SWAG STUDENTS WITH AMBITION GO TO COLLEGE

Young Professionals Program
Computer Science Guide

Table of Contents

Introduction and Overview

This guide is designed as a resource for students considering and majoring in Computer Science, and to help volunteer mentors guide college mentees interested in computer science. There is no "one size fits all" solution, and the experience will be unique to student. As a result, this guide is to be used as a supplemental tool and not as a strict format guide for students or for mentorship. However, this guide and mentors in general cannot be the primary source for providing a student with resources and information. Students must seek out resources on their own, with their mentor providing them support and direction when possible.

Anyone who has a passion for computer science and the right work ethic can succeed in the field. Besides a good work ethic, there is no longer a specific skill set that is essential for the field because there are so many distinct subfields. A modern computer science undergraduate degree will introduce students to many of these fields and career paths. However, many computer science jobs do expect students to be adept at a core set of computer science skills, and technical interviews that test these skills are a core part of the hiring process.

Mentors should help mentees understand the range of opportunities available to them as CS students and employees. A basic knowledge of computer science will be helpful in the modern day workforce, as it is being used to improve methods in other areas. Furthermore, a deep knowledge of other professional fields may be important for CS majors who want to work on applications in such fields.

A computer science education is about learning the technical skills you need to succeed in the workforce, but also to develop yourself as a well-rounded person who can diagnose problems and lead teams to solve them. A few things to keep in mind:

a. College is about learning skills, but also about expanding your interests in general. Don't pigeonhole yourself too much or too early.
b. Peers, older students, professors, and universities provide resources that are all incredibly helpful. Student organizations (both in the major and throughout the college) are also good resources and offer opportunities to meet peers, professors, and even recruiters.
c. Internships are a key part of eventually getting a full-time job, especially in computer science. By the time you are a senior in college, you should have at least one internship experience in a field related to your desired full-time position.
d. There are also many non-internship opportunities for you to develop and demonstrate your skills, including pursuing projects on your own and participating in hackathons.
e. If you aim to pursue further education, such as a MS or PhD in computer science, excelling in your classes, meeting your professors, and conducting research is vital.

In the next few sections we'll dive into some of these aspects in more detail.

What is Computer Science

Computer science, most generally, is the study of the design and engineering of computer systems. These systems include the apps and websites with which students may be familiar — but also the operating systems of their phones and computers and the software that are used to make our favorite movies at Pixar. The role of computer scientists is to formalize, automate, and scale processes that need to occur — for example, what needs to happen when someone clicks the "buy" button on Amazon.

Computer science isn't just a tech industry focused on programming and app development; it is also now a part of every industry. As one investor recently put it, "Software is eating the world." Computer science now encompasses a large diversity of skills and types of jobs, which are all open to students. In computer science there is a big emphasis on learning certain skills pertaining to the specific field in which you go.

Fields of Study within CS

There are many different subfields of CS. A traditional CS undergraduate degree will cover many of them in their core courses, and students often will have the opportunity to focus on a few areas during their later years. One common myth may be that every class in a CS curriculum is about how to write code or how to use various programming languages. In reality, fewer than half of a student's courses may require writing code. Below, we mention a few such sub-fields to highlight the diversity of skill-sets to which CS caters. For a more complete list, see here.

a. **Algorithms and data structures.** The bread and butter of a computer scientist. Algorithms are formal specifications -- think step by step instructions for a computer -- to solve a given problem. Common algorithms, such as for sorting lists, searching from lists, or for finding the shortest path between two points on a map, are used in many software systems. Efficient ways to perform such tasks are known and continually in development for different settings. Data structures are things that store and help access data, and different data structures are suited for different tasks. Algorithms and data structures require logic skills, and every computer scientist should have this skillset; many technical interviews often test these skill sets.
b. **AI/Machine Learning/Data Science**. Development of computer systems that can improve over time with data, or trying to understand something through the data it generates. Can be very application and programming heavy (where you learn certain techniques and apply them to biology, social sciences, and other fields), where one can be expected to apply and adapt existing methods. Can also be more theoretical to develop such methods.
c. **Human Computer Interaction.** The design and study of the interface between humans and computers. Attracts many design oriented people, and often requires little to no math skills.
d. **CS Theory**. Consists of more traditional mathematics to study the limits of computation. Mostly studied in late undergraduate or graduate school.

