

AC 2007-1485: UTILIZING PROGRAMMING PROJECTS IN A FRESHMEN PROGRAMMING COURSE

Steven Lehr, Embry-Riddle Aeronautical University

Masters in Aerospace Engineering and Masters in Software Engineering. Associate Professor in Freshmen Program at Embry Riddle Aeronautical University College of Engineering and software consultant.

Christopher Grant, Embry-Riddle Aeronautical University-Prescott

Program Chair for the Freshmen Program Embry Riddle Aeronautical University College of Engineering

Utilizing Programming Projects in a Freshmen Programming Course

Background

In the fall of 2003, Embry-Riddle Aeronautical University formed separate colleges and the College of Engineering was born. One of the first initiatives of the college was to strive to have a common first year among all its engineering programs (Aeronautical, Civil, Computer, Electrical, Mechanical, and Software Engineering). Having a common year would allow first year engineering students to switch degrees with no impact to their schedule.

One course used by most engineering majors was “CS223 Computer Programming for Engineers” which was originally taught in FORTRAN then migrated to C in the mid 90’s. The course taught up to structures in C and was basically a C programming course taught by predominantly adjunct professors. The course had a high failure rate and did not expose students to MATLAB which many of the upper level engineering classes were starting to use. Additionally, this course was not being utilized by students in Computer and Software Engineering, though they had a pre-requisite that students must have had a programming course prior to entering the first JAVA course.

In the fall of 2003, I was asked to take the feedback from all engineering professors and come up with a course that would be the pre-requisite class to JAVA, and meet the needs of the upper level engineering professors. Talking with upper level engineering professors, we found that many students could not program. In fact, many of the students did not like programming. They had taken a course in programming in the long ago past, but their skills were how to code in C. The upper level engineering professors were using MATLAB, so first they needed to teach MATLAB, and then they could assign tasks and projects.

Analyzing the CS223 course, students had written programs, but usually small programs that focused on the topics covered in the textbook. The course covered good programming techniques, but not necessarily software engineering fundamentals. The new course would introduce software engineering techniques, expose the students to procedural programming, and add a MATLAB component. The course would be geared toward incoming freshmen students with little or no programming background.

EGR-115

In the Spring 2004, two experimental sections of EGR115 were taught. The focus shifted from coding in C to how to program. More software engineering techniques were applied and students were required to follow the software engineering life cycle: specification, requirements, design, code, and test. The topics in C were covered more generically, covering the fundamental principles of variables, loops, logic, functions, files, arrays, and structures. We first covered the fundamental principal in a language independent manner, then taught how to implement in C. Later in the term the same

topic was taught again using MATLAB. We eliminated the time consuming, complex elements of C – like base conversion, pointers, sorting algorithms, etc; and focused on the fundamental constructs common to all programming languages. To fill the gaps in teaching, we utilized programming projects. We no longer believed a student could become an expert C programmer in one semester, but rather believed in one semester we can introduce software engineering fundamentals, fundamentals of programming, and expose students to C and MATLAB. The course became nine weeks of programming concepts with C, two weeks of individual programming projects, and four weeks of MATLAB (the projects spanned six calendar weeks, though consumed two weeks of class time).

Programming Projects

We used two programming projects to solidify the fundamentals taught throughout the semester. The first project is a project called introduction to structures. Students had to analyze, describe, and then modify a C program that was coded for them. The other project was an individual programming project where the student had to come up with the problem, solve it, and document the complete solution.

Programming Project 1 – Modification of Existing Code Base

After teaching arrays and strings, we introduced the first programming project, introduction to structures. Students were given code that was very modular and reinforced all the concepts taught prior in the semester. The program read data from a file into an array of structures of the student type defined, and provided menus to add, delete, and sort the array of students. Each time the data in memory was modified, the program wrote all the data back out to disk, allowing the program to save its state. Students were asked to download and run the program, print out the source code, and to write out the pseudo code for what that program did.

