Welcome to CS 1342

Programming Concepts C++

Today’s Agenda

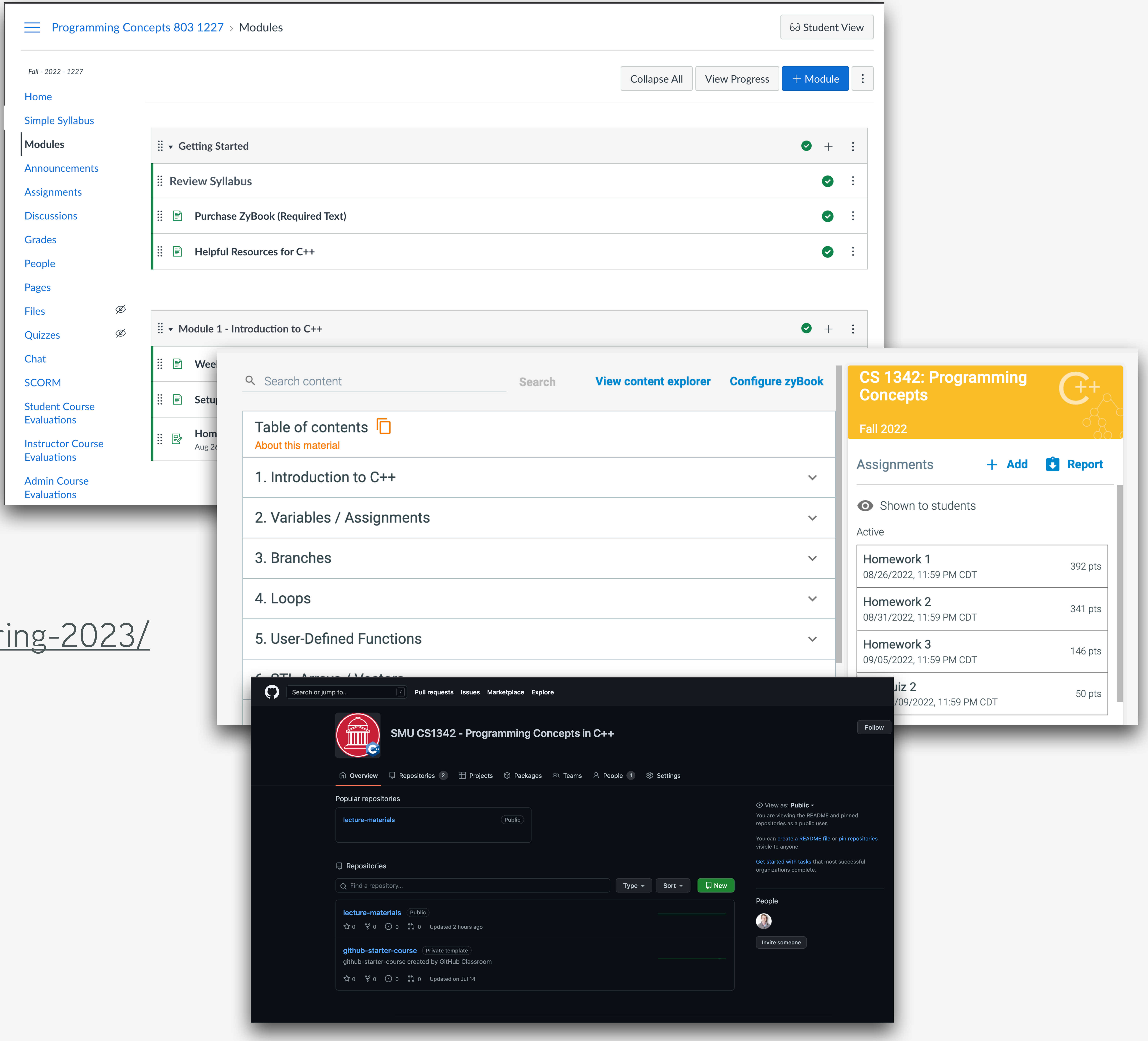
- Syllabus Review
- Class Expectations
- Brief History of C++
- Chapter 01

Review Syllabus

Class Expectations

What do you need?

