1. Assuming your background is in software, what other duties and/or responsibilities may be required by employers for individuals with this skill?

* Working within a team. Almost all software of any consequence is done by a team or a number of teams working together. Software has become so complex and interacts with so many disparate systems; it is never done in isolation. This is one of the least cultivated skills I see with new grads.
	+ There are a number of Agile practices that have become common place such as Pair Programming and Test Driven Development where one developer thinks about the unit test first and the other then writes code to make the test pass. Then the roles reverse, where the unit tester is now the coder. Then the pairs rotate every hour or so. So imagine working with completely different people all throughout the day.
	+ Any Agile development involves what is called a planning meeting. Imagine your Scrum team of around 7 developers and QA, sitting in a room for an entire day and designing, breaking work into developer and QA tasks as a team. And this typically occurs every 1 or 2 weeks.
* Communication. The most successful developers are often the ones who are both smart and can convey their vision. Often the only difference between a senior developer and a software architect is being able to communicate to a team how something is built.
* Knowledge outside of software. Most software isn’t bought for its algorithms; it’s bought because of the value it adds. This is usually business value, but that can be anything from increasing efficiencies to social networking. I’ve worked with successful software companies that manage golf courses, manage resources for legal courts and even promote rock bands. If you can find a way take a passion of yours, combine it with software and make it profitable, you can have a license to print money.

2. What is the most interesting example of the application of your skills in either your current or past employment?

* In my current job, I’ve been able to take years of working within highly functional teams and been given the ability to build my own. I’ve hired a bunch of developers, quality assurance, and user experience. But then you have to get them all working together, come up with the right processes. It’s almost like building your own company.
* Another one was actually my first job out of university. Because I was both technical and could communicate well, I ended up as a Technology Evangelist for a couple of years, living in an airplane, working with potential customers, travelling across America, Europe and the UK.

3. To what extent did your university education prepare you directly for the responsibilities your currently hold?

* It was both the best and the worst experience. I can honestly say I have not used anything I have learned at university in the real world at all. Technology moves faster than academia, so everything taught in class was dated and useless. But…
* The co-op program at Waterloo was fantastic. My first co-op term, I learned about building software product for the enterprise. My second co-op term taught me about consulting. I spent 8 months in New York consulting for MSNBC and Nabisco and partied 6 days a week living in a great apartment on the upper west side.
* The other value is the network. I still stay in touch with people I met in my classes. Two of my closest friends just started a company together.

4. To what extent did your RSGC education prepare you for the academic expectations of university and how did your education compare to those of fellow undergraduates?

* You need to pat yourself on your back for taking this class. I can say that hands down, Mr. Darcy was the best teacher I had, including university. I felt like I learnt more about software in his class than I did in all of university. The only challenge was that Mr. Darcy taught valuable skills, while universities tend to teach academic theory, which I often didn’t care about. I care more about how to use a database, than how database was first built ten years ago. So going to Waterloo almost felt like taking a step back. I’ve told co-workers and friends about the all night take home exams we used to do and people were blown away.
* In summary, if you think you are going to go to any of the best technical universities, Waterloo, MIT and then be able to walk into a software company and kick ass, you are have another thing coming. All of the best people I know in software are passionate about what they do, constantly reading, teaching themselves skills at home, reading blogs, listening to podcasts, etc.

5. As students, we are told that our futures will require strong communication, collaborative, and problem-solving skills. To what extent have you applied skills in enabling you to assume your current roles and responsibilities?

* As I said, software these days is all about how effective a team is. Whenever there are issues with software delivery, it’s always a people problem, not a technical problem. Usually, lack of communication is a key reason software projects fail. There has been a growing movement for the past 10 years to deliver software with smaller, smarter teams specifically because it is easier for smaller teams to communicate. If you are the best developer in the world, but can’t work with people, convey your ideas, collaborate on problems, you will be at the bottom of the heap.