This was the first “big” program the students were exposed to. Prior to discussing the operations of the program, the students analyzed the program and submitted the pseudo-code. After discussing the operations of the program, we then asked the students what was missing from the program, allowing them to think about all the features that could be added to the program. Each student was then assigned to document one feature to add to the code, then write the test cases to verify that the feature was added correctly, and then write the code and verify the code was written correctly.

The project introduced an important concept of modifying existing code and “adding a feature” to an existing code base. Additionally it provided an opportunity for them to see a large program that actually performed something meaningful; and contained all the constructs previously taught in the class (loops, arrays, files, functions, strings, and structures). Exposing the students to structures helped the transition from procedural to object oriented for those students moving on to JAVA and pursuing a computer science minor or a software engineering degree. Additionally, many students utilized the code base and/or concepts for their course project.

Programming Project 2 – Developing their own program.

Next students were assigned the class project: develop a software program to solve a problem which was important to student (see end of document for project assignment). We allowed the students to make their own decision on the program they wanted to create.

With less emphasis put on teaching “C”, it was crucial to add an individual programming assignment to the course. Rather than trying to come up with one problem and have all the students solve it, we let the students define the problem they wanted to solve. Letting the students decide created an inherent ownership. To further solidify the student ownership, we had them document why they wanted to solve the problem; according to Napoleon Hill, “if you have enough why you want to do something, persistence comes naturally”.¹

Frequently, open ended, self constrained problems do not sit well with freshmen. It helped to do some coaching, asking them if they have an aunt or uncle in business for themselves that may need something automated. Or ask if they have any hobbies or interests, perhaps they were involved in a student project. Perhaps an educational game to teach a younger sibling to count money, add, etc. We let them think about what they “want” to do. Within five minutes of talking with a student, a suitable problem can be found. Again we left it open ended and allowed them to choose what they wanted to do.

Incremental Deliverables

In software “Bing Bang” deliveries are frequently failures,² so rather than assigning a large project with one due date, we broke the assignment into three manageable, interim deliveries. Breaking the project down eased student anxiety, and made it easier for the students to get rolling, gain some confidence, and gather momentum.

Phase 1 Deliverable

Students were given one week to submit the first project deliverable, consisting of the first four tasks in the assignment: a title page, a short description of the problem they wanted to solve, a description of why they wanted to solve the problem, and a system level diagram. We made it easy for them to get rolling, first put together the title page. Now describe what you want to do. Why do you want to do this? Now draw a picture of the system you are proposing.

From the instructor perspective we collected the assignments and graded them on a done or not done basis. A homework grade was given for the first deliverable. The initial delivery provided an opportunity to determine whether or not the student could really solve the proposed problem. Student might not have the right tools, data, or expectations; so each project was reviewed and approved.

Any student not submitting the first assignment were immediately emailed and asked to submit the assignment. During the next class any student who still had not submitted an assignment were asked to come to office hours to find a project. Usually within a five minute conversation a suitable project was found that interested the student.

Context Diagrams

As part of the first deliverable and part of our course goals of introducing software engineering concepts, we had the students generate a context diagram of the system to be modeled. This diagram was constructed between the specification phase and the requirements phase of the software life cycle. The context diagram is a system level diagram, or a high level diagram, that shows the overall system, what it interacts with, and the data that flows from external entities into and out of the program.

To create a Context Diagram

1. Draw the big circle in center of page, and label the action the software is to perform inside the circle
2. Determine the external interfaces that the system will interface with (for example: user, files, network, etc). Draw a rectangle and label the external interface.
3. Draw arrows into and out of the circle from the external interfaces, and label the data passed in/out.

The context diagram is a valuable tool, it quickly shows at a high level what the system does and what the system interfaces with. Students find the diagram very useful as it quickly shows the boundaries and helps them visualize and clarify the system.³

Deliverable 2: Requirements, Algorithm, and Test Cases

The second deliverable was submitted about ten days from the first deliverable. Students submitted the requirements, pseudo code, and software test cases. The students documented their software requirements by listing the features/functions that the software must perform.

Next they wrote pseudo code for each feature's implementation. I often tell my students to write out a 10-Step Process to solve the problem. What must be done first, next, next, and so on. Labeled a 10-Step process because the students can quantify this, and many of the programs written to date could be summarized in about ten steps. Writing down the process helps the students with algorithm development and gets them thinking logically and sequentially.