- Canvas
  - Lecture materials (slides)
  - Schedule
  - Assignments
  - Other Resources
  - Announcements
  - Grades
- ZyBook - required online textbook
- GitHub - https://github.com/SMU-CS1342-Spring-2023/



Getting Started

Review Syllabus

Purchase ZyBook (Required Text)

Helpful Resources for C++

Module 1 - Introduction to C++

Wee

Setu

Hom
Aug 2

Table of contents
About this material

1. Introduction to C++
2. Variables / Assignments
3. Branches
4. Loops
5. User-Defined Functions
6. STL Arrays / Vectors

CS 1342: Programming Concepts
Fall 2022

Assignments + Add Report

Shown to students

Active

Homework 1 08/26/2022, 11:59 PM CDT	392 pts
Homework 2 08/31/2022, 11:59 PM CDT	341 pts
Homework 3 09/05/2022, 11:59 PM CDT	146 pts
uiz 2 /09/2022, 11:59 PM CDT	50 pts

SMU CS1342 - Programming Concepts in C++

Popular repositories

lecture-materials Public

Repositories

lecture-materials Public
☆ 0 0 0 0 Updated 2 hours ago

github-starter-course Private template
github-starter-course created by GitHub Classroom
☆ 0 0 0 0 Updated on Jul 14

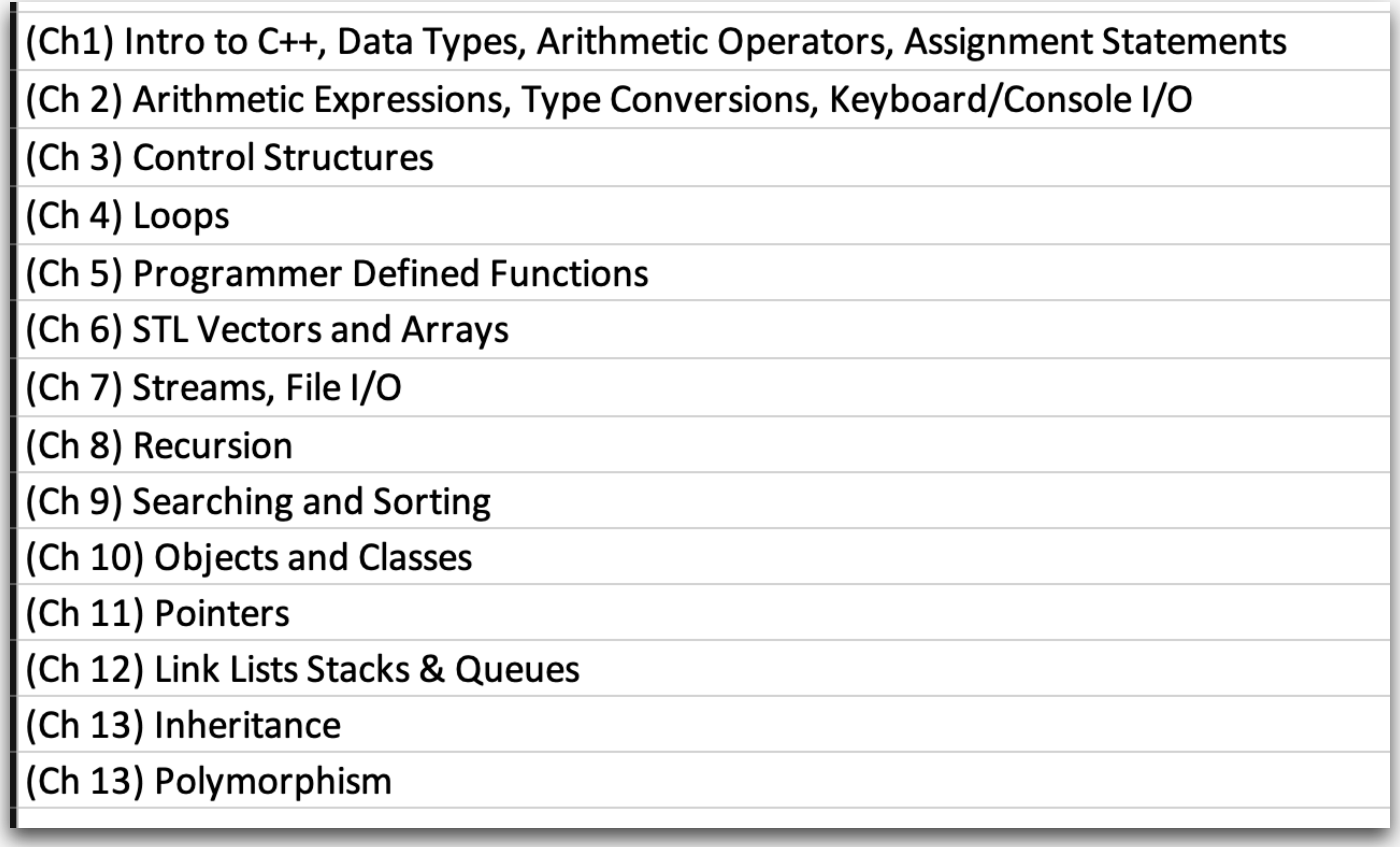
View as: Public
You are viewing the README and pinned repositories as a public user.

