COMP 520 - Compilers

Spring 2024, TuTh 3:30PM – 4:45PM

Instructor: Syed Ali

Course website: http://cs.unc.edu/~swali/comp520

Course Website

http://cs.unc.edu/~swali/comp520

- Lectures, Assignments, and Due Dates will be posted there
- Syllabus

What is a compiler?

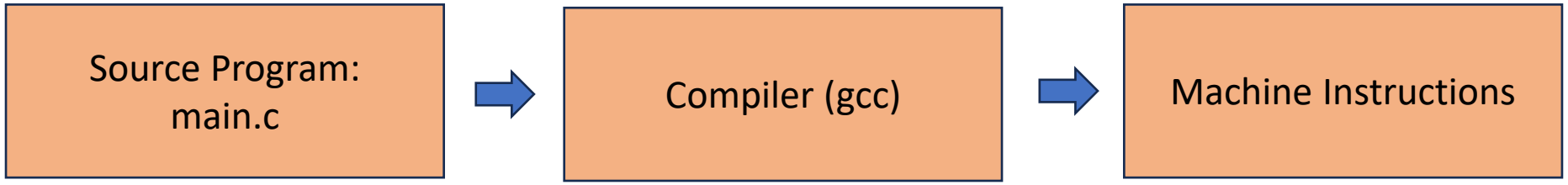
What is a compiler?

- Input: Source code written in a programming language accepted by the compiler

- Output: A representation of that source code
  - Often more compact byte-code that can be executed on a processor

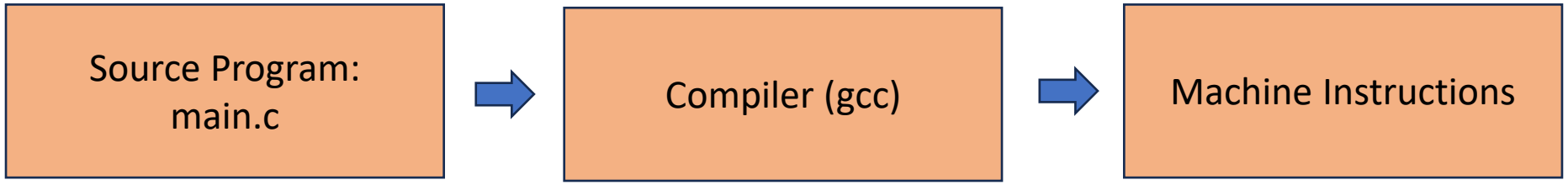
Compiler Example

Consider a C program:



Compiler Example (2)

Consider a C program:



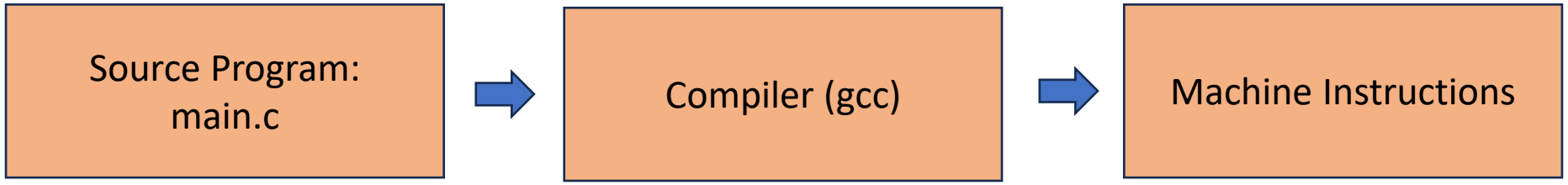

```
void main() {
   printf("Hello World");
}
```

```
mov edx,12
mov ecx,BE9E80DE
mov ebx,1
mov eax,4
int 0x80
ret
```

```
BA0C000000B9DE809EBE
BB01000000B804000000
CD80C3
```

Compiler Example (3)

Consider a C program:




```
void main() {
   printf("Hello World");
}
```

```
mov edx,12
mov ecx,BE9E80DE
mov ebx,1
mov eax,4
int 0x80
ret
```

```
BA0C000000B9DE809EBE
BB01000000B804000000
CD80C3
```

Where did this come from?

