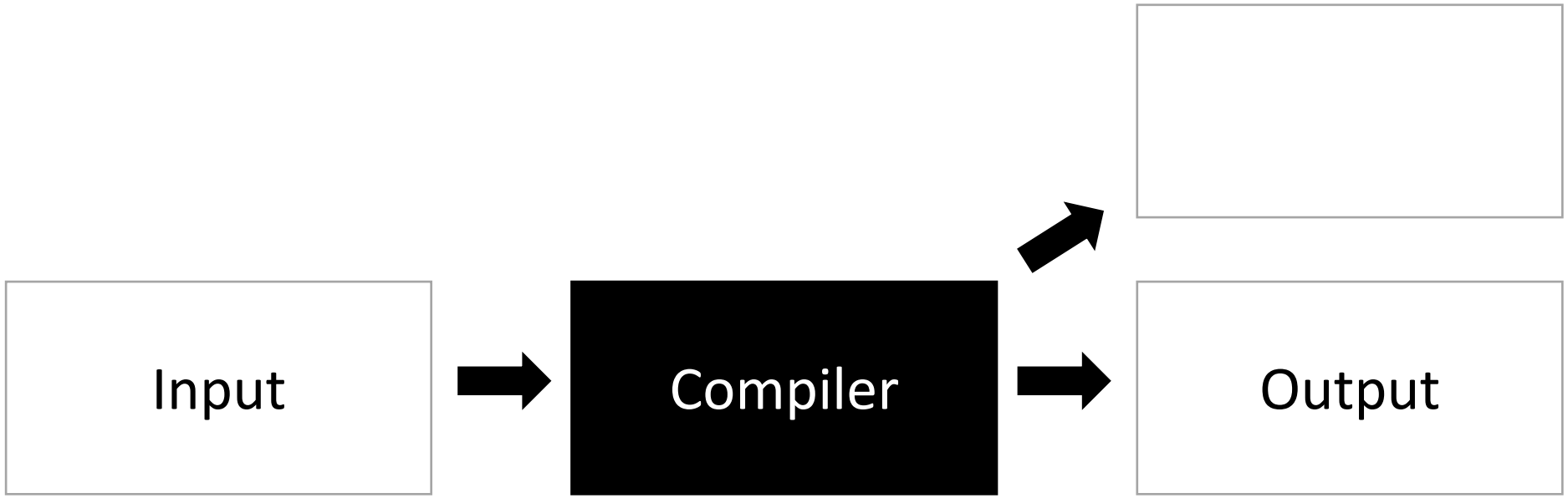
Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

What is a compiler?

What else does a compiler give you?



Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

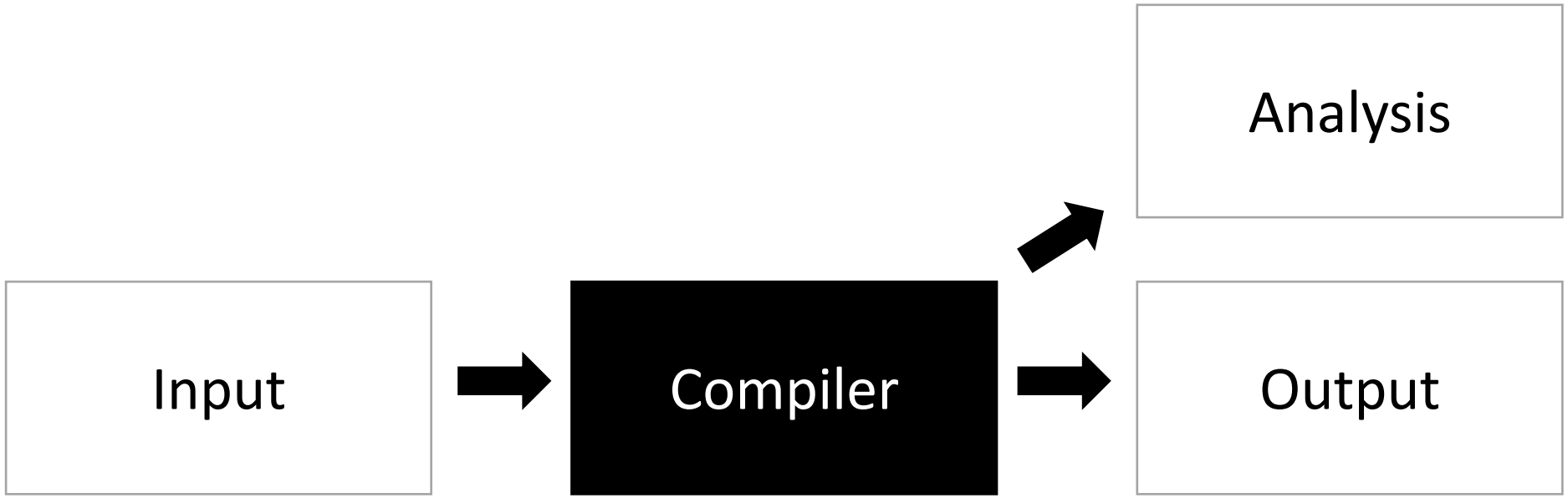
Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

