CS 254: Concurrent Object-Oriented Programming with Java & Android

Course Overview & Logistics

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science
Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA
Learning Objectives in this Lesson

• Understand the course topics & logistics
  • Course philosophy
  • Course contents
  • Structure of the lecture material
  • Overview of the assignments & assessments
• Setting up the Java & Android IDE on Android Studio
• Setting up GitLab et al.
• Accessing Android & Java source code
Course Philosophy
There's a growing need for software developers who know how to write concurrent programs for a range of computing platforms, e.g., mobile devices, laptops, desktops, & cloud environments.

The topics covered in this course apply to many platforms, not just Android.
Course Philosophy

• Demand is driven by advances in software & hardware infrastructure
  • e.g., multi-core & many core processors, mass storage, ubiquitous network connectivity, & commodity hardware & software platforms

See www.gotw.ca/publications/concurrency-ddj.htm
Concurrent Programming

- Concurrency is also commonly used to offload work from the user interface (UI) thread to background thread(s) in mobile devices.

See [developer.android.com/training/multiple-threads/communicate-ui.html](http://developer.android.com/training/multiple-threads/communicate-ui.html)
Course Philosophy

- Effective techniques & practices for designing & programming concurrent (mobile) apps are *not* best learned through generalities & platitudes

[Image: KEEP CALM AND [insert platitude]]
Course Philosophy

- Effective techniques & practices for designing & programming concurrent (mobile) apps are *not* best learned through generalities & platitudes.

“Sitting & thinking” is not sufficient...
Course Philosophy

• Instead, it’s better to see by example how concurrent programs can be made
  • easier to write & read,
  • easier to maintain & modify,
  • more efficient & resilient
by applying time-proven software patterns & object-oriented (& some functional) design & programming techniques

This course involves lots of hands-on software development & testing!
Summary of the Course Contents
Summary of Course Contents

- Coverage of foundational Java concurrency mechanisms

Summary of Course Contents

- Coverage of foundational Java concurrency mechanisms
  - e.g., Java threading & synchronizer mechanisms
Summary of Course Contents

- Coverage of foundational Java concurrency mechanisms
  - e.g., Java threading & synchronizer mechanisms

**Threading mechanisms**
- Thread
- Runnable
- Executor
- ExecutorService
- ScheduledExecutorService
- ExecutorCompletionService
- Future
- FutureTask
- ThreadPoolExecutor
- ForkJoinPool

**Synchronizer mechanisms**
- Synchronized methods & statements
- Notification methods
- ReentrantLock
- ReentrantReadWriteLock
- StampedLock
- ConditionObject
- Semaphore
- CountDownLatch
- CyclicBarrier
- Phaser

We provide roughly equal focus on Java synchronization mechanisms & Java threading mechanisms in this course.
Summary of Course Contents

- Coverage of foundational Java concurrency mechanisms
- Patterns/frameworks for concurrent programming

See [en.wikipedia.org/wiki/Concurrency_pattern](en.wikipedia.org/wiki/Concurrency_pattern)
Summary of Course Contents

• Coverage of foundational Java concurrency mechanisms
• Patterns/frameworks for concurrent programming
• We assume you know (or can quickly learn) Android, Android Studio, modern Java, & Git

See item #12 at github.com/douglasraugschmidt/CS254/wiki/CS-254-FAQ
Structure of the Lecture Material
This course has three main topics

<table>
<thead>
<tr>
<th>Section</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Threading</td>
<td>• Coverage of basic &amp; advanced Java threading mechanisms, e.g.</td>
</tr>
<tr>
<td></td>
<td>• Java Threads &amp; Runnables</td>
</tr>
<tr>
<td></td>
<td>• The Java Executor framework</td>
</tr>
</tbody>
</table>
Structure of the Lecture Material

- This course has three main topics

<table>
<thead>
<tr>
<th>Section</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Threading</td>
<td>• Coverage of basic &amp; advanced Java threading mechanisms, e.g.</td>
</tr>
<tr>
<td></td>
<td>• Java Threads &amp; Runnables</td>
</tr>
<tr>
<td></td>
<td>• The Java Executor framework</td>
</tr>
<tr>
<td>Java Synchronization</td>
<td>• Coverage of basic &amp; advanced Java synchronization mechanisms, e.g.,</td>
</tr>
<tr>
<td></td>
<td>• Build-in monitor objects</td>
</tr>
<tr>
<td></td>
<td>• Myriad synchronizer classes in java.util.concurrent</td>
</tr>
</tbody>
</table>
Structure of the Lecture Material