Career Options in Computer Science

Just as there are many subfields of CS, there are many types of jobs and career options available to CS students. Here, we highlight a few.

a. **Software Development.** Build computer systems (e.g. websites and mobile applications) through code. Many different types of software developers, and often characterized as either front-end (those who build user interface code) and back-end (those who build functionality). Software development is what people typically think CS is. As software developers advance in their career, they often move into software architect roles in which they design entire systems for others to develop.

   b. **Product Management**. Develop the specifications of what a program or feature should do and often manage/oversee the development of it.
      a. Requires good person-facing skills, as you will have to talk to customers, developers, and designers to figure out what customers want and what developers can deliver.
      b. Product management pursued right out of undergrad (for example Microsoft hires college graduates for development, product management, and data science). Though "management" is in the name, often product managers do not directly manage people (the developers) but are on their own separate teams together.
   c. **Data Science.** Analyze data about how a company's products are being used and try to figure out patterns that explain it. Often then lead or are involved in the design of new features in response to the patterns they have found. For example, a data scientist at a ride sharing company (such as Uber or Lyft) may learn through data that one of the biggest use cases of the product is to go to the airport, and the company may develop special features that make those rides especially smooth.
      a. Requires curiosity, ability to ask questions, and knowledge of statistics and algorithms.
   d. **Design**. Design and often program/build what users see when interacting with a program/computing system
      a. Does not always require extensive coding knowledge
      b. Related to art and design

Further reading: https://en.wikipedia.org/wiki/Outline_of_computer_science

Who is CS for

Getting a computer science degree is hard work and often requires taking many classes that are project heavy, and so they take time. However it uses surprisingly little *specific* skills from high school, besides a general ability to reason, unlike potentially other STEM degrees. For example, you probably will not use much geometry or calculus in your entry level CS classes.

Furthermore, though some entering students nowadays have a fairly strong CS skill-set out of high school through competitions and high school classes, that is not mandatory at all, though it can of course help.

If you are still deciding on a major, sit in on CS classes and talk to CS students. CS is also at the cutting edge of moving classes online, so consider watching (free) lectures or completing introduction CS classes on websites such as Coursera or EdX (these websites are host many advanced courses taught by excellent professors). A computer science degree may not suit everyone, but the only way to find out is to give it a shot.

As for any other major, list the reasons you have for choosing CS, and reasons you may have for not choosing it. Discuss these reasons with your mentor or others. While no decision is right/wrong for everyone, some reasons may be better than others.

On the other hand, students with CS skills are often some of the most sought after in the job market, with some of the highest starting salaries. It is also something where your skillset often matters more than the actual degree/where you got it (though both are important to get interviews).

Professor Relations and Recommendations

As in college as a whole, professors are a great resource for computer science students, whether the student is looking to enter industry or go to graduate school after graduation. Professors often have extensive and experience with industry -- they often collaborate with industry in their research. If you have a specific problem or concern, professors may have advised students in similar situations and so may know how to help.

Professors can especially help you with the following:

a. Questions about the class material.
b. Pointers to further resources (such as books or other things to learn), whether directly related to a class they may be teaching or just other things they may know.
c. Advice for which classes you should take, and in what order.
d. Advice about whether to go to graduate school, and how to apply.
e. Advice for how to look for jobs, or what types of positions you may enjoy/should look for.
f. If you are considering graduate school, especially pursuing a PhD, professors are vital in the process. Recommendations the most important aspect of a graduate school application. Talk to professors about your interests, and try to get involved research.
g. Any other concern you may have, especially if it relates to a class you are taking with them.

Unfortunately, professors are often busy creatures, working with undergraduate students may not be their first priority, and they are notoriously bad about responding to emails. Some of the following tips may help you communicate more effectively with professors. Of course, professors are human, and the below may not apply to all professors. Some are more helpful, nicer, and more approachable than others. But all are busy, and most respond well to the below behavior.

