Research and Career Interests. I am interested in the field of Programming Languages; specifically, I want to work on Program Synthesis. After completing a Ph.D., I wish to become a professor.

Past Research and Industry. In my second year of undergrad, I participated in the "Early Research Scholars Program" (ERSP), a program run by Christine Alvarado, which is geared towards offering second-year undergraduates the opportunity to work with CSE professors. Through ERSP, I worked on a project in which we analyzed the bottlenecks of the `mmap()` Linux system call on different Linux kernel versions with Professor Jishen Zhao. The program showed me how to read research papers and introduced me to the world of CS research.

Despite enjoying the program, I was still convinced, for no particular reason, that industry was where I should end up. Instead of continuing research, I decided to go to the industry that summer and worked at Google instead. There, I worked on adding code coverage reporting for `KUnit`, a unit testing tool for Linux developers. I learned a lot that summer, like how to ask questions productively and how to effectively communicate technical information. But something seemed to be missing – I wanted to have more control over what I worked on and to build something that was properly exciting to me and not just something that was dictated by what my company wanted.

This yearning for something more pushed me to give research another honest chance. After my internship, I enrolled in Nadia Polikarpova's undergraduate Programming Languages course. This class was the first class in college that made me want to explore a subject beyond the scope of the class; I knew it was a sign. The following quarter, I enrolled in her graduate Program Synthesis course and started working on one of her team's projects, Hoogle$^+$.

Current Research. I have been working with Nadia now since January 2020 on this very project with another undergraduate student. Hoogle$^+$ is a tool that allows users to provide a type signature and optional examples and get back a code snippet that consists of a composition of Haskell library functions. We wrote an algorithm that serves as a simpler top-down enumerative alternative to Hoogle$^+$'s TYGAR algorithm. Because of its use of Petri nets to tackle the synthesis problem, TYGAR reduces the type system to first order. This means programs with λ-abstractions cannot be synthesized. Our tool, PETSY, aims to solve the same problem as TYGAR while also allowing for λ-abstractions in our synthesized programs. To speed it up, we added a memoization feature to cache previously computed subprograms. Recently, we submitted a 3-page abstract to the POPL 2021 Student Research Competition that details the intricacy of our technique, and plan to write a full paper when we improve upon and finalize our memoization approach.

What started as a class project unexpectedly turned into months and months of research. We originally assumed that since we were doing a top-down enumerative algorithm, we could simply extend already existing first order algorithms, like in the SYGUS setting. But we quickly realized that higher order and polymorphic queries could not be synthesized this way. This was mostly because function arity, which is a key part of the SYGUS-style enumeration search, is not fixed in higher order settings. Say, for example, we have the type query f :`Maybe` $(a \to b) \to x : a \to b$, and a component `fromJust` with type `Maybe` $\alpha \to \alpha$.

A solution to the query would be the program `fromJust f x`. Here, you can see that while `fromJust` seems to require only 1 argument at its definition, the given result in fact requires 2 arguments. After reading multiple papers on the subject, we eventually drew inspiration from SYNQUID's search algorithm and extended its type system. After many iterations, we finally got our algorithm working. I never thought the three simple words `fromJust f x` of a returned code snippet could bring me so much joy.

When we started exploring memoization techniques, we encountered a whole new set of challenges. To our knowledge, no one had ever presented an efficient memoization technique for polymorphic type-driven synthesis before. We drew our inspiration from the MYTH algorithm to get it working and extended its caching techniques to take into account polymorphism and the issues that arise with having free type variables in memo-map queries.

Becoming a Professor. After I complete my Ph.D., I want to become a professor. It wonderfully combines my passion for teaching and solving problems that many people don't even know exist.

Concerning the *research*, I find it extremely fulfilling. Research is what kept me going through the 9 months of being stuck inside because of the pandemic. Despite the civil unrest, political turmoil, and environmental disasters happening right outside my window, I continued to get up bright and early, log onto my computer, and work on my project. It gave me purpose, allowed me to control where my work went, and provided hope for a future in which maybe, amidst all the chaos, my tool would make someone's day that much easier. Program Synthesis has emerged as a field with the potential to completely rewrite the way we program and build tools. With program snippet automation, there are endless application domain possibilities, such as making a programmer's life more efficient or generating programs in non-programmer settings like FlashFill. I want to continue researching what these possibilities are.

Concerning the *teaching*, I think it is the best way to help shape the future. I became a tutor in college as soon as I could and am currently tutoring my 7th straight quarter. I have tutored classes that range from data structures to discrete math to programming languages. By far my favorite class to teach is the Intro to CS course. Each fall, I tutor over 600 students, most of whom are first-year CS majors who have no prior coding experience. This quarter, I am teaching that same intro class with Professor Sorin Lerner and am a head tutor to almost 40 undergraduate tutors. So many people drop out of the major after taking their first CS class, and I have made it one of my goals to make sure I encourage as many students as I can, particularly girls, to stay, by introducing complicated topics in more approachable ways and showing them that girls can code too. I want to continue this effort that I started as a tutor and turn it into my career.

Why [enter school name]. In this section, I wrote about what parts of the school appealed to me. Then I listed the professors I was interested in working with and what projects they were working on that sounded interesting. In total, this section was only about 5-6 lines, and one sentence per professor. I tried to list 2-3 professors.