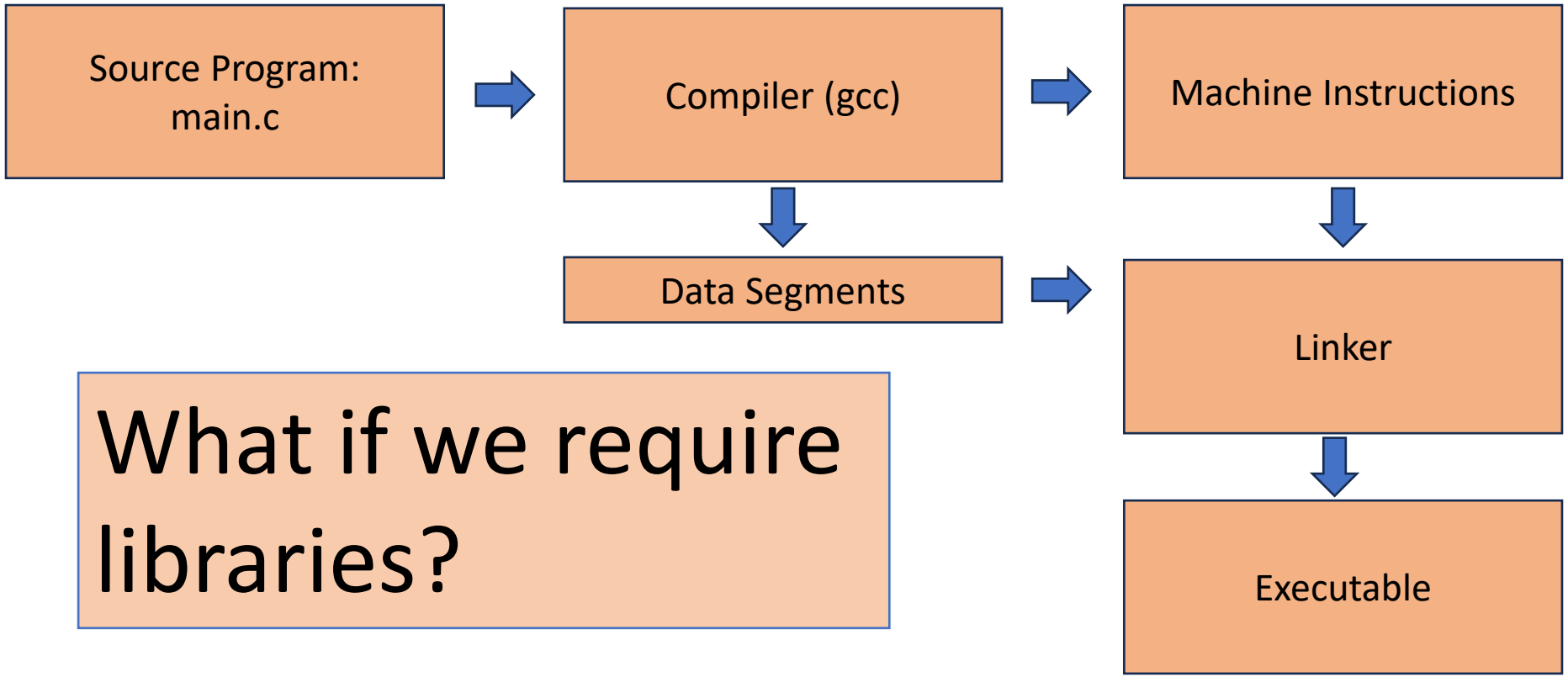
More steps needed

- Different parts of our program
- Code, data, etc.

- Need to combine them!