Lastly we have them develop the software test cases. This is another software engineering concept, writing the test cases prior to writing the code. This helps to further clarify the problem. The idea is to think about how the code will be verified, before the code is written⁴. Knowing how the code will be tested prevents surprises later, and prevents code omissions. Verification should not be an adhoc process, you verify your design satisfies your requirements and you verify your code satisfies your design. Early

test case generation helps clarify requirements. As Elemer Magaziner says, “you know your code is complete when it passes all the tests”⁵. So, develop the test cases first, then testing becomes verification. The students document test cases by filling out the following table:

Test Number	Input	Expected Output	Actual Output	Notes

Instructor Feedback

This was a hard deliverable for the students; they really had to solidify what they were doing, and how they were going to put the pieces together. At this stage, they may or may not have solidified the final requirements, nor the final algorithm. It was quite likely that the content provided in this deliverable would change by the time of final project delivery. This deliverable was really about verifying student progression in the process and gently forcing them to make decisions. It was counted as a homework assignment and basically graded as either done or not done, with copious comments. Again any stragglers were contacted and prodded along.

Deliverable 3 – Status Report

One week prior to final project delivery students submitted a status report identifying how they are progressing with their projects, identified any unknowns or stumbling blocks, and gave an indication if they were on track to complete the project by the due date. Basically we want to know that the student knows the final project is due in one week.

Deliverable 4 – The Final Project

The final deliverable contains the complete project. Each of the interim deliverables were refined and updated, and resubmitted in the final deliverable. Students now had working source code. They printed out all the source code, completed the verification by filling in their test cases, captured screen shots of their final application, and spent some time doing postmortem. Possibly for the first time, students looked back and analyzed what they had done. Separate sections existed for their conclusions and lessons learned. The students had an opportunity to say what they learned, what they struggled with, and what they would do if they had more time. Students documented the limitations and assumptions designed into and made during the project. i.e. what did they “hard code”? What things does their software depend on? Lastly students were asked to summarize their projects into two sentences for additions to their resumes.

Instructor Feedback.

The final project was due the Friday prior to the last week of class. This was fair to the students because many students had last tests and final exams in the final week of class. When the final projects came in on Friday the students are pretty wiped out. Many of them had been up all night long to complete the project; having it due that Friday gave them the weekend to recuperate and did not interfere with final tests.

Projects were graded that whole weekend and returned back on Monday. Most students do more work than I could have ever assigned them, and the quality is generally high because they and their friends are the graders of the work. When I give all the projects back on Monday they are amazed that I have gone through all their projects and am giving them immediate feedback on their efforts. Here are some tips for project grading:

- Make lots of comments, this is the first thing students read, and really affirms that their efforts are respected.
- Grade each section as to whether or not it is done or not. Points have already been allocated to each section, and with many of these projects to go through make quick decisions and assessments.
- Focus on the project not just the code - in software development software is at most 20% of the work (Requirements, Design, Code, Test, and Delivery).
- Assess whether or not the student solved a problem and adequately documented the solution.
- Focus on what they said they were going to do, what they did, and what they learned.
- Examine their limitations and assumptions, do they make sense, do they understand the limitations.
- Examine programming constructs utilized and complexity of the code; compare with student in question. If student is a C student struggling to get through course and never programmed before, expecting them to use arrays of structures and pointers might not be realistic; though the use of loops, files, and functions would be expected; along with a well documented, complete project.
- If the student did what was required, the project is neatly put together, and the project make sense it is likely they will get a B. Above and beyond an A, lower than expected level for student in question a C. Most students receive an A or B.

Cheating

In my experience I have had three known cases of cheating (less than one per semester). When cheating is suspected, i.e. if they use complex logic not taught in class or beyond expectations of the student in question, type up a three question quiz based on the source

code of the project submitted. Call the student in to the office and have them take the written quiz on paper and describing why they did what they did. In two of three cases the students confessed immediately, and were handled appropriately. The other student confessed he got help from a senior, and demonstrated exactly how, where, and why the code was used.