You can create a README file or pin repositories visible to anyone.

Get started with tasks that most successful organizations complete.

People

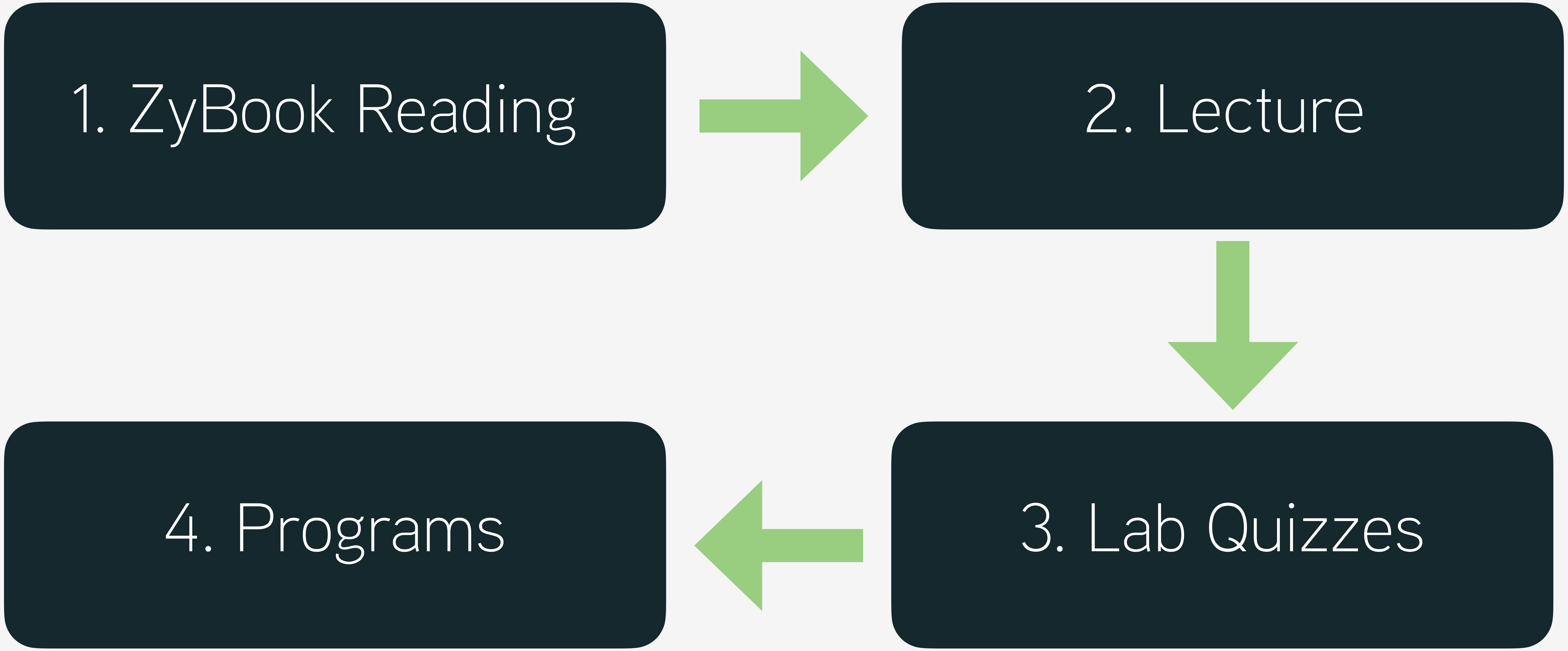
High Level Schedule



(Ch1) Intro to C++, Data Types, Arithmetic Operators, Assignment Statements
(Ch 2) Arithmetic Expressions, Type Conversions, Keyboard/Console I/O
(Ch 3) Control Structures
(Ch 4) Loops
(Ch 5) Programmer Defined Functions
(Ch 6) STL Vectors and Arrays
(Ch 7) Streams, File I/O
(Ch 8) Recursion
(Ch 9) Searching and Sorting
(Ch 10) Objects and Classes
(Ch 11) Pointers
(Ch 12) Link Lists Stacks & Queues
(Ch 13) Inheritance
(Ch 13) Polymorphism