Compiler Example (4)

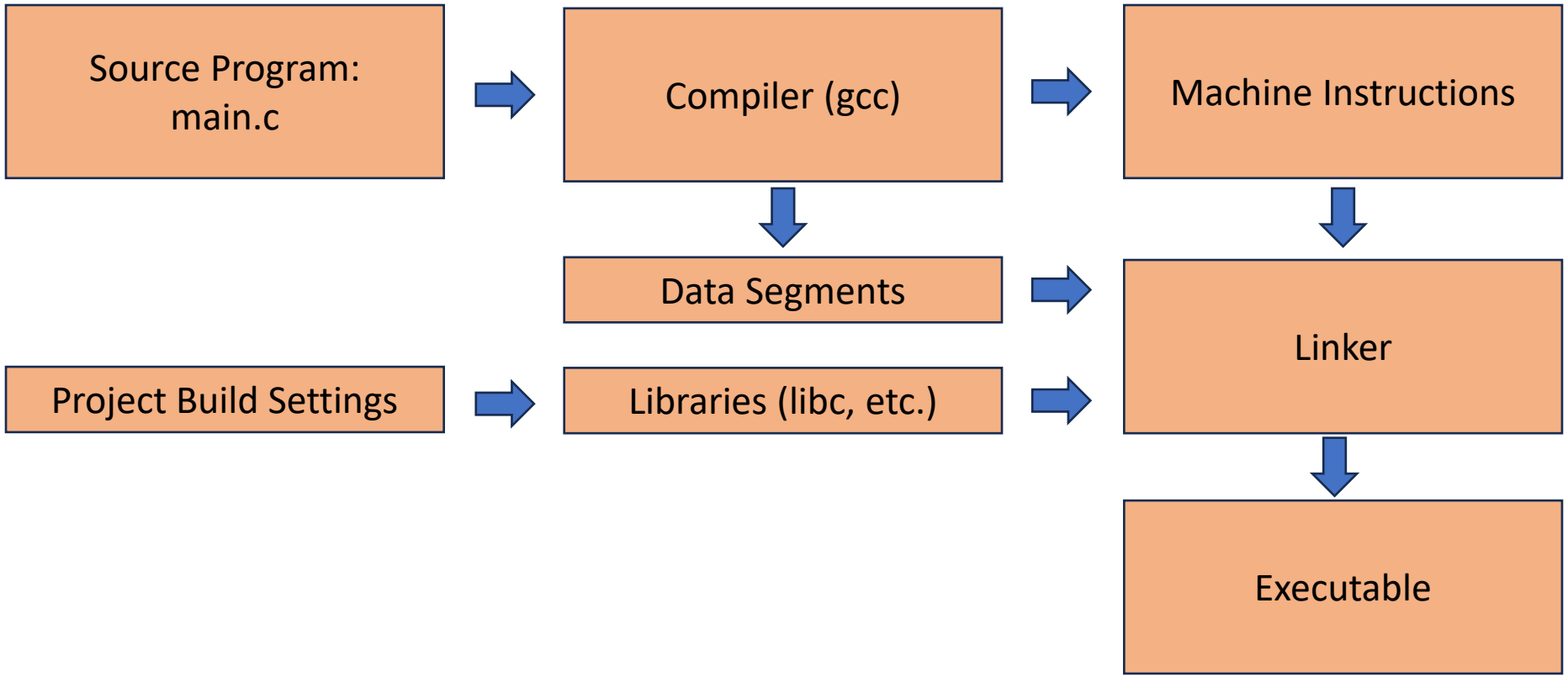
Consider a C program:



What if we require libraries?

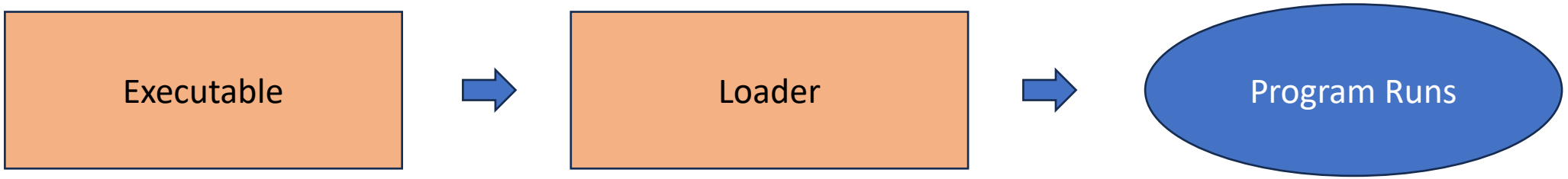
Compiler Example (5)

Consider a C program:



Execution

- The OS's loader takes the executable as an input, then ensures the program is configured correctly before it starts running.