• This course has three main topics

<table>
<thead>
<tr>
<th>Section</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Threading</td>
<td>• Coverage of basic &amp; advanced Java threading mechanisms, e.g.</td>
</tr>
<tr>
<td></td>
<td>• Java Threads &amp; Runnables</td>
</tr>
<tr>
<td></td>
<td>• The Java Executor framework</td>
</tr>
<tr>
<td>Java Synchronization</td>
<td>• Coverage of basic &amp; advanced Java synchronization mechanisms, e.g.,</td>
</tr>
<tr>
<td></td>
<td>• Build-in monitor objects</td>
</tr>
<tr>
<td></td>
<td>• Myriad synchronizer classes in java.util.concurrent</td>
</tr>
<tr>
<td>Software Patterns</td>
<td>• Concurrency patterns</td>
</tr>
</tbody>
</table>
# Structure of the Lecture Material

- This course has three main topics

<table>
<thead>
<tr>
<th>Section</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Threading</td>
<td>• Coverage of basic &amp; advanced Java threading mechanisms, e.g.</td>
</tr>
<tr>
<td></td>
<td>• Java Threads &amp; Runnables</td>
</tr>
<tr>
<td></td>
<td>• The Java Executor framework</td>
</tr>
<tr>
<td>Java Synchronization</td>
<td>• Coverage of basic &amp; advanced Java synchronization mechanisms, e.g.,</td>
</tr>
<tr>
<td></td>
<td>• Build-in monitor objects</td>
</tr>
<tr>
<td></td>
<td>• Myriad synchronizer classes in java.util.concurrent</td>
</tr>
<tr>
<td>Software Patterns</td>
<td>• Concurrency patterns</td>
</tr>
</tbody>
</table>

We’ll bounce around when covering these topics to facilitate assignments
Structure of the Lecture Material

- This course has three main topics
  - Each topic is composed of lessons
Structure of the Lecture Material

- This course has three main topics
  - Each topic is composed of lessons
  - Each lesson is composed of parts
Structure of the Lecture Material

- This course has three main topics
  - Each topic is composed of lessons
  - Each lesson is composed of parts
  - Each part is a single lecture

Screencasts of each lesson “part” & PDF versions of the slides will be uploaded to [www.dre.vanderbilt.edu/~schmidt/cs254#lectures](http://www.dre.vanderbilt.edu/~schmidt/cs254#lectures)
Structure of the Lecture Material

- This course has three main topics
  - Each topic is composed of lessons
  - Each lesson is composed of parts
  - Each part is a single lecture
    - Each part is composed of segments
Structure of the Lecture Material

- There will be periodic tests on material covered in the lectures
Structure of the Lecture Material

- There will be periodic tests on material covered in the lectures.
- All tests (including the final) are "closed book," "closed Internet," "closed electronics," etc.

1st test quiz will be on Wednesday, January 26th via Brightspace
There will be periodic tests on material covered in the lectures.

- All tests (including the final) are “closed book,” “closed Internet,” “closed electronics,” etc.
- We’ll try to grade & review the tests by the next class.

One of the benefits of a smaller class ;-)
Structure of the Lecture Material

• There will be periodic tests on material covered in the lectures
  • All tests (including the final) are “closed book,” “closed Internet,” “closed electronics,” etc.
  • We’ll try to grade & review the tests by the next class

I recommend that you study for tests by reviewing slides & watching screencasts available at www.dre.vanderbilt.edu/~schmidt/cs254#lectures
Structure of the Lecture Material

• There will be periodic tests on material covered in the lectures
  • All tests (including the final) are "closed book," "closed Internet," "closed electronics," etc.
  • We’ll try to grade & review the tests by the next class
    • If you don’t attend the next class & don’t get your quiz you will be penalized 50%

See [www.dre.vanderbilt.edu/~schmidt/cs254/work-summary.html#quizzes](www.dre.vanderbilt.edu/~schmidt/cs254/work-summary.html#quizzes)
Structure of the Lecture Material

• There will be periodic tests on material covered in the lectures
  • All tests (including the final) are “closed book,” “closed Internet,” “closed electronics,” etc.
  • We’ll try to grade & review the tests by the next class
    • If you don’t attend the next class & don’t get your quiz you will be penalized 50%
    • Likewise, if you just show up for the test & don’t attend class you’ll be penalized 50%

See [www.dre.vanderbilt.edu/~schmidt/cs254/work-summary.html#quizzes](http://www.dre.vanderbilt.edu/~schmidt/cs254/work-summary.html#quizzes)
Structure of the Lecture Material