Learning Strategy

For each new topic:



1. ZyBook Reading

2. Lecture

3. Lab Quizzes

4. Programs

Keys to success

- All programming classes require a LOT of time outside of class. Make sure you stay on top of readings and assignments and budget at least **10 hours** for each Programming Assignment
- **Program Assignments are meant to challenge you.** Spend time understanding the problem ***before jumping into the code***
- ***Attendance is mandatory for this class.***
  - ***If you must miss class for a valid reason, communicate with me ahead of time and make plans to collect notes from a classmate.***
- ***Utilize Office Hours! I will schedule office hours sessions for each program to help individuals through debugging and talk through strategy.***

Things to avoid

- Skipping class
- Collaborating on Lab or Programming assignments
- Asking for makeup quiz, exam, or lab extension (unless you have an official university absence)

Lastly…

Have Fun!!

Next steps!

- Register for ZyBook
- Homework 1 - Chapter 1 & 2
- Homework 2 - Chapter 3 & 4
- Environment Setup (Instruction in canvas)
  - C++ Compiler
  - CLion IDE
  - GitHub Account
- Bookmark our course GitHub page!

Overview of C++

Background and History

Why Study Computer Science?

- Computer Science teaches you how to be a **problem solver.**
- **CS Majors** are the most sought after Engineering position in industry
- The average starting salary in Dallas, TX: **$75,783**
- Learning a programming language opens the door for many things
    - Machine Learning and AI
    - Networking
    - Security Engineer
    - Blockchain and Cryptocurrency
    - Web / Mobile Development
    - Quantum Computing
- What does a day in the life of a Software Developer look like?

What is C++?

C++ is a **statically typed**, general-purpose programming language that supports **Object Oriented Programming** and **memory control** / manipulation.

- Considered a “Middle Level” Language
- 4rth Most Popular Language (IEEE 2016)
- C++ is compiled directly down to machine code, as opposed to Java which is compiled into Java Byte code (interpreted by the JVM)