What is a compiler?

What are some examples here?



Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

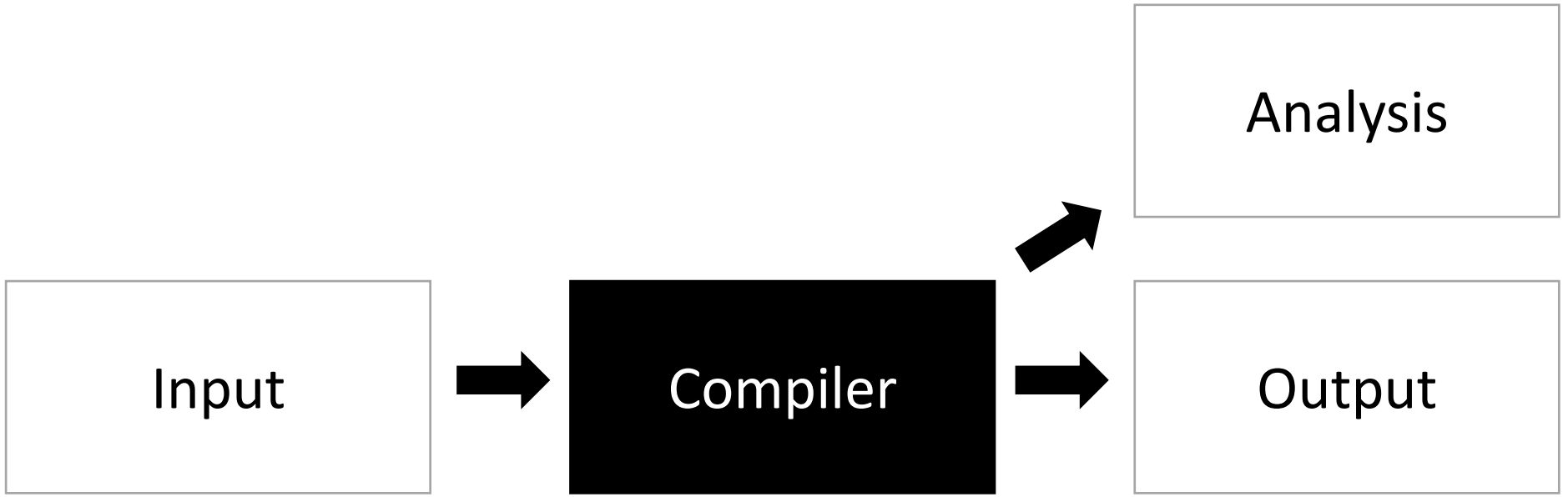
Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

What is a compiler?

What are some examples here?