Student Comments and Instructor Feedback

“Writing this program exposed me to how C-Structures work to a greater extent than was taught in the classroom”, Andrew M. Students do get much deeper knowledge of what they need and when they need it. This is similar to just in time production. The student actually did receive the information in class, but they really learned the material when he applied it and used it.

“I am really looking forward to taking more programming classes in the future but I will also program on my own time just for fun and to gain more experience.” Sebastian K. Students enjoy building something of their own creation. They really enjoy the freedom of building their own project.

“I developed a great product which I have tons of fun with. I learned a lot doing this project, such as being able to think in C and see what I am trying to do in my head before I tackle it. One lesson I learned was don’t wait so long to get started cause then you disappoint yourself more than anyone else.” Ryan D. This student really demonstrates the buy in and the ownership of the software he developed. He did a lot of struggling to develop his race car track, “crazy car”, and he persisted and he succeeded because it was his.

“Looking over it now, the program could have a cleaner function organization.” Matt G 2006. I see this often and since its only an introduction to programming course, I do not count it against the student when they could have coded something more optimally, I am happy to see that he saw there was a better way after he had reflected upon his own work.

“I had a great time programming my game. I also had a great time showing it to my friends.” Azden C 2004.” Some of the students get a bit obsessed with the code they developed, and spend hours on hours developing a big program. In fact one student who was almost failing, and could not get an idea for a project, and I asked him what he liked, Brian liked baseball. He went on to write a 2000 line of code C program that played a baseball game. Reading batting averages and pitcher ERA’s from files and computing random hits based on the hitters average, slugging percentage, and the pitchers ERA.

“When this end of semester project was assigned to me my first reaction was fear, completely gut-wrenching fear. I felt that I didn’t know enough to be able to write a program without any guidelines, any specifications, and mainly, without a friend an instant message away with the same problem.” Jane M. Through trial and error Jane came through just fine, she comments later about the confidence she obtained by struggling and persisting. *“I see it as proof that I have achieved more than I ever expected.”*

Conclusions

Projects are integral to our EGR-115 course, because it really closes the gap between teaching and learning. The students apply the concepts learned in class, then they go find out what they didn't learn to solve problem at hand. Since we can not cover everything in a 15 week course, we cover the fundamentals and provide examples and let them learn what they need to learn to solve the problems.

In doing this for 6 semesters now I have had 100% of the students actively participating in the class at the time of the test two on arrays and structures complete projects (well 188 out of 190; the two of them cheated and received F's, though they did submit a project).

Students are actually learning and they are actually programming. They develop useful programs that are meaningful to them. Most students do more work than I could have ever assigned them, and the quality is generally high because they and their friends are the graders of their work. In the end they are rewarded for their effort and they now have a project to put on their resume, an artifact to bring to an interview, confidence they can build a program, and skills they can take with them forever.

Attached at the end of the document are many student responses, they are claiming that they are learning. We gave them the freedom to choose what they wanted to do, and by doing so they spend a lot of time and effort getting it right. They take a lot of pride in their work, and they get the confidence that they can do it.

For some students, this is first project they have ever documented from end to end, from the problem statement to the design, to the construction, and verification. They spend time to reflect back on what they accomplished. Reward the effort, they have just learned far more putting a project together end to end than in the class, this is good this is what we were striving for; for the students to learn to program, get exposed to software engineering, and to learn to learn.

EGR-115 Final Project

The principal reason for this project is to utilize your acquired skills to solve a problem with a computer programming language that is meaningful to you. This project represents 20% of your final grade. Treat this as a project that you would bring on an interview with you someday, so professionalism is paramount. Add two bullets to your resume upon the completion of this class.

Computer Skills:

C & MATLAB Programming in MS Windows environment

Projects:

Designed and built a program in C to 500 LOC.