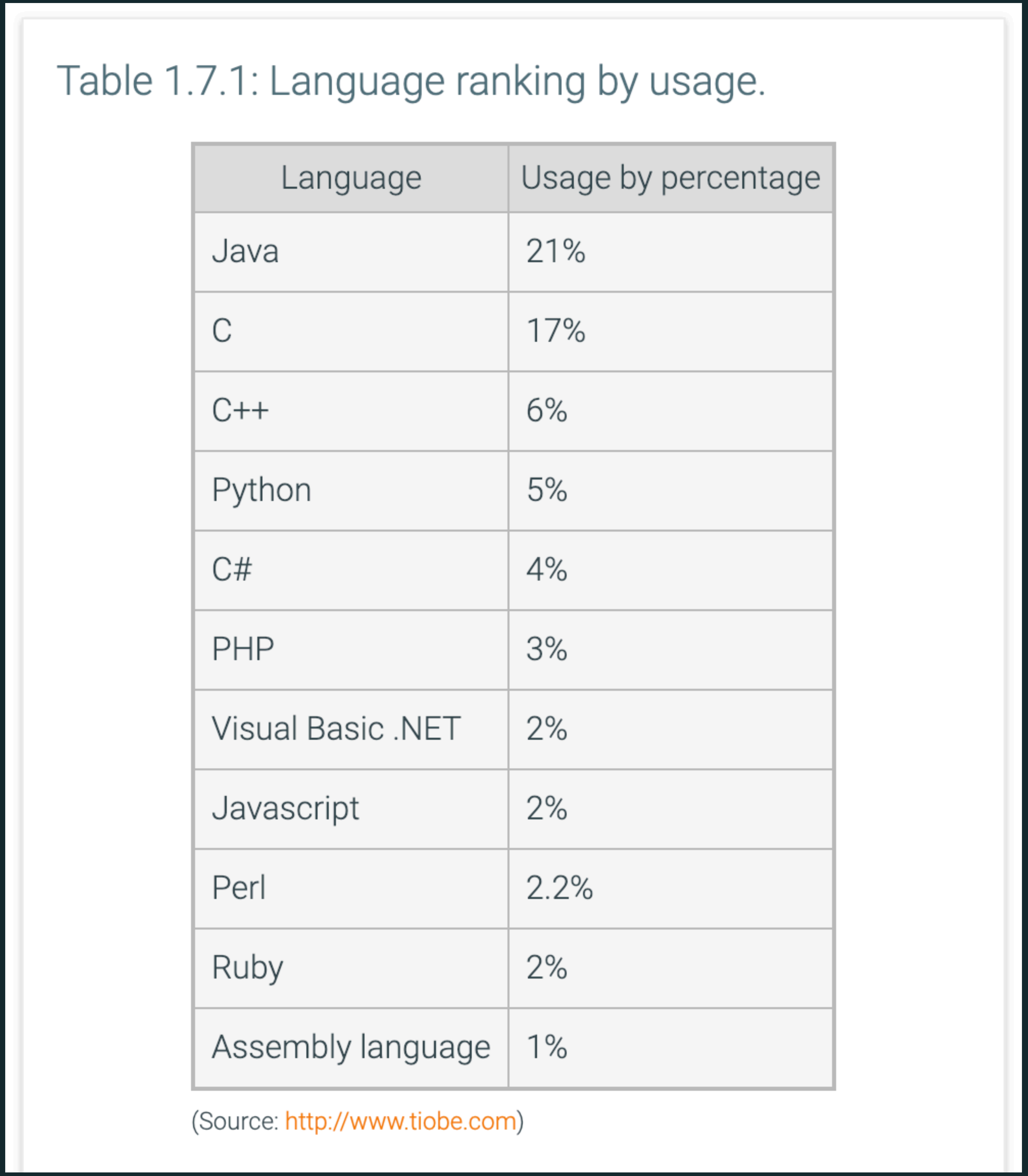


Table 1.7.1: Language ranking by usage.

Language	Usage by percentage
Java	21%
C	17%
C++	6%
Python	5%
C#	4%
PHP	3%
Visual Basic .NET	2%
Javascript	2%
Perl	2.2%
Ruby	2%
Assembly language	1%

(Source: http://www.tiobe.com)

Who uses C++?

- Applications
- Operating Systems (MacOS, Linux, Microsoft, etc…)
- Cloud / Distributed Systems (Think AWS, Azure, or GCP)
- Games
- Embedded Systems and IOT Devices
- Some apps using C++ that you may know:
  - Adobe Photoshop
  - Spotify
  - YouTube
  - Amazon
  - Microsoft Office Suite Applications
  - Google Chrome and Google Search
  - Bloomberg Financial Data
  - Facebook

Major Versions of C++

- **C++98** - Released in 1998 and released the STL (Standard Template Library)
- **C++11** - Released in 2011. Added improvements to the STL including many BOOST Libraries
- **C++14** - Released 2014 - Introduction of polymorphic lambdas, digit separators, generalized lambda capture, variable templates, binary integer literals, quoted strings etc.
- **C++17** - Released 2017 - Introduction of fold expressions, hexadecimal floating point literals, a u8 character literal, selection statements with initializer, inline variables etc.
- **C++20** - Released in 2020. Will include Concepts, Coroutines, and many other features.