Warnings
Errors
Performance logs

Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

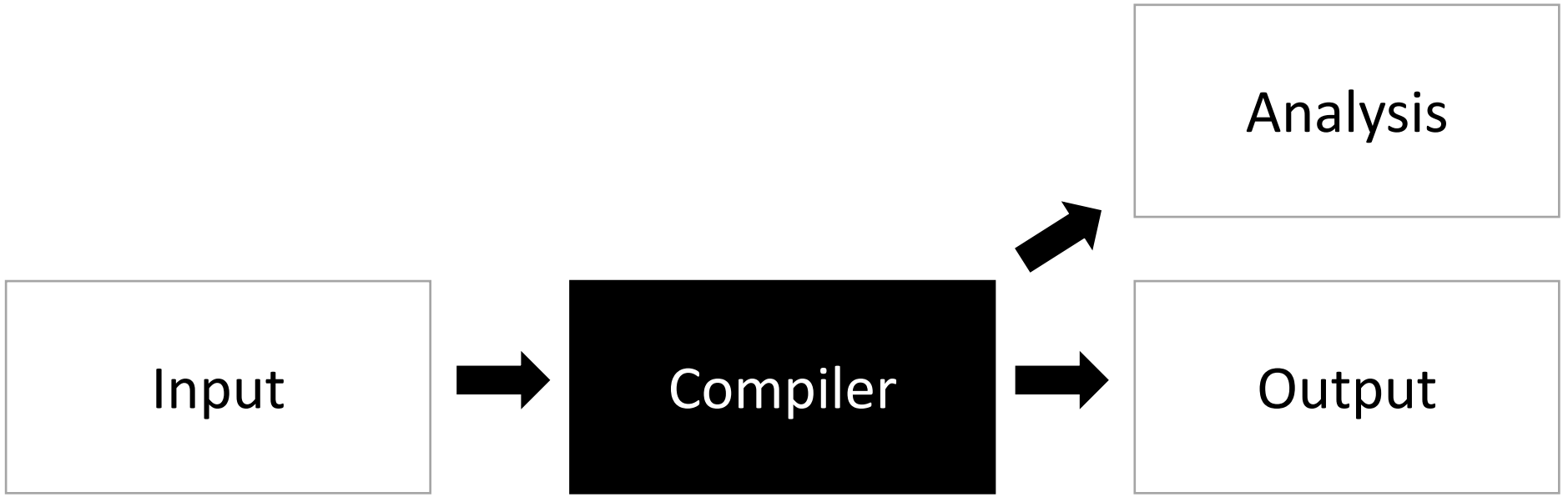
Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

What is a compiler?

What are some examples here?



Warnings
Errors
Performance logs

Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

Demo

- What are some examples of code that might give a warning?

What can happen when the input isn't valid?

```
int foo() {
  int x;
  int y = x;
  return y;
}
```

Try running this through the compiler

What can happen when the Input isn't valid?

```
int foo() {
  int x;
  int y = x;
  return y;
}
```

```
int foo(int condition) {
  int x;
  if (condition) {
    x = 5;
  }
  int y = x;
  return y;
}
```

What about this one?

Try running this through the compiler

test4.c

What is a compiler?

A valid input must have an equivalent valid output.
Semantic equivalence

Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

Strings belonging to language L’