a. **Go to office hours!** Often-times their office hours are empty, and (many) professors love helping out students directly during office hours. If you are taking a course with the professor, refer to the syllabus for their office hours. If not, try to find them online on their website (professors often have a personal/academic website highlighting their research, teaching, and other activities).
b. **Be prepared, and show that you've done your homework.** (both literally and figuratively). Professors love to help students that have tried on their own, and nothing annoys them more than laziness. If you have a question, be ready to show what you've tried before (such as reading the book, paying attention in lecture, or asking others). If it's about coursework, it's best to go to a professor with, "I have tried XYZ, but don't understand ABC still." (Unless of course if you have a clarification question during class or right after a lecture).
c. **Use appropriate, designated channels first.** There are often many students in a class, and only a few TAs and one professor. The teaching staff may set up or designate certain channels through which to ask your question -- such as class forums or email lists. Use them first. You will be more likely to get a response. If you do not use such channels, the first thing a professor/TA will say to you is to look for the answer there.
d. **In person is often better than email**. One shock many younger students face is receiving a terse reply from a professor over email, or no reply at all. It's often not the student's fault -- professors often get many emails and are poor at responding to all of them. It is often easier to approach the professor before or after class, or during office hours. If you must send an email, do so using your school-provided email address if you have one, and have a specific email subject.
e. **Refer to their websites and figure out their research interests. You can contact professors that you do not know.** Many professors are also researchers who have an expertise in a specific sub-field of computer science. Many CS departments will have professors with a wide range of expertise. If there is a specific subfield or type of job in which you are interested, contact the professors with that expertise. Your department website will often link to the websites of all the faculty in the department, and these websites will discuss their research interests. Feel free to reach out to the professors who do things that interest you, even if you have never taken a class or otherwise interacted with them. Write a short introduction email (or go to their office hours) and include why you are contacting them in particular. Expect some to never respond, but others may.

Internships/Jobs

Internships are important in ultimately finding a full-time position -- it gives you experience with what a real CS position is like and gives confidence to employers that you will be an effective employee. Furthermore, CS internships are often well-paid, and so they remain an option for students who may have to earn money over summer.

However, finding an internship can be one of the most stressful parts of college life, especially as recruiting cycles at many colleges start earlier and earlier -- even more so for CS positions. This guide is not meant to serve as a comprehensive guide to finding a CS internship, and there is only so much help that mentors can provide. Students must be willing to seek out resources available at their college -- most of all other students.

Career fairs are often the best way to meet recruiters and apply for internships, but keep in mind that good positions are well sought after and each company may only be able to interview a few applicants. Recruiters often visit various CS student organizations or may hold departmental informational sessions before the career fair. Attending these sessions will help you learn more about the company and potential positions, and will also give you a chance to meet the recruiters in person and share your resume. Often you may have to fill out an online application for a position if a certain company does not visit or interview at your school in person.

The general process often has the following steps, though of course it varies:

a. You meet a recruiter at an event/career fair and hand off your resume.
b. You are asked to fill out an online form with your resume details.
c. You are scheduled for an on-campus interviews. These typically all occur together a few weeks after your school's career fair.
d. You are scheduled for follow-up interviews, which may be over the phone. You may be flown to the company's offices to interview for a full day in person.
e. You are given an offer, and have a couple of weeks to decide whether to take it. You may be in the process of interviewing at other places, and you may decide to inform the other companies that you have an offer so that they speed up their own process.

Of course, many other steps may be taken before you enter meet a recruiter:

1. Identify which employers have formal summer internship programs in which you are interested.
2. Create a list of employers that conduct on-campus recruitment using campus career services centers or advice from more senior students
3. Begin building relationships with older students who have successfully completed internships the previous summer that you would like to pursue
4. Create a resume for summer internship recruitment
5. Attend on-campus career fairs and recruitment informational sessions. Also attend networking events with companies of interest to you when they send teams on campus to better your chances of a summer internship interview.

6. Begin resume practice and coaching with fellow students and available on-campus organizations.

Career fairs often happen early, including in the Fall semester at many colleges. The earlier you apply, the more positions that may be available. Furthermore, note that you will not get an offer from every company to which you apply; rejection is part of the process, so apply to as many as interest you.

Resume

There is no single template for what makes a good resume or what components you must include in it. The goal of the resume is to get an interview, and thus you must include things that convinces recruiters to give you one. Keep in mind that resumes are often skimmed quickly, so *less is often more* -- don't stuff the resume with irrelevant things. Furthermore, expect to be asked about anything that is on your resume in an interview. If you're unable to answer well, then that is much worse than if you didn't include it in your resume in the first place, especially when it comes to a technical skill.

Some things that often belong in CS resumes:

a. Your major/education information. Potentially but not necessarily your GPA.
b. Past work experience.
c. Classes you have taken, especially technical ones that relate to the type of position to which you are applying. Order them with the most advanced/most related courses first.
d. Technical skills. This is often unique to CS/STEM resumes. Include programming languages with which you have experience, or types of programming (e.g., web development, user interfaces, back-end, systems, robotics).
    a. Note: Interviewers often ask technical questions about specific parts of your resume, and not being able to answer is much worse than not including a specific skill on your resume in the first place.
e. Descriptions of technical projects or projects you have done in your CS classes.
    a. If you have a github account with examples of code/projects, link to it. Github is a (free) website where you can upload code on which you have worked and make it available to share. Some of your classes may teach you *git* which is how you can use Github to collaborate on code with others.