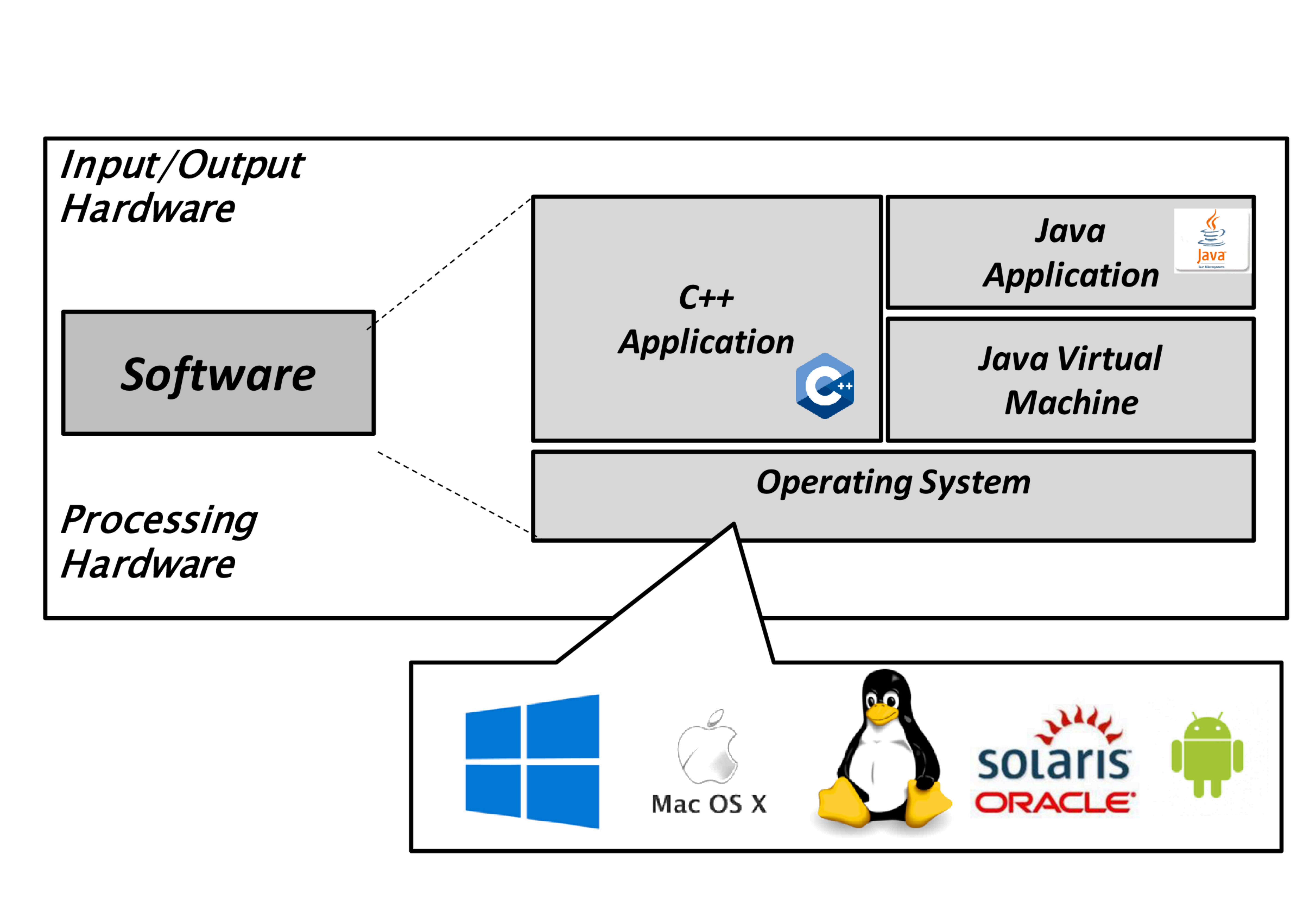
In this class we will focus mostly on C++98 and C++11 features.

Chapter 1: Introduction to C++

Console I/O and Operators

C++ vs Java

- Both ***statically*** typed
- Portability
  - Java uses the JVM
  - C++ compiles directly to byte code
- Memory Management
  - Java - JVM manages memory
  - C++ - user must manage memory



Java vs C++

- Basic unit of code:
  - Java - a **class**
  - C++ - a **function**

C++

```cpp
#include <iostream>

using namespace std;

int main() {
  cout << "Hello World!" << endl;

  return 0;
}
```

Java

```java
public class HelloWorld {

    public static void main(String[] args) {
        System.out.println("Hello, World");
    }

}
```

C++ Programming Basics

- Every C++ program starts in **main()**
- Each statement is followed with a semicolon
- **main()** returns 0
    - A non-zero return notifies the operating system that something went wrong

```
#include <iostream>

int main() {

  // code goes here

  return 0;
}
```

Comments

- Comments are ***extremely*** important in all programming languages. It is especially important in C++ where your source code can be tricky to understand.
- All programs should include the following comments:
  - A comment on the top of any file
  - Above a function implementation
  - In line