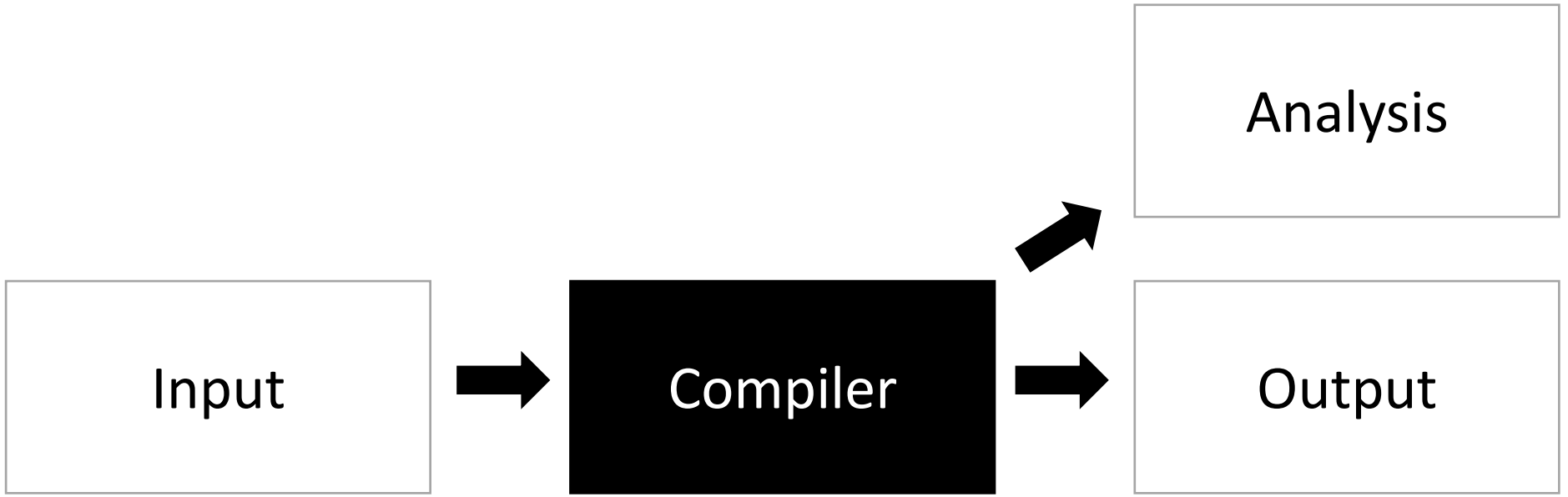
An executable binary file in an ISA language

An x86 Binary executable

Uninitialized variable example

What is a compiler?

What are some examples here?



Warnings
Errors
Performance logs

Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

What can happen when the input isn’t valid?

```
int foo() {
  int my_var = 5;
  my_var = my_car + 5;
  return my_var;
}
```

Try running this through a compiler

test1.c

What can happen when the Input isn’t valid?

```
int foo() {
  int my_var = 5;
  my_var = my_car + 5;
  return my_var;
}
```

Try running this through a compiler

You get an error and a suggestion these days

What can happen when the Input isn't valid?

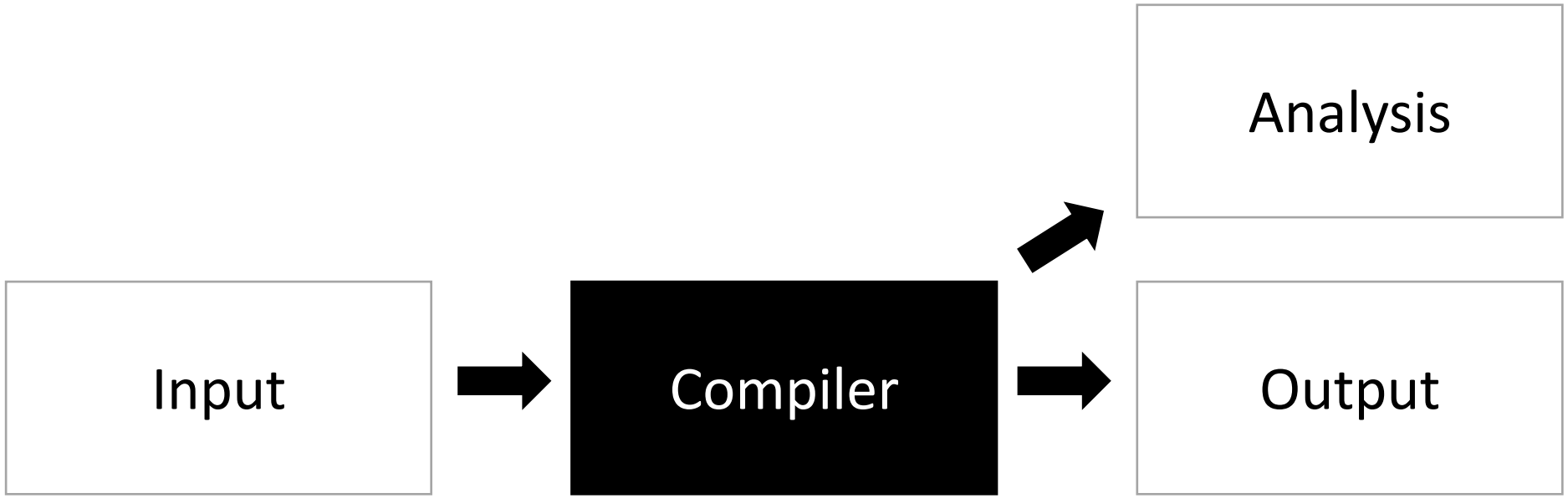
```
int foo() {
  int *x = malloc(100*sizeof(int));
  return x[100];
}
```