Technical Interview

Another aspect of the CS job or internship process is the *technical interview*. In the technical interview, you will be asked to write code or design a system, or otherwise solve a mock problem representative of the type of job to which you are applying. They are often the most important part of the process -- your resume/school/degree may get you an interview, but from there passing the technical part is essential.

These interviews take many potential forms:

a. Write code -- Given a problem, and are asked to write code to solve it. The emphasis is often not on getting every aspect of the syntax right, but often on the general approach and your ability to talk through what you are trying to do.
    a. May be on a whiteboard while talking to the interviewer.
    b. May have to write it on a computer/google doc if online.
    c. May be given a written problem, and given a few days to write actual working code to solve the problem.
b. General technical questions
    a. May be asked about a programming language's features for design.
    b. May be asked which algorithm one would use to solve a given problem.
c. System design questions -- Given a product/or feature that the interviewer may be interested in building, and you are asked to help design it. This may include discussing which features to include and which to avoid. May also include sketching out a potential software architecture to build the product.

How to approach it

a. Practice. There are many practice coding interview websites, books, and other resources out there. Some are linked below, but there are many others as well.
b. If something is on your resume, be ready to answer technical questions about it.
c. Talk to older students or recruiters about the types of skills one is expected to know for the position, or what the interview will be like.
d. Practice talking while writing on a whiteboard.
e. Certain classes will be much more helpful than others for technical interviews. Of course any class in which you code is helpful. Algorithms and data structures courses are in general extremely helpful.

Additional resources online for CS internships

There are many resources online that help you practice and prepare for coding interviews and write a resume. Some are paid, but many are free and of similar quality. We list some resources below that may be useful. Paying for a resource is probably not necessary or desirable given the quality of free resources, but make your own decision after looking through the resources.

a. http://web.stanford.edu/class/cs9/. A Stanford class that has helpful information (see notes, slides) on writing a CS resume and practicing for technical interviews, including links to practice problems. Some of the advice is Stanford specific (e.g. which classes to take), but can be modified for other schools.
b. https://www.cs.utexas.edu/blog/how-get-internship-freshmen. Internship advice for first years from UT Austin students
c. https://www.cmu.edu/career/documents/sample-resumes-cover-letters/sample-resumes_scs.pdf. Examples of CS resumes from CMU
d. https://www.testdome.com/for-jobseekers, https://coderbyte.com/challenges, https://leetcode.com/problemset/all/, https://www.hackerrank.com/. Collections of practice problems. Mostly free, but some sites have extra problems for which you have to pay.
e. http://www.crackingthecodinginterview.com/. A book that some people have found helpful, with many practice questions and other tips.
f. ...and many more. Search online, and ask others if they have found any particularly useful.

Personal Development Outside of the Classroom

Learning should not be confined to the classroom, and this is also true in CS. Here we talk about several ways to develop yourself after the classroom: student organizations, research, and side projects.

Student Organizations

Student organizations make up a core part of the college experience for many students. They are excellent ways to meet people both inside and outside your major. Within CS, organizations are especially useful as ways to meet people from industry, including recruiters -- student organizations often partner with companies to host events, either for just their members or for any student.

Within CS, several national organizations have student branches at many colleges and universities. These organizations provide structure and support for events, a recognizable name when talking about your involvement, and a chance to attend regional or national events. These organizations include ACM, IEEE, HKN, and TBP.

Research Experiences

Research is “systematic investigation in order to establish new facts or reach new conclusion.” At many universities, it is often the real reason why professors and graduate students are there.

For an undergraduate student, participating in research has many potential benefits:

a. Get in-depth knowledge of a particular area and get to apply and further develop the skills you are learning in the classroom.
b. Figure out if pursuing research after graduation -- by pursuing a PhD -- is for you. Undergraduate research is important for admission into a PhD program.
c. Interact with professor far more, including on a 1-1 basis. As professors consider their “real job” to be research, they are often far more available and helpful to their research assistants.
d. Potentially be paid to be a research assistant, or get tuition reimbursements.