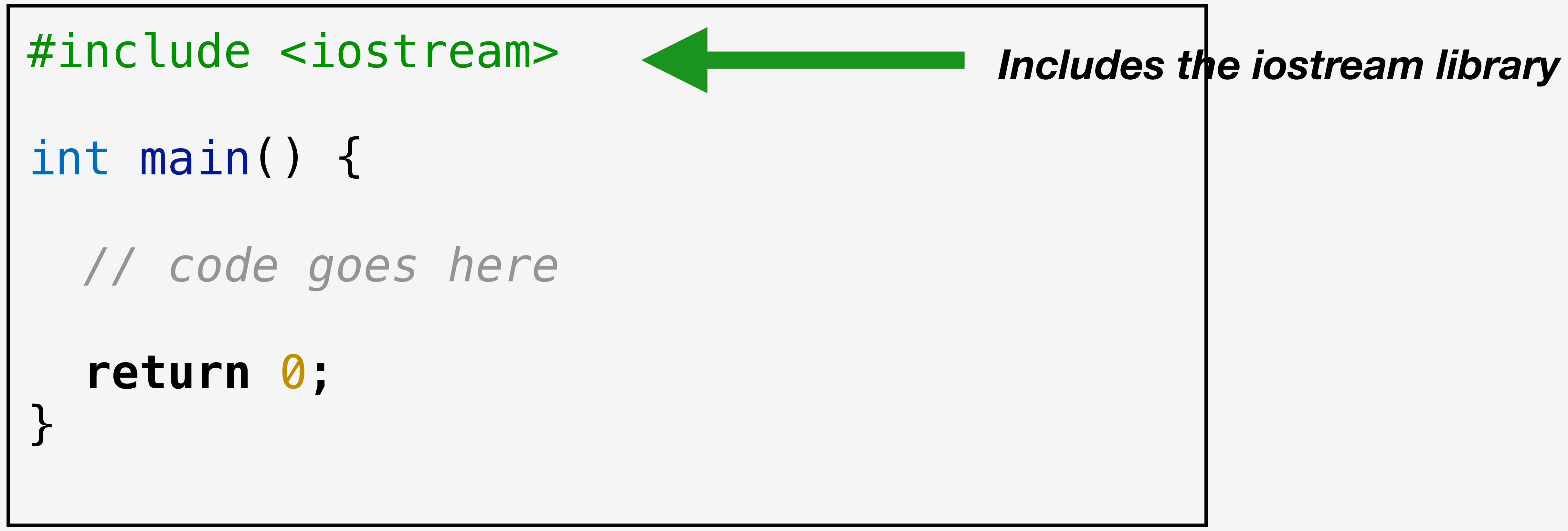
```
// Inline Comment

/**
 * This is a multiline comment
 *
 */
```

Header files

In C++, we often times need to include other libraries or files into our program so that we can have access to additional functionality

- The ***#include*** keyword is used to indicate importing a file into our program (similar to Java import)
- This gives you access to things implemented in the c++ standard library
  - Ex: ***strings, cout, cin,*** etc…



```
#include <iostream>

int main() {

  // code goes here

  return 0;
}
```

Includes the iostream library

Console I/O - ***cout***

cout is part of the ***iostream*** c++ standard library. To use cout in a program you must include the header ***#include<iostream>***

- `cout` - characters out - prints characters to console
  - In Java this was System.out.print(…);
- **`<<`** - the stream insertion operator
- String Literal “” ex: “cat”
  - A string ***literal*** is different than a string in that it does NOT have a memory address.
- `endl` - denotes a new line
- `cout` - an object responsible for printing out to the console

Console I/O - ***cin***

cin is part of the ***iostream*** c++ standard library. To use cin in a program you must include the header ***#include<iostream>***

- **`cin`** - characters in - reads in characters entered from the keyboard
  - In Java this was where you used Scanner
- **`>>`** - the stream extraction operator
- Rules to be aware of with ***cin***
  - When reading into an integer - will ignore leading whitespace and read until the first non-decimal character
  - When reading into a floating point - will ignore leading whitespace then read until the first non-decimal character (will include decimal place)
  - When reading into a single character - will ignore leading whitespace then read first character

Console I/O Exercise

What will the output look like for each statement?

```
int score = 20;

1. cout << “George’s score is” << score;
  cout << “Elroy’s score is” << score;


2. cout << “Welcome \n to Dunder Mifflin”;
```

Console I/O Exercise

What will the output look like for each statement?

```
int score = 20;

1. cout << “George’s score is” << score;
  cout << “Elroy’s score is” << score;
```

```
George’s score is20Elroy’s score is20
```

```
2. cout << “Welcome \n to Dunder Mifflin”;
```

```
Welcome
 to Dunder Mifflin
```

Console I/O Exercise

```
int a;
char b;
float c;

cin >> a >> b >> c;
cout << a << endl << b << endl << c;
```

What would be printed if the user entered the following:

- 123b13.2
- 129.11.129.498