What about this one? No error…

What sort of errors are compilers good at catching?
What ones are they not?

What is a compiler?

What are some examples here?



Warnings
Errors
Performance logs

Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

Architecture Aware Optimizations

- Example

```
int foo(int *x, int *y, int *z) {
  for (int i = 0; i < 128; i++) {
    z[i] = y[i] + x[i];
  }
  return 0;
}
```

How can we know what the compiler is doing?

```
#define SIZE (1024*1024)
int add(int * a, int * b, int * c) {
  for (int i = 0; i < SIZE; i++) {
    a[i] = b[i] + c[i];
  }
  return 0;
}
```

Use of compiler flags

```
clang –O3 -Rpass-missed=loop-vectorize \
          -Rpass=loop-vectorize  vector.c
```

How can we know what the compiler is doing?

```
#define SIZE (1024*1024)
int add(int * a, int * b, int * c) {
  for (int i = 0; i < SIZE; i++) {
    a[i] = b[i] + c[i];
  }
  return 0;
}
```

Use of compiler flags

```
clang –O3 –Rpass-missed=loop-vectorize \
          -Rpass=loop-vectorize  vector.c
```

-R stands for report flags.

-Rpass-analysis=<value> Report transformation analysis from optimization passes whose name matches the given POSIX regular expression

-Rpass-missed=<value> Report missed transformations by optimization passes whose name matches the given POSIX regular expression

-Rpass=<value> Report transformations performed by optimization passes whose name matches the given POSIX regular expression

COMPILE TIME REPORT IS:

```
vector.c:5:2: remark: vectorized loop (vectorization width: 4, interleaved count: 2) [-Rpass=loop-vectorize]
      for (int i = 0; i < SIZE; i++) {
      ^
```

Does the compiler need to perform every step?

```
int foo() {
  int my_var = 0;
  for (int i = 0; i < 128; i++) {
    my_var++;
  }
  return my_var;
}
```

Does the compiler need to perform every step?

```
int foo() {
  int my_var = 0;
  for (int i = 0; i < 128; i++) {
    my_var++;
  }
  return my_var;
}
```

Mentally we probably step through the for loop:

Does the compiler need to perform every step?

```
int foo() {
  int my_var = 0;
  for (int i = 0; i < 128; i++) {
    my_var++;
  }
  return my_var;
}
```

Mentally we probably step through the for loop:

What does the compiler do with this?

What is a compiler?

A valid input must have a equivalent valid output.
Semantic equivalence

Strings belonging to language L

A series of statements in programming language L

A program written in C

gcc/clang

Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

Does the compiler need to perform every step?

```
int foo() {
  int my_var = 0;
  for (int i = 0; i < 128; i++) {
    my_var++;
  }
  return my_var;
}
```

```
int foo() {
  return 128;
}
```

are these the same?

Does the compiler need to perform every step?

```
int foo() {
  int my_var = 0;
  for (int i = 0; i < 128; i++) {
    my_var++;
  }
  return my_var;
}
```

```
int foo() {
  return 128;
}
```

are these the same?