Compiler Programming Project

Programming Project

- Goal: Build a compiler that accepts a large subset of Java and outputs a binary file that will run *natively* on modern Linux.

Programming Project (2)

- Goal: Build a compiler that accepts a large subset of Java and outputs a binary file that will run *natively* on modern Linux.

- Java normally compiles for the JVM and is cross-platform.
- We will target x86 Linux instead.

Programming Project (3)

- Goal: Build a compiler that accepts a large subset of Java and outputs a binary file that will run *natively* on modern Linux.

- Java normally compiles for the JVM and is cross-platform.
- We will target x86 Linux instead.

- Your project will compile code and the output will be runnable on modern x86 machines!

Programming Project (4)

- The compiler project will also do some small degree of linking to generate executable files.

- Gradescope can be used to test your project.

Milestones

- Five milestones, each associated with a Programming Assignment (or PA for short)

- Each milestone is worth 12% of your total grade!

Milestones (2)

- Five milestones, each associated with a Programming Assignment (or PA for short)
- Each milestone is worth 12% of your total grade!

- PA1- Syntactic Analysis
- PA2- ASTs
- PA3- Contextual Analysis
- PA4- Code Generation
- PA5- Final Submission

Milestones (3)

- PA1- Syntactic Analysis
- Scan and interpret the source code as Tokens
- Syntax: some tokens expect more tokens, was the correct token given?

- **PA1- Syntactic Analysis**
- PA2- ASTs
- PA3- Contextual Analysis
- PA4- Code Generation
- PA5- Final Submission

Token	Token Type
>	Relational operator
&&	Logical operator
y	Identifier

Milestones (4)

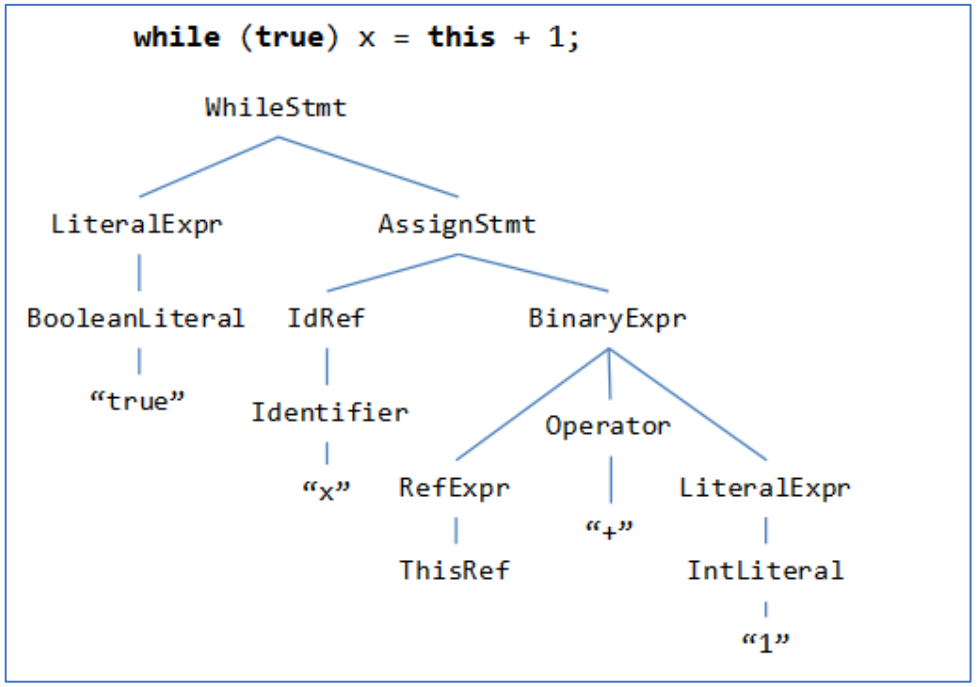
- PA2- Abstract Syntax Trees

```
while( true ) x = this + 1;
```

