CSC 181F Introduction to Computer Programming — Fall 1999/2000

Programming Assignment Guidelines[1]

General Remarks

The purpose of the programming assignments is for each student to learn:

1. *design* – How to correctly formulate algorithms to solve specific problems ;
2. *coding* – how to write those algorithms as programs in some programming language;
3. *testing* – how to test a program to provide convincing evidence that it solves the right problem correctly and
4. *documentation* - how to provide a clear, concise explanation of how a program works, how to use the program, and what problem the program solves.

Producing a correct working program is harder than it looks, and as the problems become more difficult, the difficulty increases even faster. Thus an important part of learning to program is successfully completing the programming assignments, rather than merely attempting them.

Note also that these skills are a prerequisite for more advanced courses. Consequently, *only working programs will be accepted for grading* . Programs with syntax errors or run-time errors are definitely not acceptable. However, programs that correctly solve a simplified version of the problem may be accepted for partial credit.

Every solution to a programming problem will be graded in the same way, as described below. Any exceptions will to these rules will be described in the individual assignment descriptions. If the solution handles only a simplified version of the problem, then the final grade will be proportionately reduced. **It is not intended that this general description of how programs are marked should encourage you to elaborate or extend the problem assigned. A clean and correct solution to the problem that was assigned is sufficient to receive full marks.** Some of the points listed below will not be applicable to specific assignments. In all cases. any specific instructions in the problem assignment sheet superceed these general guidelines.

Mark distribution

Programming assignments are graded using the following guidelines. **Weights given are only approximate.**

- Design and correctness (weight approximately 40 percent)
  - Does the program work correctly for all input cases including boundary cases?
  - Does the program give correct results for good input data and (if required by the assignment) appropriate error messages for bad input data?
  - Are functions and/or classes used to partition the program appropriately?
  - Are the algorithms used appropriate to the problem? Is the program efficient in terms of time and space?

[1] adopted from *T.E. Hull et.al.,* Computer Science Introductory Handbook, Dept. of Computer Science, University of Toronto, August 1985

  - Is the output self-explanatory and formatted so that it can be easily read and understood?

- Coding style (weight approximately 15 percent)
  - Are the flow of control and logic of the program well organized? Is the code written so that the structure of the program is easy to understand?
  - Are program variables and data structures well organized and easy to understand? Is there a clear distinction between variables and constants? Are mnemonic names used for variables, constants and functions?
  - Are tricky programming practices and mysterious constants avoided?
  - Could the program be modified easily if one wanted to solve a similar problem?

- Testing (weight approximately 25 percent)
  - Is the test data adequate to demonstrate that the program works properly for a reasonable number of cases?
  - Have a sufficient number of boundary cases been tested? Is it evident that all parts of the program have been tested by at least some of the test data?
  - If the assignment requires it, has the program been tested with missing data or data of the wrong type?

- Documentation and comments (weight approximately 20 percent)
  - Is the documentation readable and legible?
  - Does it correspond accurately to the program that was handed in?
  - Do the comments aid in understanding the program rather than just echo the code?
  - Do comments point out key sections of code, indicate special cases, or make assertions?
  - Can one understand the program easily once the documentation is read?

Getting help

Some help in getting started with each assignment (and especially with understanding what is required) will be provided in lecture sessions. Answers to at least some of your questions can also be obtained from the teaching assistants during the tutorials and laboratory sessions.

Up to a point, you should also discuss assignments with friends and classmates *but only up to a point.* You should discuss and compare general approaches, and also how to get around particular difficulties. **But you should not leave such a discussion with any written material provided by another person, or with any material copied from the other person. What you hand in is expected to be your own work, and to be based on your own ideas or on ideas you have discussed in relatively general terms with others. To go further is plagiarism, copying or obtaining details from someone else's solution or program. Plagiarism is a serious academic offence, and we do not treat it lightly.** It is also a form of plagarism to submit test output that was not produced by the program that you hand in.

Handing in an Assignment

For each assignment, you should hand in the following

1. A standard assignment cover page that will be provided. This cover page identifies you and gives the teaching assistant a place to comment on your program.

2. A printed listing of the entire program including all source code files. If necessary, each part should be labeled.

3. The printed output from your testing of the program. Each output should be annotated (if necessary) with the *exact* input data that produced the output and a description of the purpose of the test.

4. External documentation for the program. Any materail that you have prepared that describes how the program is designed or how it should be used.

These three items *must* be *complete and consistent*. The program listing should correspond *exactly* to the program that produced the test output, the documentation *must* describe the program that was handed in.

You will usually be requested to submit an online version of the assignment so that your program can be compiled and tested using test data provided by the instructor. Instructions for submitting the online version of your assignment will be given out separately.

The due date of each assignment will be indicated on the assignment handout. Assignments are due at the **beginning** of the tutorial or lecture on the day that they are due. **Late assignments will NOT be accepted** unless there are extenuating circumstances (e.g illness, or serious personal problems.) If you realize that you will have to hand in an assignment late and think you have a legitimate excuse for doing so, get in touch with the instructor as soon as possible. Approval must be obtained from the Instructor for all late assignments. Excuses of the form "I was too busy with other courses" or "there was a big party last weekend" are **not** acceptable. The computer systems will be heavily loaded during the last day or two preceding assignment due dates. It is to your advantage to finish work on your assignments as early as possible. **Do not leave everything until the last minute!**

Record-keeping

With large classes, it is inevitable that some assignments will be mislaid, and some marks will be recorded incorrectly. Keep **all** the runs of your programs until well after the assignments are handed back marked; then when you say, "But I **did** do it, and I handed it in on time," you can prove it, and we can believe you *as we would like to be able to do.* (And further, if you have all the trial runs, no one can copy your program, and you can show us that you really did solve the problem yourself in case there is a dispute over "originality".)

It is also important to keep the marked copy of the assignment until you have received a final mark in the course, in case of mistakes in recording marks. Your teaching assistant will show you, or your instructor will post, copies of his mark records. **Check your marks to make sure they are recorded correctly**.