Final project requirements:

1. Cover Page (2 points)
2. Overview of the project containing (8)
 - a. Problem statement (few sentences describing the problem you are solving)
 - b. Why you choose this problem (this is something you are to take pride in, and possibly bring to an interview; because I had to is not pointworthy)
3. Context Diagram (labeling data flow in and out of the system and the data items passed between the external entities/interfaces and the software system). (5 points)
4. List functional requirement of software (10 points)
5. Pseudo code algorithm of the solution (5 points)
6. C Constructs utilized (files, functions, loops, PDA, etc) (5)
7. Test cases showing input, expected output, and actual output. (5 points)
8. Capture program execution (screen shots, or write to file, or redirect to file) of the actual test cases showing your program satisfied your test cases. (5 points)
9. Source code print out (10 points, 5 for proper indentation, 5 for comments)
10. Assumptions and limitations of source code (do this after code completed) (5 points)
11. Conclusion and Lessons learned (10 points)
12. If you had more time what more would you do (5 points)
13. Project summary line that you would put on your resume (modify my two statements above as to what you would put on your resume) (5 points)
14. Electronic copy of your source code on a CD or disk, no emails. (5 points)

The other 15 points

5 points professionalism and completeness

10 points degree of difficulty relative to student progress

Deliverables

Friday November 10 submit items 1, 2, and 3 (homework grade)

Friday November 17 submit items 4-7 (homework grade)

Monday November 27 submit status report (homework grade)

Friday December 1 final project due

Note: The project may change slightly from the original direction; you need to get instructor permission for large changes.

Abbreviated, Unedited Student Feedback, Reprinted Here with Students Permission

Writing a program takes a lot of time and planning, and then even more time. Along the way you'll run into problems and you won't know how to solve them. Planning is the key. Figure out what you're going to do before you do it. You need to do some research and learn new things so that you can solve your problems. It is impossible to know how to write an entire program at the start. You just need to start working on the problem and revise your logic again and again and then again some more. It is also hard to get exactly what you want from your program. There is a point where you know that you can optimize your code even more, but it is already good enough. Writing a program is a fluid process. You and your code are always changing; hopefully for the better. Phillip W

There is a process in creating a program that functions properly. Even though I had a good idea as to what I wanted the program to output, I did not have a concrete method to get that outcome. I had to make changes along the way. The final program is rather different from the program I initially had in mind. Hence, I have learned that flexibility is the key to C programming. You must be able to solve problems as they appear and change your code accordingly. You cannot spend a lot of time getting frustrated if something doesn't work the way you originally wanted it to. There is usually more than one way to accomplish the same task. These multiple methods become more apparent as you work through the problem. I've learned to keep working instead of giving up. Megan M

I developed a great product which I have tons of fun with. I learned a lot doing this project, such as being able to think in C and see what I am trying to do in my head before I tackle it. One lesson I learned was don't wait so long to get started cause then you disappoint yourself more than anyone else. The one thing I am disappointed in is the fact that I don't have a user input to control the car, but the betting game is still great. I am proud that I hit my head on the wall hard enough and long enough to "uncover the truth that was always there", Mr. Lehr. ... Ryan D

The lessons learned from this are that programming takes time, lots and lots of time. My initial plan for this program was to make it extremely modular, with a series of functions to be called up in the program. After rewriting my program 5 times I came to the conclusion that the C language did not read certain variables from function to function in a graceful manner. Although they did not all make it, the number functions I spent so many hours writing proved to be a learning experience in their own way. Elizabeth C

C is a very versatile language with which you can do more than I first expected. It is very convenient to being able to access the system commands, and to just write the output to a text file in HTML format, which just very conveniently displays your findings. This programming project helped me very much to develop my programming skills and my understanding of what C is capable of. I am really looking forward to taking more programming classes in the future but I will also program on my own time just for fun and to gain more experience. ...Sebastian K

Writing this program has further my knowledge in C. First I have learned about data structures. A data structure is programmer declared data type. I found this ability useful because I was able to load an array of structures from a file; this made accessing data much easier. I also began to learn how to design C programs that are menu driven. This skill is very useful, because writing C programs that only serve one purpose limit your capabilities.Andrew P