Functionally - they are the same
Non-functionally - they are not

Most compilers are concerned only with functional equivalence

Topics:

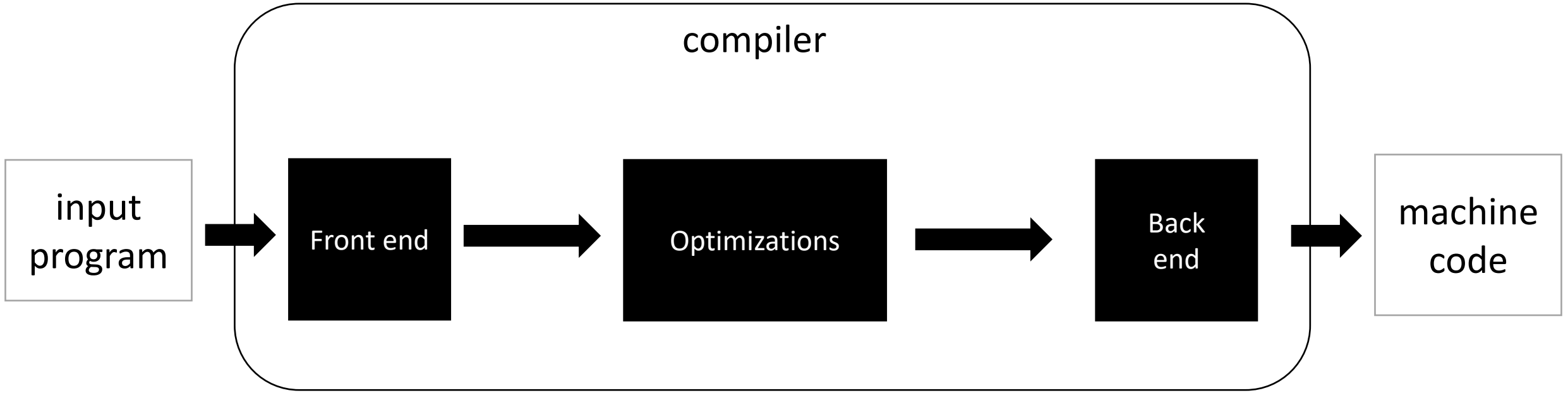
- Introduction to compilers
- **Compiler architecture**

Compiler Architecture

Compiler Architecture

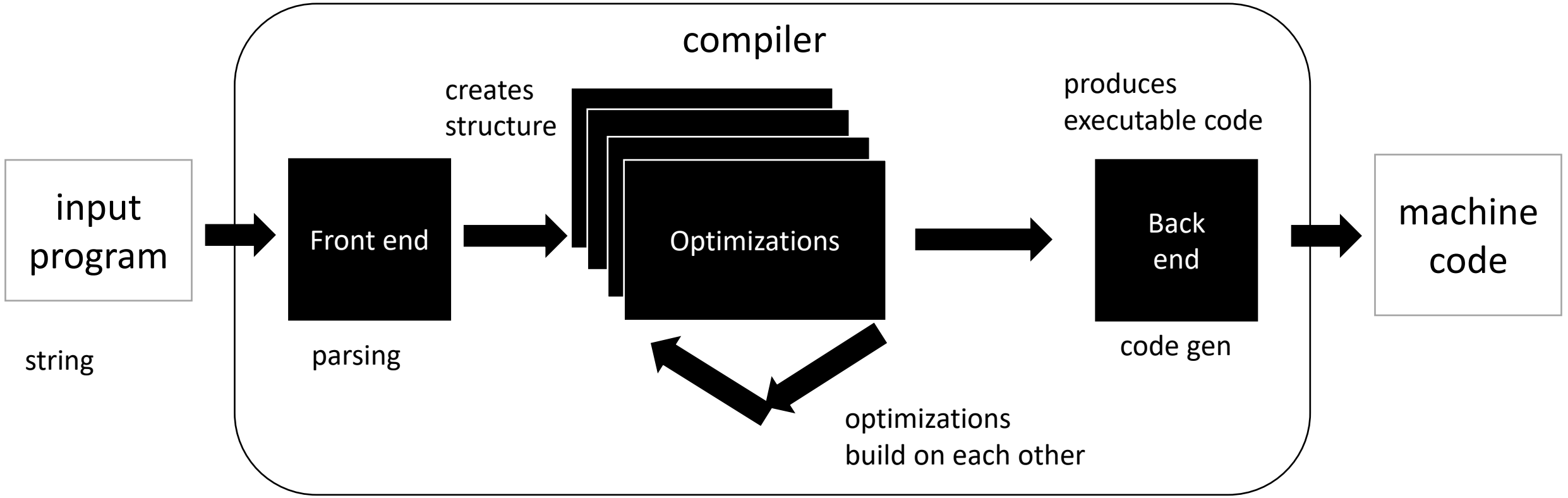
Compilers are complicated and this image is too simple