How to get research experience

a. Talk to your professors, or other professors in your department. (see Professor Relations section for some tips on how to do so). Though they often do not advertise such positions, many are willing to work with undergraduate students.
b. There are many summer research opportunities at other schools to which you many also apply instead of internships (The REU program is especially popular).

Side projects and hackathons

Projects are a great way to practice and develop your skills, and interviewers like hearing about them. It’s possible (and educational, and fun!) to build things -- phone apps, websites, or other systems -- with friends, and many software companies you have heard of started out that way. There is also now a robust ecosystem -- both nationally and locally at many colleges -- to support students working on such projects. Some examples include:

a. **Hackathons** are events in which people form small teams to intensely work on a project over a day or several days, with many such teams sitting in large rooms together, in a social atmosphere. There often are presentations and awards at the end. Student organizations may host hackathons, and some universities hold large hackathons with students around the country attending. Some examples include at UPenn, Stanford, and Texas.
b. **Online coding challenges** are ways to develop certain skills in a competitive environment with people around the world. Websites such as kaggle (data science

focused) host various competitions and an automatic way to measure your performance against others, often with prizes given to winners.

After Undergraduate Degree

You have many options available to you after a CS degree. This guide cannot answer which is best for each student, as that depends on their particular situation. Possible things a student can do after graduation:

a. Work full-time in a CS position or other type of job.
b. Pursue further CS education, such as a MS or PhD in CS.
c. Pursue other further education, such as a MD (medical school), JD (law school) or MBA (business school). Such degree programs often seek out students with technical backgrounds, though you must also complete the specific requirements for your desired program.

It is best for the mentee to discuss these options with their mentors, peers, older students, professors, and people who have taken each of these options.

Pursuing a Master's degree (MS) in CS

A MS in CS is popular nowadays to gain more specific CS skills and to go deeper in a subject of interest. It can also be used to switch to a different type of CS position (from software engineering to data science, for example). A MS degree often takes 1-2 years, or longer if pursuing it part time while working.

It is sometimes beneficial financially in one's career, but the cost of a MS degree and the opportunity cost of not working a full time job while pursuing a MS should be taken into account. Many students start a MS right after completing their undergrad (either at the same school or a different one), and others work for a few years in industry first.

Some schools now offer completely online MS degrees in CS that you can complete while working full-time anywhere, though it takes a lot of dedication and time. For example, Georgia Tech has such a program. Other schools provide part-time MS degrees for those working full-time in the same city.

Pursuing a PhD in CS

A PhD, in any field, is a long and difficult journey of 5+ years after your undergraduate degree. People who pursue a PhD do so because they want to be a professor and/or go extremely deep in a specific subject area while conducting original research. In CS, it is most likely not beneficial financially to pursue a PhD, as you will spend an extra 5+ years not earning much money. If you think you want to pursue a PhD, talk to your professors (early; first year is not too soon) and start being involved in research. You may fall in love with research and decide it is for you, or you may conclude the opposite.

College Timeline

Here, we lay out goals for students for each semester of their college years. However, this guide cannot possibly capture the experiences of every student, and it should be used as a rough guideline, supported by specific mentorship and guidance from others.

Freshman Year

The first year of college is a time for exploration and discovery. It is estimated that up to 80% of college students change their major, or area of focus at least once. While it is perfectly acceptable to change directions, it should be stressed to students that they approach their current field of study with full commitment. Focusing on academic performance for the first year is particularly important, as it sets the tone and pace for a student's college career going forward. In addition, students should focus on getting acclimated to the campus, learning more about their departments, and most importantly, making friends and building relationships.

First Semester Goals

1. Build a course schedule that is challenging, but not overwhelming
2. Establish a study group with students in core classes who also share similar areas of interest
3. Discover on-campus resources that support the student's current area of interest, including campus counselors, peer advisors, freshman seminars, career centers, and academic clubs.
4. Explore areas of potential interest outside of the student's current major
    a. Talk to and make friends in other departments
    b. Explore interests through elective courses
    c. Sit in on another department's class with a professor's permission
5. Strive to get as high a GPA as possible
    a. While this point seems obvious, it is always worth emphasizing, particularly early in a student's college career. Coursework will only become more challenging as students' progress, so setting the pace with a high GPA will provide a great advantage going forward. Furthermore many opportunities for clubs and honor

    societies will factor in current GPA, so students should strive to lock in a high GPA early to access more opportunities throughout their college career.

    b. It should be noted that grades aren't everything. As a student becomes more immersed in campus-life, they will find areas of focus outside the classroom that are equally important to pure academics. That being said, first semester is a time to put their best foot forward, and starting out with a high GPA will help a student build momentum going forward.