Time, patience, and dedication are necessary for any programming project to be successful. They must be planned out over a period of time, rather than done the night before in order for the project to have any possibility of being impressive. The project is only as good as the quality of work you put into it. Your attitude towards the project also affects the quality of its final outcome. This program is not just a final project, but a tool for me in the future. I have truly realized the power, convenience, and usefulness of the C++ programming environment and will be using it throughout my career. .. Matthew S

In conclusion, the horsepower calculator is a very handy piece of code that is of high interest to me, and some of my friends. The 'C' constructs this program has utilized may not have been the most complex, and I would like to have utilized arrays, in one way or another, but for this particular type of program it didn't seem helpful. Sending program output to the monitor seemed most practical for the given application and the types of people that I feel would use it. Sending to, and reading from files are very important constructs, that I feel I have good knowledge of but like I've said, given the application and the users, it didn't seem helpful. One important lesson I've learned on this particular project is the importance of testing and debugging several times along the way while writing the code. Overall, I feel I've probably gotten more out of this class than all of my other classes combined this semester. ... Brian M

"Code is only 1/5 of the programming process" – Professor SL

When this end of semester project was assigned to me my first reaction was fear, completely gut-wrenching fear. I felt that I didn't know enough to be able to write a program without any guidelines, any specifications, and mainly, without a friend an instant message away with the same problem. I made my context diagram; my ten-step process; my psuedo-code. I was doing well. My problems arose quickly though, once the actual coding started. I immediately had to scratch the initial idea of separating the compound into letters and numbers, because I didn't know how to do it. What I did know how to do was separate using spaces, like I did in the date program.

The main thing I have learned by doing this project is that by challenging myself I realized that I did know what to do and I tried my best to make it work but in the end I had to make my program work on my level instead of trying to make myself work at my program's level. Looking at my code and thinking back on the first weeks of class it hit me that I have learned so much. I went from not knowing what a Dos prompt was and not understanding a printf/scanf sequence to writing over 10 pages of code, most of which works. So I do not see this project as a failure at all, even if the end product wasn't exactly what I planned. Instead I see it as proof that I have achieved more than I ever expected. Jane M

Before starting work on the project, I felt that the goal I was trying to achieve was difficult, but not impossible. I felt that I had all of the knowledge required, and could solve this problem relatively easy. I quickly realized that this program was going to take a lot more thought than I had originally planned. It took me several tries to get past the first step, which was loading and saving the data to and from a file. I was forced to change my original design of logging in with the users name to using the account number. I was also having trouble organizing the order of calling different functions. However, I put in the extra effort and completed the program. I learned that I should take a little more time planning my design, either with a context diagram or drawing out the data flow. I also learned the value of learning from past programs. It was very helpful to view source code from other class programs to help understand what I needed to do. ... Steven P

While doing this program I have discovered that what we have learned in this class is not only valid to the problems in the text book but also to real life problems that have significance beyond the class. I am now able to use the capabilities of the computer to complete tasks and perform jobs as I was never able. We now have a new problem solving tool to which we will continue to improve upon and further our skills.

The program that I have written completes the goals that I set but I realize that it has a potential to do much more that would make it more valuable in the future. Actual boundary surveys are usually made from multiple reference points because of obstructions in the line of sight. I would later like to incorporate a way of switching reference locations when parsing through the input data to handle a more practical situation where not all vertices of the boundary can be seen from a single reference station. I would also like to have the program be able to deal with multiple boundaries and make calculations between the two. This would be useful because when doing a foundation survey it is necessary to locate the foundation on the property and assure that it complies with zoning law and regulations. Besides these improvements I feel my program does a good job of completing a useful task. I only wish I had this on the day last summer when today's technology did not cooperate. .. Aaron S

Bibliography

1. Think and Grow Rich, Napoleon Hill, 1960, Fawcett Books.
2. Introduction to the Personal Software Process, Watts S Humfrey, 1997, Addison Wesley.
3. Software Design, David Budgen, 1994, Addison Wesley.
4. Team Software Process, Watts S Humfrey, 1997, Manuscript.
5. "Better Software Requirements", Presentation an Rational Software Conference, Elemer Magaziner, Orlando FL, 1996.