Case study

CS program suggestions for one university

2017

Purpose: Sharing information that benefited one university

Google is frequently asked to provide an industry/employer perspective on what students need to know to be career-ready. This slide deck represents the thoughts of a Google engineer who is also an adjunct faculty member in CS at a major university. He created this deck for the CS department at another institution, and they found the information helpful.

Google recognizes that CS programs and students have different needs and that faculty are the best judges of what will most benefit their students and their institutions. We hope some of this information will be useful to you.

Examples of skill areas that contribute to students' success in industry

- Problem-solving large, complex challenges; for example:
    - Experience and comfort working with very large pieces of code
    - Designing solutions before starting to code
    - Generating multiple solutions and choosing the most suitable one
    - Articulating the logic behind design and coding decisions
    - Choosing appropriate concepts to apply to open-ended problems
    - Solving problems with ambiguous and/or incomplete requirements; and effectively filtering and using the most relevant information about a problem
- Working in existing code bases, including reading and understanding existing code, and being able to test and debug code you've received or added
- Experience and comfort working with open source resources (for example, GitHub)

Categories of questions that might be asked in a technical interview

- Coding
  - Java/C++/Python
  - Recursion
- Math
  - Basic probability
  - Div/mod
- Operating Systems Concepts
  - Threads and processes
  - Memory management
  - Filesystems and networking
- Algorithms
  - Sorting
  - Searching
  - Graphs
  - Big-O analysis
- Data Structures
  - Lists
  - Stacks and queues
  - Trees and graphs
  - Sets
  - Hashes and maps
  - Heaps

Possible areas of emphasis for freshman (1st) year

- Minimum of one CS course and one core math course
- Proficiency in an object-oriented programming language (e.g. Python/Java/C++)
- Ability to write tests and debug
- CS project outside of the classroom
- Ability to read and understand existing code
- Basic problem-solving; basic understanding of formal logic
- Other important topics: Documentation, version control
- Familiarity with using Github

Freshman (1st) year

Semester 1

Goal: Introduction to programming concepts

- Intro to Programming (Python or Java)
- General education courses
- Technical/Non-technical electives

Important topics/milestones:

- Debugging and testing
- Commenting and documentation

Semester 2

Goal: Comfortable with at least one language

- Object-Oriented Programming (Java)
- General education courses
- Technical/Non-technical electives

Important topics/milestones:

- API documentation (Javadocs)
- Version control (GitHub)

Google

Possible areas of emphasis for sophomore (2nd) year

- Understanding of algorithms, data structures and discrete structures
- Proficiency in one or more programming languages (Python/Java/C++)
- Demonstrated ability to investigate and solve problems
- Ability to provide and receive code feedback
- Other important topics: Time and space complexity, MIPS, simulation-based projects
- CS projects (Examples: Independent projects, team projects in and out of class, hackathons, small role in a larger open source project at site like OpenHatch.org or GitHub)
- Active participation in CS clubs or organizations

Sophomore (2nd) year

Semester 3

Goal: Understanding of data structures

- Data Structures (Java)
- Discrete Structures
- General education / technical electives

Important topics/milestones:

- Collaboration on open source site
- Team projects (e.g. hackathons)

Semester 4

Goal: Exposure to algorithmic complexity

- Algorithms (Java)
- Computer Organization (Assembly)
- General education / technical electives

Important topics/milestones:

- Time and space complexity
- MIPS Simulator-based projects

Possible areas of emphasis for junior (3rd) year

- Demonstrated mastery in a programming language
- Proficiency with algorithms and advanced data structures: lists, hash tables, trees, graphs, sorting algorithms, etc.
- Demonstrated knowledge of algorithmic efficiency and design tradeoffs, threads and processes, concurrency and synchronization
- Solid foundation in discrete mathematics
- Internship experience in software engineering
- Other valuable courses/topics: Linear Algebra, Computer Networking, Operating Systems, Database Systems

Junior (3rd) year

Semester 5

Goal: Advanced programming concepts

- Linear Algebra/Num Analysis (Python)
- Computer Networking (Java or Python)
- General education / technical electives

Important topics/milestones:

- Advanced data structures (graphs/trees)
- Filesystem and networking

Semester 6

Goal: Knowledge of core systems

- Operating Systems (C or Java)
- Database Systems (Python or JS)
- General education / technical electives

Important topics/milestones:

- Threads and processes
- Concurrency and synchronization

Possible areas of emphasis for senior (4th) year

- Demonstrated mastery in a programming language
- Demonstrated mastery in data structures and algorithms
- Databases: locks, concurrency, RAID, APIs
- Other topics: Computer security, big data, machine learning
- Software engineering industry experience/internship, that includes application of SWE and CS concepts and work on real world project(s)
- Update resume (May want to seek a formal review, e.g. from career center)
- Interview prep, including advice/referrals from former interns, alums, etc.

Senior (4th) year

Semester 7

Goal: Industry preparedness

- Computer Security
- Software Engineering
- General education / technical electives

Important topics/milestones:

- Technical interview prep

Semester 8

Goal: System design experience

- Big Data or ML course
- Senior Capstone
- General education / technical electives

Important topics/milestones:

- Practical industry level project

Summary of sample courses by semester

Semester 1

- Intro to Programming (Python/Java)

Semester 2

- Object-Oriented Programming (Java)

Semester 3

- Data Structures (Java)
- Discrete Structures

Semester 4

- Algorithms (Java)
- Computer Organization (Assembly)

Semester 5

- Linear Algebra/Num Analysis (Python/R)
- Computer Networking (Python/Java)

Semester 6

- Operating Systems (C/Java)
- Database Systems (Python/JS)

Semester 7

- Computer Security
- Software Engineering

Semester 8

- Big Data or ML course
- Senior Capstone

More Resources

- Google's Guide to Technical Development: g.co/techdevguide
- Google for Education: https://edu.google.com/resources/computerscience/
- Recommended textbook for interview prep: Cracking the Coding Interview
- Google Cloud Platform Education Grants: https://cloud.google.com/edu/
- Sites with practice problems:
  - LeetCode: https://leetcode.com/
  - Project Euler: https://projecteuler.net/
  - CodingBat: http://codingbat.com