Compiler Architecture



Medium detailed view

Compiler Architecture



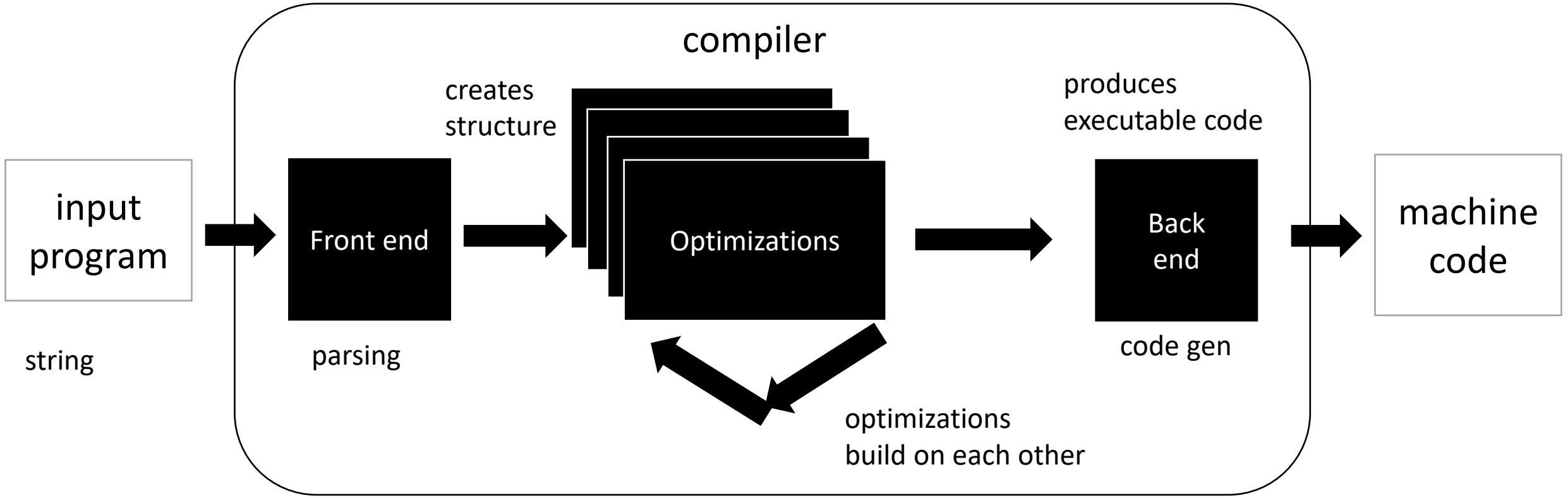
Medium detailed view

More about optimizations: https://stackoverflow.com/questions/15548023/clang-optimization-levels

Compiler Architecture

What are some of the benefits of this design?

What are some of the drawbacks of this design?



Medium detailed view

more about optimizations: https://stackoverflow.com/questions/15548023/clang-optimization-levels

We need a more detailed compiler view, but this cannot not fit it on one slide.

So let’s take a compiler journey!!

back