6. What has been your most humbling failure(s) and to what extent have these contributed to your current success?

* Short term, I was working at a great company as their software architect, built a kick ass system over a couple of years and then started to coast. I was able to keep the project going without really refreshing my skills. When I finally changed jobs, I realized that it only took 3 years for my skills to be out of date. It took me 6 months of intensive learning to get back to the same level as the other architects at the company. So I’ve taught myself, to always be learning.
* Longer term, since the beginning, I’ve switched between consulting companies to product companies, back and forth. I liked the camaraderie and variety of working in consulting, but would also like working on a single product, thinking about long term strategy. Almost 5 years ago, I finally realized that working for consulting companies was just not a good long term career move. A lot of my closest friends stayed in consulting and now they are miserable. A few of them want to get out, but they are struggling to make the jump into product. So my failure was staying in consulting as long as I did, but now I’m smarter and will never go back. Now I’m working for a great product company and couldn’t be happier.
* I also regret not pulling the trigger on starting my own business at some point. I did try a few times, but it’s hard to walk away from a great salary and to take a big gamble like that.

Bonus: What is the one thing I know now, I wish I knew going into university.

Your first jobs have a very big effect on your whole career and you will likely change jobs often. I’ve seen a lot of people who ended up in the same industry niche that they started in, for most of their career. I’ve also seen a lot of people who want to transition, but can’t. The only ones who transition easily are the ones that consciously switch often. In my opinion, there are almost like 6 potential tracks. You may want to think about where you want to go much earlier than you would expect. Caveat is that this is software focused, not hardware, biotech or anything else.

* Enterprise IT – Food manufacturers to Banks
	+ Pros
		- Often the easiest to get into and often people’s first jobs
		- Large number of jobs
		- Can transition out of tech into something else. Go from programming stock broker application to being a stock broker.
		- Can be quite stable
	+ Cons
		- Not as challenging
		- Technology is rarely the driver
		- Tech can be perceived as a cost center, not an enabler
		- Don’t pay as well long term
* Consulting companies – T4G to EDS
	+ Pros
		- Lots of variety
		- Very broad range of technologies
		- Pays well
		- Camaraderie within your company
	+ Cons
		- You rarely deep dive on anything unless you specialize. E.g. you become the an SAP consultant
		- Volatile, especially in sweat shops
		- Arm’s length from clients
* Independent consulting – You work for yourself, billing by the hour
	+ Pros
		- When it’s good, pays very well
		- You can be flexible. When your contract is done, take 2 months off
		- Tax benefits like writing off our car
	+ Cons
		- You have to stay on top of your game. Either always learning something new on the cutting edge or highly specialized in a valuable skill.
		- You are the first to go in a bad economy. You need a war chest
		- You have to be always marketing yourself. Some people say 25% of your time is selling.
* Software product – Small iPhone app companies to Electronic Arts
	+ Pros
		- Technology is the driver, so it’s treated very well.
		- Work with other very smart people
		- Pays well
		- In most cases, quite stable
	+ Cons
		- You have to stay on top of your relevant technology
		- Can be hard to transition if you don’t keep your skills up to date
* Large technology companies – Google to Microsoft
	+ Pros
		- So many different roles
		- Great infrastructure
		- Well respected
		- If you are really successful, can be very wealthy
		- You can change to different areas
	+ Cons
		- Lost in the noise
		- Feel like a number
		- Hard to move up
* Entrepreneur – You want to be the next Facebook?
	+ Pros
		- There is no better feeling than owning your own successful company
		- Huge potential profit. The richest people I know are the successful entrepreneurs
		- You can create things your own way
		- Huge learning experience. I’ve seen mediocre people become amazing after their first company
	+ Cons
		- You will fail. I’ve heard people say that successful entrepreneurs will fail 3 companies before they succeed
		- There is no worse feeling than going bankrupt. Really hard to rebound.
		- You will be poor for a long time, maybe until you give up and join the real world
		- Really hard to come up with a “great” idea. This is what I really struggled with.
		- Equally hard to see an idea through. I’ve heard the “I thought of that X years ago, why didn’t I do it” so many times.
		- You will do everything, including scrub the toilet of your first office
		- Even if you own your own company, you always report to somebody
			* Customers
			* Investors
			* Board
		- Work will be your life

So that is my 10 minute overview of the software industry. Any questions?