

The following is the winning essay which
I wrote for the 1996 Buhler Memorial Scholarship:

**Question: What led
me to choose CS as a major and
what are my future career plans?**

by Geoff Fortytwo

Something big happened in 1985. Something happened that would have far reaching effects in the lives of everyone in my family and which would significantly alter the course of my own life. 1985 was the year that my parents bought an Apple //e computer. My life would never be the same. I was 10 years old when that historic computer entered my life. Up until that time the only computer I had ever seen was my neighbor's Commodore 64 which I had been rarely allowed to use.

My parents had purchased the computer for the usual reasons like word processing and record keeping and its use was subject to strict control because of my father's fear that the computer would break down if used too much. (That fear was apparently unfounded because that faithful old Apple //e is still working today, with no problems, over eleven years from the day we bought it.)

My first uses of the computer were to play games, but as time went on I discovered BASIC. It seems to be a natural evolution for young programmers to embrace some version of a Beginner's All-purpose Symbolic Instruction Code and just as natural for "adolescent" programmers to learn to despise it.

I felt pretty good about myself when I wrote my first BASIC program which printed out my name in an infinite scrolling loop along the left side of the screen. Later on I progressed to the point where I was able to use control and input statements to create a program called "ICE BREAKER" which was an interesting little program that gave the user puzzles to solve and codes to break.

One of the more complex programs I wrote in BASIC was an Apple //e version of the classic text game of Empire which I was introduced to on some TRS-80's that my middle school used to teach typing courses. My classmates were wowed when they saw the game because it was so much like the game on the TRS-80 yet it didn't have the problem of crashing after a short period of game play and it had a number of improvements that I had added.

By 6th grade I had pretty much decided that I was headed for a career as a hotshot computer programmer and as a true child programmer I was able to apply my programming knowledge to good use. I was able to get out of having to spend money on birthday and Christmas presents by writing

little programs that had my parent's name in them and displayed some sort of birthday or Christmas greeting. An example was one where I had an airplane fly across the screen on random lines with a message trailer behind it which said, "Happy Birthday Mom!"

BASIC carried me through my years of enlightened childhood, but sadly, at about the age of 15 I decided to put BASIC to rest. I had fallen in love with another language. I had become attracted to C. What attracted me most was the ability to separate code into things called functions. From my point of view today, I consider functions to be obvious basic necessities for programming of any type. Back then however, I was blown away by the utter coolness of functions. The only support for functions that Apple BASIC had at that time was a gosub instruction which allowed for separation and reuse of code, but all variables were still global! I attempted to buy a C compiler for the Apple //e, but when I asked the shop keeper at Walden Software if he knew where I could get one, he actually laughed at me! When he told me that none existed for the Apple I was left wondering what to do.

What I finally ended up doing was to get my parents to allow me to cash in the bonds that my dad had gotten using my birthday and Christmas money from years past and use it all to buy a computer. The computer turned out to be of rather poor quality, but it worked well enough to run the Borland C++ 2.0 compiler for DOS. I only used it to learn to program in C, but I was able to learn most of the basics of programming in a truly useful high-level language.

I wrote some simple programs such as "Mad Chad the Rad Bad Lad from Baghdad" and "Deadly Dungeon", but as I progressed I wrote more serious and goal oriented programs. I labeled one such program as a "Gravitational Vector Mapper" which computed graphical maps which showed the relative strength of the gravitational forces at different positions on the screen given a number of point masses at specified locations. I wrote the Gravitational Vector Mapper immediately before I entered college using C and a little bit of basic C++. At that point I knew what objects, templates, constructors, and destructors were and knew basically what operator overloading, and basic inheritance was. Beyond that I only knew that there was a lot more to learn about object oriented programming.

I continued to do some programming on my own outside of school, but it wasn't until I took a continuing education class on C++ that my eyes were opened to the true significance of object oriented programming. In that class I was enlightened about the wonders of dynamic binding and virtual inheritance 1. My interest in programming was energized and I haven't come down from the clouds since.

There are many reasons that I enjoy programming so much. The central reason seems to be that I'm addicted to the feeling of accomplishment that I get when I see a program that I've written running smoothly. The feeling that I've created something of my very own is one I cherish greatly. Beyond the innate pleasure of creating something, I also enjoy being able to show the result of my work to other people as proof that the many hours, days, and weeks that I've invested in my personal projects

have actually produced something interesting, tangible, and possibly even useful.

I've since upgraded my career goals from wanting to be a programmer to wanting to be a software engineer. A software engineer is actually a superset of a programmer. A software engineer is involved in both designing and implementing software whereas a programmer merely implements someone else's design or hacks code without any prior design at all. I've learned the hard way how design is truly an important step in the overall software creation process.

I hope that upon graduation from college I'll be able to start my career doing something that I perceive to be important to the overall field of computer science. I dread thinking about the possibility that I might get stuck as a low level grunt worker who does nothing but update old code for computers that are far past their prime. I dream of doing things which would significantly effect the industry as a whole, but I realistically hope to become part of a software engineering team which would work to produce complex software for knowledgeable computer users. The specific type of software which I would be helping to write isn't terribly important as long as the software itself is important.

I've come a long way since my days with Apple //e BASIC and I hope to go a lot further. I've put a lot of thought into where and with whom I would eventually like to start my career and I'm certain that I will put a lot more thought into it before I finally find my place. Above all else though, I will never be satisfied in any career unless I end up working on a team with many other talented, intelligent, hard working computer scientists whom I can truly respect and admire. If I can achieve that, then I will be truly happy.

FOOTNOTES

¹ Actually, at the time I wrote this essay I did not know that I was actually referring to polymorphism.