

## Grading Codes for COMPSCI 119

When your programming assignments are to be graded, the TAs and UCAs will start with 10 points (the maximum possible) and then will remove points for each infraction that they find. Some infractions are weighted more heavily than others. When you get the grade on an assignment, there will be both the numerical score and a set of letters indicating the errors found in grading. Each infraction is indicated by a unique alphabetic error code. Some error codes are common to all programming assignments; others are unique to each particular assignment.

### A. -10 Student's name and other information was not in a comment at the top of the program

If you don't sign your work, you don't get credit for it. Period. Whenever you write a program, the first thing you do is to put a comment starting with a # symbol as the first line of the program. This should be automatic with you.

In particular, you must put **your name**, the **lab assignment number**, the **date**, and if you are resubmitting a lab that you've already turned in a note saying that this is a resubmit. For example:

```
# William Verts, Lab #1, January 31, 2020
```

*or*

```
# William Verts, Lab #1, February 3, 2020 (REPLACES EARLIER LAB #1)
```

### B. -10 Program cannot run due to syntax errors

If Python can't handle it, you don't get credit for it. Period. Don't turn in a program that contains any syntax errors. We won't debug your program for you after it is turned in.

### C. -10 Program crashes before finishing

The program runs up to a point, but crashes somewhere. If it doesn't run to completion, you don't get credit for it. However, it is possible that 99% of the program runs nearly to completion and prints out most or all of what is required, and then crashes on one of the very last steps. We *may* be lenient in that case depending on where the crash actually occurred and how much output was produced, but don't count on it. If there are multiple pathways in your program from start to finish, you must test all those pathways. If most work but some don't, we *may* be lenient, but again, don't count on it.

### D. -10 Program runs to completion but does not solve the intended problem

The program runs without crashing, but does not solve the assigned problem at all. This should seem fairly obvious, but it has happened in the past that a student expected credit for turning in a working program that solves some problem other than what was assigned. Bzzzt. Thanks for playing.

**E. -5 Program runs to completion but does not solve all of the assigned problem**

The program as submitted is a noble attempt to solve the assigned problem, and while the program code that is present may or may not work as intended, not all of the required functions have been completely written, or they may be missing entirely. This is unfinished program code. (Note that this is different from completed functions that have been written but have not yet been debugged; see below.)

**F. -5 Program uses advanced methods not covered in class**

Some students already know some Python. That's cool. However, I have very specific reasons for introducing the material in the order that I do, and I want students to use *only* the material I've covered up to that point. If you use a function or technique that is (A) not something I will cover or have not yet covered in class, (B) not in the *Computer Science Companion*, or (C) something I've never even heard of, then you won't get credit for it.

**G. -5 Program works correctly but changes the assignment in order to do it**

In a number of assignments I will provide a programming framework for students to follow and fill in with their own program code. That framework *must not* be changed. If you change my program code to get your own code to work, you will lose credit for it.

**H-Y. -1 Program runs to completion but contains bugs**

The program is basically complete, but some functions don't work as required. This can be as simple as a formatting issue when printing, an incorrectly computed value, an incorrect method, or something similar. Each infraction will be assessed -1 point, and these will be specific to the assigned problem. Most people who turn in a program that basically accomplishes what is assigned but still contains some bugs will see codes H-Y (or maybe Z).

**Z. -1 Program contains an error not covered by codes H-Y**

I can't predict all of the ways that people will find to mess things up. If there is an error that the graders find which is not covered by codes A-Y, then code Z can be used to indicate this.

**SUMMARY AND CHECKLIST**

1. **Make sure your name, lab number and date are at the top of the program in a comment.**
2. **Make sure there are no syntax errors in your program before you turn it in.**
3. **Make sure your program runs to completion without crashing.**
4. **Make sure your program solves the assigned problem.**
5. **Make sure your program contains all the assigned pieces.**
6. **Make sure your program uses only the techniques I've covered in class.**
7. **Make sure you don't change my program framework in order to make your program work.**