6. Explore opportunities for a summer internship or work experience
    a. Internships are difficult to come by for students with only one year of experience, and the first semester may seem too early. At the least, consider attending a few informational sessions and the career fair in order to learn about the process, but do not necessarily prioritize it over the many other things you should/can do as you start college.
    b. On the other hand, some CS companies have programs especially designed for first and second year students, and it doesn't hurt to start early. Some research programs are also designed for first year students.
    c. CS internship recruiting especially is concentrated to the first semester, though opportunities are available in the second semester.
    d. A student who needs to earn income over the summer should try to find employment that gives them exposure to an office environment.
    e. Internships are a momentum game. The internships that provide better experience are often awarded to candidates with some kind of previous experience.
    f. However, do not stress if you do not get an internship -- there are many useful things one can do in the summer, even without a formal program.

Second Semester Goals

1. Select a course schedule, striving to take more challenging courses that progress the student forward in their current major track
2. Establish a study group with students in core classes who share areas of interest
3. Find a peer mentor in the student's major who are in their 2nd or 3rd year, either through personal network or a student organization
4. Join at least one academic and one social club
    a. Student organizations serve multiple purposes. They give students opportunities to experience the campus outside of the classroom and meet friends they might not have through their organic social network.
    b. All student organizations are different and a student might achieve an academic and social component from a single well developed organization.
5. Continue pursuing a summer internship or other summer plans.

Sophomore Year

The second year of college is still a time for exploration, but students should more aggressively immerse themselves in the academic and college experience. By this point, a student should start to feel comfortable in their current track or be proactive in making adjustments to change their major. While there are many successful people who change their majors late in their college career, students should be aware that changing majors in their later college years could lead to extended time and cost to gain their degree. For students confident in their major, they should begin to evaluate their area of focus.

First Semester Goals

1. Work with a guidance counselor to plan out a course schedule that allows the student to achieve their degree requirements in 4 years
2. Select an area of focus within the student' major that they would like to pursue
3. Join at least one academic club that specifically supports students interested in the student's specific major and area of interest
   a. The club should be a place where younger students can interact with upperclassmen, better understand the industry they wish to pursue, and find resources that can help them develop internship opportunities
4. A student should utilize both upperclassmen and school guidance resources to understand the opportunity landscape, and develop a schedule for exploring those opportunities.
5. Pursue summer internship, research, or other opportunities.

Second semester goals

1. Strive to take at least one upper division course relevant to the student's major
2. Try to take a leadership role in at least one club or extracurricular activity
3. Continue seeking summer opportunity.

Junior Year

By their third year, students should have confidence in the major they are pursuing and should be looking to begin actively exploring the opportunities they could find after graduation. The third year should be focused on self-development with the goal of securing a full time role post-graduation and actively finding ways to differentiate themselves. This is the year where students should feel comfortable and well adapted to the academic environment, giving them an opportunity to focus on extracurriculars, leadership roles, and summer internship recruitment. While there is never one set path to success, it should be noted that students who are able to secure a summer internship at a reputable company or firm have a tremendous advantage during recruitment for full-time roles the following year. A summer internship offers unparalleled experience that would be a true differentiator against students who lacked a similar experience. Additionally, many coveted employers select their full time hires through their internship recruitment pools, making the internship process even more vital to the student's long term success.

First Semester Goals

1. Be a part of at least two extracurricular activities on or off campus
2. Earn a leadership role in at least or club or organization
3. Continue talking to your professors and older students to plan out what you want to do after graduation and make sure you take the necessary steps.
4. Pursue a summer internship or research position, as your activities this summer will be especially important as you pursue your post-undergraduate goals.

Second Semester Goals

1. Continue finalizing your ideal plans for your post-graduation career, making sure your current and summer activities line up toward that goal.
2. If you are considering further education, make a plan for when you will take the necessary entrance exams.

Senior Year

The fourth year is often the final stage of the college experience. At this stage students should be focused on closing out their degree requirements, but equally focused on securing their opportunity post-graduation.

Depending on what one wishes to do post-graduation -- whether it be further education, a full-time job, or something else -- the timeline for Senior year can differ widely. Setting such a timeline for each opportunity is beyond the scope of this guide. However, by early first semester, a student should know what they want to do after graduation. They should then create a plan, with the help of mentors, professors, and friends, to achieve it.