Understanding, Grading, and Judging Computer Science Projects

---

Grading and judging computer science projects can be confusing, especially for people without a background in the subject. This article will help you understand computer science projects and how to evaluate them. It also includes a rubric for grading/judging.

Let's start by clarifying a common misconception: not all science fair projects follow the scientific method. Science fair projects usually fall into one of three categories, each of which follows their own process or method: science experiments, which follow the scientific method; engineering projects, which follow the engineering design process; and math projects, which follow the mathematical reasoning/proof process. Each of these types of projects is a valid science fair project.

The "classic" science fair project that follows the scientific method is the best understood and perhaps most common type of project. Nearly everyone learned the scientific method in school, so most people are quite comfortable with the scientific method and how to use it. However, often people, including teachers, are not familiar with the engineering design process. Like the scientific method, the engineering design process consists of a series of steps. This table summarizes the scientific method and the engineering design process.

The Scientific Method	The Engineering Process
1. State your question	1. Define a need
2. Do background research	2. Do background research
3. Formulate your hypothesis, identify variables	3. Establish design criteria
4. Design experiment, establish procedure	4. Prepare preliminary designs
5. Test your hypothesis by doing an experiment	5. Build and test a prototype
6. Analyze your results & draw conclusions	6. Test & redesign as necessary
7. Present results	7. Present results

What Are Computer Science Projects?

In some cases computer science projects might follow the scientific method or the mathematical reasoning/proof process, but computer science projects are usually engineering projects that follow the engineering design process. The difference between computer science projects, where the product is typically a program or improved functionality, and stereotypical engineering projects, where the product is usually a device of some sort, is analogous to the difference between an astrophysics project and a behavioral science project. Though the details of how the scientific method is used in a behavioral science study are different from the way in which the scientific method is applied in an astrophysics project, the scientific method is used during both projects. Similarly, the details of how engineering design process is applied during computer science and "typical" engineering projects are different, but the underlying process is the same.

How Does the Engineering Design Process Apply to Computer Science Projects?

Now we will take a step-by-step look at how the engineering design process applies to computer science projects. This discussion assumes that you are familiar with the engineering design process. If you are not already familiar with the engineering design process, or need a refresher, check out these resources:

http://sciencebuddies.org/science-fair-projects/project_engineering.shtml
http://synopsys.championship.googlepages.com/winningengineeringprojects
http://synopsys.championship.googlepages.com/engineeringdesignworkshop2

Step 1: State a Design Goal. The first step of the engineering design process is to define a need based on users' desire. This need is stated as a design goal. For example, the need may be to find a faster way to scan computers for virus and spyware infections. The design goal for this project might be "The goal of this project is to develop software that scans computers for virus and spyware infections faster than existing technologies."

Step 2: Background Research. The second step of the engineering design process is to do background research. The student should seek information related to their area of study (in our example, this would be antivirus/antispyware software). Students will probably use existing algorithms and frameworks in their project, so they need to understand these building blocks. For example, if compression or encryption is used, the student should understand the theory behind those concepts.

Students should also research the needs of their target user and the capabilities of existing software that may address the student's design goal. Becoming familiar with both their users' needs and the capabilities of existing products/programs will help the student establish meaningful and measurable design criteria.

Step 3: Establish Design Criteria. The third step of the engineering design process is to establish design criteria. Design criteria are requirements that help the student develop their software and determine the extent to which the final product/program meets the stated design goal. For our example, some of the design criteria might be (1) scans 1,000 files in 0.5 seconds with 99.5% detection accuracy, (2) performs a complete system scan in less time than Brand X antivirus/antispyware software, and (3) consumes less than 15 MB of memory while running.

Step 4: Preliminary Designs. The fourth step of the engineering design process is to draw up preliminary designs. In the case of a computer science project, this step usually involves writing the first iteration of the program's code. The student might write two or three (or more) completely different programs that go about reaching the design goal in different ways.

Step 5: Build and Test. The next step of the engineering design process is to build and test. For a computer science project, this step really only involves testing because the "building" (writing the first iteration of code) is done during the fourth step of the engineering design process. At this point in the engineering design process, the student tests the first version of their product or program. During this test, the student should note any bugs in the program, slow parts in the code, fast parts of the code, etc. A test plan is an important part of the testing process because it allows students understand the results of their test and use that information to improve future versions of their product. For our example, the test plan might include scanning 1,000 files on a specific computer. During the scan the student might note the speed of the scan, the amount of memory and processor speed the program uses, and the places where the program slows down/stops.

Step 6: Redesign and Retest. The sixth step of the engineering design process—redesign and retest—is usually the longest step of the engineering design process, and computer science projects are no exception. During this phase, the student works on debugging, rewriting, and optimizing the code. In doing so, the student should conduct several different tests of the code (remembering to use the test plan developed during the "build and test" step) and use failure analysis, the design criteria, and the design goal to guide revision of the code.