- There may be a final exam that can cover all the lectures
- The focus will be on the last week(s) of the semester

The final exam is noon to 3pm, Monday, May 2\textsuperscript{nd} via Brightspace

Keep Calm and Good Luck on Final Exams
Overview of the Assignments & Assessments
Overview of Assignments & Assessments

• Programming assignments should be written in Java 11 using Android Studio

You can use any IDE, but your final submission must build/run with the latest Android Studio & Android 11 “R” (API level 30)
Overview of Assignments & Assessments

- Programming assignments should be written in Java 11 using Android Studio
- The Java 11 runtime environment (JRE) comes pre-installed with Android now

You can use cool Java 11 features starting with Android Studio “Arctic Fox”
Overview of Assignments & Assessments

• Android programming assignments must be submitted using Android Studio

- A wizard for creating new apps
- A visual editor for creating GUIs
- An editor for manipulating Android XML descriptors needed for your app
- An emulator for testing your apps on your PC
- A debugger for finding errors in the emulator or on a device

See developer.android.com/sdk
Overview of Assignments & Assessments

• Android programming assignments must be submitted using Android Studio
• Please install Android 11 “R” (API level 30)

See en.wikipedia.org/wiki/Android_11
Overview of Assignments & Assessments

- All source code for assignments & examples available at GitHub

Go to GitHub at github.com/douglasraigschmidt/CS254
Overview of Assignments & Assessments

- All source code for assignments & examples available at GitHub
- You will need to learn how to use GitLab et al.

See item #13 at github.com/douglascraigschmidt/CS254/wiki/CS-254-FAQ
Overview of Assignments & Assessments

• All source code for assignments & examples available at GitHub
  • You will need to learn how to use GitLab et al.

• Be prepared to update your repositories multiple times
  • i.e., you need to understand Git!

“If you don’t like change, you’re going to like irrelevance even less.”
Overview of Assignments & Assessments

- Assignments will provide a range of experience with modern Java & Android concurrent programs

See [github.com/douglascraigschmidt/CS254](http://github.com/douglascraigschmidt/CS254)
Overview of Assignments & Assessments

• In particular, you’ll implement multiple variants of a Java concurrent resource manager & an associated Android app

See en.wikipedia.org/wiki/Palantir
Overview of Assignments & Assessments

- In particular, you’ll implement multiple variants of a Java concurrent resource manager & an associated Android app, e.g.
  - Java Thread & Runnable
  - Java Executor framework (e.g., ExecutorService, ExecutorCompletionService, & ForkJoinPool)
  - Java Semaphore, built-in monitor objects, Stamped Lock, ReentrantLock, ConcurrentHashMap, etc.

The topics covered by the assignments generalize to more than Android!
Overview of Assignments & Assessments

• Assignment assessments will be done via reviews by course staff
Overview of Assignments & Assessments

- Assignment assessments will be done via reviews by course staff
- Assignments *must* be submitted on time or you’ll get a 0

See item #4 at [github.com/douglasraigschmidt/CS254/wiki/CS-254-FAQ](https://github.com/douglasraigschmidt/CS254/wiki/CS-254-FAQ)
Overview of Assignments & Assessments

- Assignment assessments will be done via reviews by course staff.
  - Assignments must be submitted on time or you’ll get a 0.
  - Your initial submission must compile & be largely complete or you won’t get a review or a final grade.

Diagram:
- Submit Program
  - Feedback to students
    - Instructor Review
    - Revise Program
    - TA/Grader Review
    - Resubmit Program
- Final grade
Overview of Assignments & Assessments

• Assignment assessments will be done via reviews by course staff
  • Assignments *must* be submitted on time or you’ll get a 0
  • Your initial submission must compile & be largely complete or you won’t get a review or a final grade
• You will not receive a grade for assignments if you do not attend class regularly

See [www.dre.vanderbilt.edu/~schmidt/cs254/assignments.html](http://www.dre.vanderbilt.edu/~schmidt/cs254/assignments.html)
Overview of Assignments & Assessments

- Assignment assessments will be done via reviews by course staff
  - Assignments *must* be submitted on time or you’ll get a 0
  - Your initial submission must compile & be largely complete or you won’t get a review or a final grade
  - You will not receive a grade for assignments if you do not attend class regularly
- Work *must* be your own
  - This applies for quizzes & programming assignments

www.vanderbilt.edu/student_handbook/the-honor-system#statement-of-the-honor-code
Overview of Assignments & Assessments

• The bulk of your grade is based on the results of the automated unit tests

See [www.dre.vanderbilt.edu/~schmidt/cs254/assignments.html](http://www.dre.vanderbilt.edu/~schmidt/cs254/assignments.html)
Overview of Assignments & Assessments

• The bulk of your grade is based on the results of the automated unit tests

It’s important that your current assignment also passes all the tests for previous assignments!

