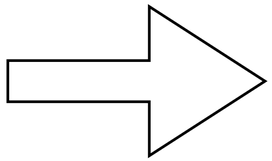
What is a Compiler?

traditionally …

- A program that *translates* from a high-level language (e.g., C++) to low-level assembly language that can be executed by hardware

```
int a, b;
a = 3;
if (a < 4) {
   b = 2;
} else {
   b = 3;
}
```

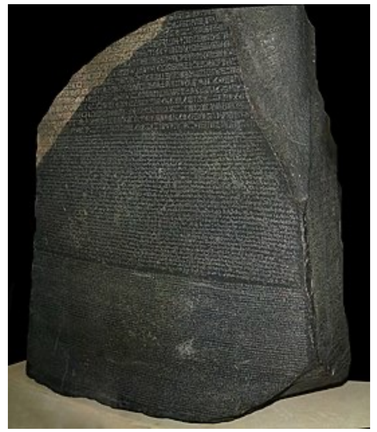


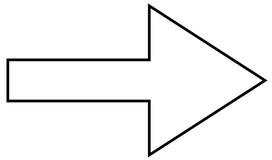
```
        var a
        var b
        mov 3 a
        mov 4 r1
        cmpi a r1
        jge l_e
        mov 2 b
        jmp l_d
l_e:    mov 3 b
l_d:    ;done
```

… but really

- A program that translates or transforms one *representation* of a program to *another* [and perhaps does some analysis along the way]

  - Fortran
  - C
  - C++
  - Java
  - Text processing language
  - HTML/XML
  - Command & Scripting Languages
  - Natural language
  - Domain specific languages





  - Machine code
  - Virtual machine code
  - Transformed source code
  - Augmented source code
  - Low-level commands
  - Semantic components
  - Abstract syntax trees
  - Another language

historically (66 years ago)

The FORTRAN Automatic Coding System

J. W. BACKUS†, R. J. BEEBER†, S. BEST‡, R. GOLDBERG†, L. M. HAIBT†,
H. L. HERRICK†, R. A. NELSON†, D. SAYRE†, P. B. SHERIDAN†,
H. STERN†, I. ZILLER†, R. A. HUGHES§, AND R. NUTT‖

INTRODUCTION

THE FORTRAN project was begun in the summer of 1954. Its purpose was to reduce by a large factor the task of preparing scientific problems for IBM's next large computer, the 704. If it were possible for the 704 to code problems for itself and produce as

system is now complete. It has two components: the FORTRAN language, in which programs are written, and the translator or executive routine for the 704 which effects the translation of FORTRAN language programs into 704 programs. Descriptions of the FORTRAN language and the translator form the principal

"AI is whatever hasn't been done yet." – Tesler's Theorem

philosophically

Chris Lattner (designer of LLVM and Swift programming language):

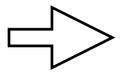
> “The most important part of a compiler is: the **design**, **representation**, **validation** and **translation** of structured data. The mentality and design center crosscuts all of computing, from the simplest json payload to the most fiddly compiler IR

We will touch on different aspects of this in this course

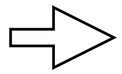
optimization as translation

- When you translate from English to Mandarin, there are many choices in how you translate
  - What kinds of vocabulary? What kinds of idioms?
  - Translate literally? Or get the right meaning across?
- **Key**: different translations have different properties
- Code translation is the same!

```
int a, b, c;
b = a + 3;
c = a + 3;
```



```
lw r1 a
addi r2 r1 3
sw r2 b
lw r3 a
addi r4 r3 3
sw r4 c
```



```
lw r1 a
addi r2 r1 3
sw r2 b
sw r2 c
```

why do we need compilers?

- Compilers provide portability
- Old days: whenever a new machine was built, programs had to be rewritten to support new instruction sets
- IBM System/360 (1964): Common Instruction Set Architecture (ISA) — programs could be run on any machine which supported ISA
  - Common ISA is a huge deal (note continued existence of x86)
- But still a problem: when new ISA is introduced (RISC-V) or new extensions added (x86-64), programs would have to be rewritten
- Compilers bridge this gap: write new compiler for an ISA, and then simply recompile programs!

why do we need compilers?

- Compilers enable high performance and programming productivity
- Old: programmers wrote in assembly language, architectures were simple (no pipelines, caches, etc.)
  - Close match between programs and machines — easier to achieve performance
- New: programmers write in high level languages (Ruby, Python), architectures are complex (superscalar, out-of-order execution, multicore)
- Compilers are needed to bridge this *semantic gap*
  - Compilers let programmers write in high level languages and still get good performance on complex architectures

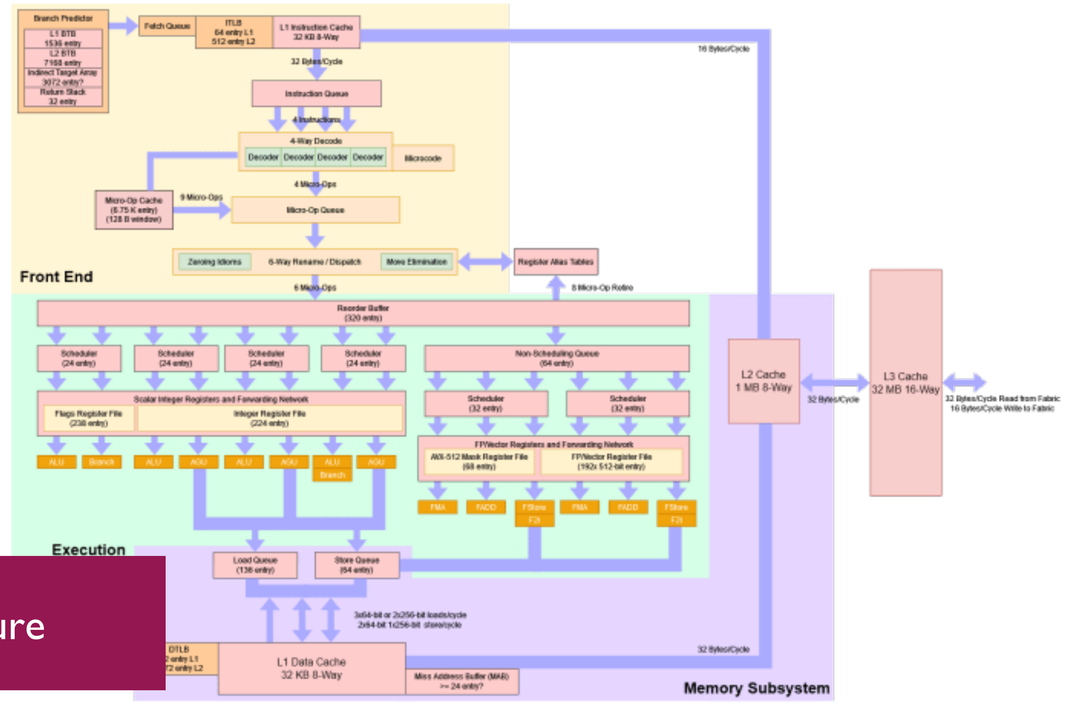
semantic gap

”Generate code for a ‘Buy’ button supported by Shopify.”

ChatGPT prompt

semantic gap

AMD Zen 4 architecture



next: what are the types of compilers?

Or: What are typical translations that compilers do?