This step of the engineering design process involves iteration—repeatedly testing and revising the code until the stated design criteria and design goal are reached. Keeping an accurate and detailed record of this part of the project is essential to a high-caliber computer science project.

The redesign and retest step is an opportunity for a student to make sure they follow good programming practices. The coding style should allow make it easy for other programmers to understand the code. Students should also consider the programming language that they use. The student should strive to detect and account for all error conditions. Detecting and eliminating bugs are an integral part of the redesign and retest process. Making the program robust against errors is another good programing practice: can the program adapt to faults, or does it simply fail? Finally, the user interface should be easy on the eyes and easy to use.

Rewriting and revising code is the crux of a project, and the focus of this step of the engineering design process. Students should iterate—redesign and retest, debug, optimize, etc.—until the design criteria have been met and the design goal has been clearly fulfilled. (Unless, of course, your student runs out of time because the science fair is next week—in that case it would be best to stop and move on to step seven without achieving all the design criteria).

Step 7: Present Your Work. The seventh and final step of the engineering design process is to present the project. The presentation should highlight the final product, its usefulness, and merit *and* outline the process that led to that final product. All too often students doing engineering and computer science projects only address one of these. They either present the product without the process, in which case judges who prod into how the student actually obtained that product to doubt the final products' merit. Some students present only the process, in which case the final product, which may be quite outstanding, isn't appreciated.

A winning computer science project highlights both the product and the process. The product/program that truly addresses a meaningful need will attract judges' attention and a thorough design process will provide convincing evidence of the project's quality.

How Do I Grade These Projects?

Once teachers and judges understand how the engineering design process applies to computer science projects, grading and judging these projects should be easier. You are welcome to use the rubric at the end of this article. The rubric is detailed in order to help those who are unfamiliar with computer science projects. It is hoped, however, that even seasoned teachers and judges will find the rubric helpful.

Acknowledgments:

This article includes ideas from Bruce Kawanami, Allen Wallace, Terik Daly, SCVSEFA, and Science Buddies.

Rubric for Evaluating Computer Science Projects

A high-quality computer science project not only provides an innovative and useful product; it also shows evidence of a logical, structured design process.

Process (60 points)

Did the student state a specific design goal? Does that statement clearly identify the product/program to be developed? The need the product/program will satisfy? Does the design goal identify the target user? Did the student achieve the stated design goal?

1 2 3 4 5 6 7 8 9 10

To what extent did the student conduct background research? Did this background research address all important facets of the project (science concepts, mathematical formulas, existing products/programs, etc.)? Does the student understand the theory behind the algorithms and frameworks used in the project? Is borrowed code clearly identified and cited?

1 2 3 4 5 6 7 8 9 10

To what extent did the student develop meaningful design criteria? Did the student keep the target user in mind when developing these criteria? Did these design criteria guide the student in building/programming, testing, and revising the product/program? Were the design criteria met?

1 2 3 4 5 6 7 8 9 10

Did the student evaluate multiple approaches to solving the problem/filling the need? Can the student justify the chosen approach? To what extent did the student develop a test plan for evaluating each iteration of the program/product? Did the student follow this plan when testing the initial program/product design and subsequent designs? Did the student follow good programming practices (documentation, code readability, error checking, error recovery, user interface, etc.)?

1 2 3 4 5 6 7 8 9 10

Did the student use information from testing to improve the product/program? To what extent did the student redesign and retest the product/program until the design goal and design criteria were reached (e.g. through debugging, optimizing, etc.)? Did the student use graphs, mathematical analysis, and other appropriate measures to thoroughly evaluate the results of each iteration? Did the student use failure analysis? Were all possible error conditions detected and accounted for? Can the program adapt to faults?

1 2 3 4 5 6 7 8 9 10

Does the student's project notebook, display, and oral presentation provide ample evidence that the student used the engineering design process throughout the project? Is the project more than gadgeteering?

1 2 3 4 5 6 7 8 9 10

Score /60

Product (40 points)

In *your* experience, to what extent does the student's product/program represent significant improvements over existing products/programs?

1 2 3 4 5 6 7 8 9 10

Does the *student* understand the extent to which the product/program represents significant improvements over existing products/programs?

1 2 3 4 5 6 7 8 9 10

To what extent is the final product useful to the target user? Does the project fill a meaningful need?

1 2 3 4 5 6 7 8 9 10

Is the program code or product design clear enough that others would be able to replicate the student's work?

1 2 3 4 5 6 7 8 9 10

Score /40

Presentation/Interview* (10 points)

To what extent does the student's presentation/interview communicate both the merits of the final product/program and the process that the student went through to reach that final product/program?

1 2 3 4 5

To what extent can the student communicate effectively about the project? Can the student provide cogent responses to questions? Can the student defend the design choices that s/he made?

1 2 3 4 5

Score /10

Comments:

TOTAL /100

*If the grading is done based solely upon the student's display board, then the points in this section may be based on the display board alone.