See item #16 at [github.com/douglasraigschmidt/CS254/wiki/CS-254-FAQ](https://github.com/douglasraigschmidt/CS254/wiki/CS-254-FAQ)
Overview of Assignments & Assessments

- The bulk of your grade is based on the results of the automated unit tests

See item #17 at github.com/douglascraigschmidt/CS254/wiki/CS-254-FAQ
Overview of Assignments & Assessments

- The relative weighting of each portion of the course is:
  - 45% Quizzes
  - 40% Programming projects
  - 10% Final exam
  - 05% Participation

These weightings may change, depending on various factors.
Overview of Assignments & Assessments

• The relative weighting of each portion of the course is:
  • 45% Quizzes
  • 40% Programming projects
  • 10% Final exam
  • 05% Participation
  • Participation includes attendance, involvement, & “following directions”
Overview of Assignments & Assessments

• The relative weighting of each portion of the course is:
  • 45% Quizzes
  • 40% Programming projects
  • 10% Final exam
  • 05% Participation

  • Participation includes attendance, involvement, & “following directions”

Attendance also affects other aspects of your quiz & assignment grades

See www.dre.vanderbilt.edu/~schmidt/cs254/work-summary.html#quizzes & www.dre.vanderbilt.edu/~schmidt/cs254/assignments.html
Overview of Assignments & Assessments

• The relative weighting of each portion of the course is:
  • 45% Quizzes
  • 40% Programming projects
  • 10% Final exam
  • 05% Participation
  • Participation includes attendance, involvement, & “following directions”

Don’t expect to get an A in this class if you do not actively participate!!!!
Setting Up the Android & Java IDE on Android Studio
Installing Java/Android Developer Tools

• To use Android, you need to install the latest release of Android Studio

See developer.android.com/studio
Installing Java/Android Developer Tools

- Installation steps
Installing Java/Android Developer Tools

- Installation steps
  - Download & install the latest version of Android Studio

New features in Android Studio Preview

Warning: The JCenter repository became read-only on March 31st, 2021. For more information, see JCenter service update.

Android Studio Arctic Fox | 2020.3.1 has been released to the stable channel. Download it here.

Android Studio Bumblebee | 2021.1.1 is currently in the Canary and Dev channels.

Android Gradle plugin (AGP) 7.0 has been released to the stable channel. For more information, see the AGP release notes.

For the latest news on releases, including a list of notable fixes in each release, also see the Release updates.

If you encounter any problems using a preview version of Android Studio, please let us know. Your bug reports help to make Android Studio better.

See developer.android.com/studio
Add Components to the SDK

- Launch the Android Studio SDK Manager
- Select “R” version of Android (11, API 30)

Add Components to the SDK

- Launch the Android Studio Virtual Device Manager
- Create an Android API 30 emulator

[Image 1: Android Virtual Device Manager interface showing virtual devices and selection options]

[Developer Link: developer.android.com/tools/devices/managing-avds.html]
## Intel HAXM Virtualization Driver

### Requirements
- Intel virtualization extensions (VT, VT-x, vmx)
- AMD virtualization extensions (AMD-v, SVM) [only supported on Linux]
- Download an x86 emulator image

### Windows & Mac OSX
- `<sdk>/extras/intel/Hardware_Accelerated_Execution_Manager/IntelHAXM.exe/dmg`

### Linux
- Install KVM & pass “-enable-kvm” flag to emulator when starting

---

[developer.android.com/studio/run/emulator-acceleration](developer.android.com/studio/run/emulator-acceleration)
Setting Up GitLab et al.
Setting Up GitLab et al.

1. Create Your GitLab Repo
www.gitlab.com

```
git clone git@gitlab.com:your-name/CS-254-spring-2022.git
```

2. Clone your GitLab repo

```
Working Folder (Student’s)
```

3. Change Director into Your Working Folder

```
cd CS-254-spring-2022
```

See [docs.gitlab.com/ee/ssh](docs.gitlab.com/ee/ssh) for info on setting up an SSH key for GitLab et al.
Setting Up GitLab et al.