- PA1- Syntactic Analysis
- **PA2- ASTs**
- PA3- Contextual Analysis
- PA4- Code Generation
- PA5- Final Submission

Milestones (5)

- PA2- Abstract Syntax Trees



- PA1- Syntactic Analysis
- **PA2- ASTs**
- PA3- Contextual Analysis
- PA4- Code Generation
- PA5- Final Submission

Milestones (6)

- PA3- Contextual Analysis

- Context is important.
- Variable can only be used after it is declared.
- Type-checking

- PA1- Syntactic Analysis
- PA2- ASTs
- **PA3- Contextual Analysis**
- PA4- Code Generation
- PA5- Final Submission

Milestones (7)

- PA4- Code Generation
- Generate ELF files
- Generate simple x86
- Your compiler now generates bytecode that will run on real hardware!

- PA1- Syntactic Analysis
- PA2- ASTs
- PA3- Contextual Analysis
- **PA4- Code Generation**
- PA5- Final Submission

Milestones (8)

- PA5- Documentation & Extra Credit Opportunities

- Properly document your compiler.
- Add features for extra credit.

- PA1- Syntactic Analysis
- PA2- ASTs
- PA3- Contextual Analysis
- PA4- Code Generation
- **PA5- Final Submission**

Why Compilers?

- Understand high-level language.
- Understand compiler error and warning messages.
- Have an end-to-end understanding starting from your source code to it running on a modern device.
- Understand programming language features and complexities.

- Optimization has plenty of currently unsolved problems.
- Will be able to create a compiler optimized for a special purpose!

Written Assignments (WAs)

Written Assignments

- Five written assignments submitted on Gradescope.
- Each is worth 1% of your total grade.

Written Assignments (2)

- Five written assignments submitted on Gradescope.

- Each is worth 1% of your total grade.
- Late Policy: 10% penalty to the assignment's grade every day it is late. The number of days late is rounded up.

Programming Assignment Late Policy

- 15% penalty for every day that it is late.
- Not accepted at all if more than 3 days late.

- Full autograder is released after 3 days to ensure errors are resolved before working on the next checkpoint.

Collaboration

Collaboration

Written Assignments

- List your collaborators at the top of the submission.
- Everyone must write their answers in their own words.

Programming Assignments

Collaboration (2)

Written Assignments

- List your collaborators at the top of the submission.
- Everyone must write their answers in their own words.

Programming Assignments

- No collaboration.
- Discuss overarching theory only.
- Do not discuss code, organization, or solutions.
- Can obtain help during office hours.
- Should be considered an independent assignment.

Grading

Grading

Details

The point distribution of how your grade is calculated is shown on the right.

Syllabus

- Project: 60%
- Final Exam: 15%
- Midterm 1: 7.5%
- Midterm 2: 7.5%
- Written Assignments: 5%
- Participation: 5%

Grading

Details

- The midterms are in-person.
- 75 minutes.

- Dates (Subject to change)
  - Midterm 1: 2/22/24
  - Midterm 2: 4/2/24

Syllabus

- Project: 60%
- Final Exam: 15%
- **Midterm 1: 7.5%**
- **Midterm 2: 7.5%**
- Written Assignments: 5%
- Participation: 5%

Grading

Details

- Participation is required.
- Participation isn't necessarily in-class but during office hours as well.
- I highly encourage attending office hours.

Syllabus

- Project: 60%
- Final Exam: 15%
- Midterm 1: 7.5%
- Midterm 2: 7.5%
- Written Assignments: 5%
- **Participation: 5%**

Use of Technology

- I do allow the use of laptops in the classroom
- Limit what you use your laptop for
  - Course material
  - Searching for course-related material

Any Questions?

- Remember to check the syllabus!
- The course website will be updated with the latest information.

End