1. Update from Read-Only GitHub Repo
   git remote add skeletons
   git@github.com:douglasraignschmidt/CS254.git

2. Get Current Version
   git pull skeletons master

3. Do work!

Working Folder (Student’s)
   Assignment1a
   ...

Local Repo (Student’s)
   Assignment1a
   ...

4. Commit Changes
   git commit

GitLab Repo (Student’s)
   Assignment1a
   ...

5. Send Changes to GitLab Repo
   git push origin master

GitHub Repo (Instructor’s)
   Assignment1a
   Assignment1b
   Assignment2a
   Assignment2b
   ...

See item #13 at [github.com/douglasraignschmidt/CS254/wiki/CS-254-FAQ](https://github.com/douglasraignschmidt/CS254/wiki/CS-254-FAQ)
Setting Up GitLab et al.

Welcome to GitLab
Code, test, and deploy together

Create a project
Projects are where you store your code, access issues, wiki and other features of GitLab.

Create a group
Groups are the best way to manage projects and members.

Explore public projects
There are 1,385,678 public projects on this server. Public projects are an easy way to allow everyone to have read-only access.

Learn more about GitLab
Take a look at the documentation to discover all of GitLab's capabilities.
### New project

A project is where you house your files (repository), plan your work (issues), and publish your documentation (wiki), among other things.

All features are enabled for blank projects, from templates, or when importing, but you can disable them afterward in the project settings.

To only use CI/CD features for an external repository, choose CI/CD for external repo.

Information about additional Pages templates and how to install them can be found in our Pages getting started guide.

**Tip:** You can also create a project from the command line. Show command

<table>
<thead>
<tr>
<th>Blank project</th>
<th>Create from template</th>
<th>Import project</th>
<th>CI/CD for external repo</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Project name</strong></td>
<td>CS-254-spring-2021</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Project URL</strong></td>
<td><a href="https://gitlab.com/icfox/">https://gitlab.com/icfox/</a></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Project slug</strong></td>
<td>CS-254-spring-2021</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Want to house several dependent projects under the same namespace? Create a group.

---

**Description format**

---

**Visibility Level**

- Private
  - Project access must be granted explicitly to each user.

- Public
  - The project can be accessed without any authentication.

**Initialize repository with a README**

Allows you to immediately clone this project’s repository. Skip this if you plan to push up an existing repository.

Create project

---

72
Setting Up GitLab et al.
Setting Up GitLab et al.

## Project members

You can invite a new member to **CS-254-spring-2021** or invite another group.

### Invite member

<table>
<thead>
<tr>
<th>GitLab member or Email address</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Douglas Craig Schmidt</td>
<td></td>
</tr>
</tbody>
</table>

### Choose a role permission

<table>
<thead>
<tr>
<th>Role permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintainer</td>
</tr>
</tbody>
</table>

[Read more about role permissions](#)

### Access expiration date

<table>
<thead>
<tr>
<th>Expiration date</th>
</tr>
</thead>
</table>

[Add to project](#)  [Import](#)

### Existing members and groups

**Members of CS-254-spring-2021**

<table>
<thead>
<tr>
<th>Member</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lindsey Fox @lcfox</td>
<td>Maintainer</td>
</tr>
</tbody>
</table>

[Given access 2 minutes ago](#)
Setting Up GitLab et al.
Accessing Java & Android Source Code
Accessing Java & Android Source Code

- Android source code is available
- For browsing android.googlesource.com

**android Git repositories**

To clone one of these repositories, install git, and run:

```
git clone https://android.googlesource.com/name
```

**Name**
accessories/manifest
device/asus/deb
device/asus/flo
device/asus/flo-kernel
device/asus/grouper
device/asus/tilapia
device/common
device/generic/armv7-a
device/generic/armv7-a-neon
device/generic/art
device/generic/common
device/generic/goldfish
device/generic-mini-emulator-armv7-a-neon
device/generic-mini-emulator-mips
device/generic-mini-emulator-x86
Accessing Java & Android Source Code

- Android source code is available
  - For browsing android.googlesource.com
  - For downloading source.android.com

The Android Source Code

Android is an open-source software stack created for a wide array of devices with different form factors. The primary purposes of Android are to create an open software platform available for carriers, OEMs, and developers to make their innovative ideas a reality and to introduce a successful, real-world product that improves the mobile experience for users. We also wanted to make sure there was no central point of failure, where one industry player could restrict or control the innovations of any other. The result is a full, production-quality consumer product with source code open for customization and porting.

Governance Philosophy

Android was originated by a group of companies known as the Open Handset Alliance, led by Google. Today, many companies – both original members of the OHA and others – have invested heavily in Android. These companies have allocated significant engineering resources to improve Android and bring Android devices to market.

The companies that have invested in Android have done so on its merits because we believe an open platform is necessary. Android is intentionally and explicitly an open-source – as opposed to a free software – effort; a group of organizations with shared needs has pooled resources to collaborate on a single implementation of a shared product. The Android philosophy is pragmatic, first and foremost. The objective is a shared product that each contributor can tailor and customize.

Uncontrolled customization can, of course, lead to incompatible implementations. To prevent this, the Android Open Source Project also maintains the Android Compatibility Program, which spells out what it means to be “Android compatible” and what is required of device builders to achieve that status. Anyone can (and will!) use the Android source code for any purpose, and we welcome all legitimate uses. However, in order to take part in the shared ecosystem of applications we are building around Android, device builders must participate in the Android Compatibility Program.

The Android Open Source Project is led by Google, who maintains and further develops Android. Although Android consists of multiple subprojects, this is strictly a project management technique. We view and manage Android as a single, holistic software product, not a "distribution", specification, or collection of replaceable parts. Our intent is that device builders port Android to a device; they don't implement a specification or curate a distribution.
Accessing Java & Android Source Code

- Java 11 source code is available
  - For browsing zgrepcode.com
Accessing Java & Android Source Code

- Java 11 source code is available
  - For browsing zgrepcode.com
  - For downloading github.com/openjdk
Summary
Summary

• You will get out of this course what you put into it
Summary

• You will get out of this course what you put into it
• Be prepared to work hard

HARD WORK

“Human Felicity is produc'd not so much by great Pieces of good Fortune that seldom happen, as by little Advantages that occur every Day” - Benjamin Franklin
Summary

• You will get out of this course what you put into it
• Be prepared to work hard
• Do *not* miss deadlines...

See [github.com/douglascraigschmidt/CS254/wiki/Assignment-Deadlines](https://github.com/douglascraigschmidt/CS254/wiki/Assignment-Deadlines)
You will get out of this course what you put into it
- Be prepared to work hard
- Do *not* miss deadlines...
- Participate in discussions in class & on piazza

See [piazza.com/vanderbilt/spring2022/cs254](piazza.com/vanderbilt/spring2022/cs254)
Summary

• You will get out of this course what you put into it
  • Be prepared to work hard
  • Do *not* miss deadlines...
  • Participate in discussions in class & on piazza
• Avail yourself of available resources

See [www.dre.vanderbilt.edu/~schmidt/cs254](http://www.dre.vanderbilt.edu/~schmidt/cs254)
Summary

• You will get out of this course what you put into it
• Be prepared to work hard
• Do *not* miss deadlines...
• Participate in discussions in class & on piazza
• Avail yourself of available resources

Please resist the urge to email me directly unless it’s a confidential matter or you’d like to schedule a meeting!
• There are abundant opportunities!

**Summary**

<table>
<thead>
<tr>
<th>BROAD CATEGORY</th>
<th>2020 SALARY PROJECTION</th>
<th>2019 SALARY PROJECTION</th>
<th>% CHANGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engineering</td>
<td>$69,961</td>
<td>$69,188</td>
<td>1.1%</td>
</tr>
<tr>
<td>Computer Science</td>
<td>$67,411</td>
<td>$67,539</td>
<td>-0.2%</td>
</tr>
<tr>
<td>Math &amp; Sciences</td>
<td>$62,488</td>
<td>$62,177</td>
<td>0.5%</td>
</tr>
<tr>
<td>Business</td>
<td>$57,939</td>
<td>$57,657</td>
<td>0.5%</td>
</tr>
<tr>
<td>Social Sciences</td>
<td>$57,425</td>
<td>$57,310</td>
<td>0.2%</td>
</tr>
<tr>
<td>Communications</td>
<td>$56,484</td>
<td>$52,056</td>
<td>8.5%</td>
</tr>
<tr>
<td>Humanities</td>
<td>$53,617</td>
<td>$56,651</td>
<td>-5.4%</td>
</tr>
<tr>
<td>Agriculture &amp; Natural Resources</td>
<td>$53,504</td>
<td>$55,750</td>
<td>-4.0%</td>
</tr>
</tbody>
</table>

Summary

• If there’s an emergency, pay attention to the escape route!
• See engineering.vanderbilt.edu/about/evacuationplans.php