

IS 230 – Computer Programming I

Course Syllabus and Calendar – Winter 2000

Professor Don Colton

Brigham Young University—Hawaii Campus

Abstract

- **Course Number:** IS 230
- **Title:** Computer Programming I
- **Course Description:** An introduction to computer programming. Emphasis on fundamentals of structured programming design, development, testing, and implementation. Basic control structures of sequence, selection, and iteration. (Does not cover sequential file processing.)
- **Textbook:** *C for Business Programming*, by: John C. Molluzzo.
- **Class Time:** MWF 7:00–7:50 AM
- **Class Time:** MWF 2:00–2:50 PM
- **Final Exam:** before Apr 14, or 17 Apr 3–6 PM, or 21 Apr 7–10 AM.
- **Classroom:** GCB 140
- **Instructor (me):** Don Colton
- **My email:** don@colton.byuh.edu
- **My Office:** GCB 130 B, Phone: 293-3478
- **My Office Hours:** Daily 1–2
- **Teaching Assistant:** Margaret Miller
- **T.A. Hours:** Daily 8–10 PM
- **T.A. Room:** GCB 140 or 150

1 Early Completion Option

This course is being structured into a framework called “year round, open entry, open exit.” Year round means offered all the time, even between semesters. Open entry means you can add the course at any time, even if the add-drop deadline is past. Open exit means you can totally complete the course (and know your final grade) ahead of schedule. You don’t need to wait for the last day of class. It is an experimental concept.

We are still working on the “year round” and “open entry” parts of it, but “open exit” is working. You can take the final today if you like (but you only get one shot at it). Details below.

2 Why Take This Course?

In the old days (when I was young) IS professionals wrote programs. Today many IS professionals still write

programs, while many others do not but still must understand programming. For many, the focus of an IS professional’s life has shifted from COBOL and RPG to the Internet. Often programs are bought off-the-shelf and customized rather than being built from scratch.

This however does not remove the need for an understanding of what goes on in a computer, or what goes into a program. I believe there will always be many IS jobs that require programming as a routine part of their workday, and people who can program will be sought-after and respected (and employed). (CGI scripting and automation of web pages come to mind.)

This course and its successor (IS 231) will teach you to program well enough that you can easily learn any language employers want, now or in the future. The foundation of most modern languages is ALGOL, and the most popular and respected language of that class is C. **You will learn C moderately well in this course, and much better in IS 231.** With the skill at C learned in this course, you will be able to continue learning C, or learn C++, JAVA, PERL, COBOL, RPG, BASIC, or any of the other languages (including 4GLs) that are likely to be encountered in IS settings. You will know the fundamentals of computer programming. After these two classes, I believe those of you that get “A”s will be good enough to get entry-level programming jobs.

Knowledge of operating systems is also very important. Today’s client-side world seems dominated by Microsoft Windows, but there is a strong server-side presence from Unix. UNIX and Windows are the two operating environments that I believe will dominate the IS computing world in the next decade and beyond. Therefore, this class also introduces UNIX to a modest degree. You will learn the most commonly used commands, including those for file system maintenance (how to move from directory to directory, make new directories, move, rename, and delete files, etc.). You will learn to operate the most prominent free-software text editor, EMACS.

At the end of this course, you should feel comfortable listing C, UNIX, and EMACS among your skills on your résumé.

3 Prerequisites

There are no formal prerequisites for this class. To be successful you will need to use a computer, type, read, and recognize patterns in the things you see. I expect that you can manage your time well enough to get the work done, and not wait until the last week or two. I expect that you can avoid the temptation to cheat.

I assume you have **no experience** writing programs. We start from the very beginning in that regard. You must, however, be willing to work hard, two to three hours per class session.

4 Grading

Your grade is earned by getting points for completing labs and tests. When you have earned enough points, see me and I will certify your final grade. Once your is230 computer account is set up, progress reports are available to you by computer at any time.

dem programming labs	14	140 pts
prob programming labs	4	60 pts
pgm programming labs	22	400 pts
reading (textbook)	16	100 pts
quizzes (round 1)	8	100 pts
quizzes (round 2)	3	90 pts
final exam	1	100 pts
extra credit		60 pts
total points possible		1050 pts

930+	A	900-929	A-	870-899	B+
830-869	B	800-829	B-	770-799	C+
730-769	C	700-729	C-	670-699	D+
630-669	D	600-629	D-	0-599	F

Deadlines: Each assignment has a deadline. You can see these deadlines by sending email to GradeBot (see below) asking for a **status** report. Most deadlines are “soft.” Before the deadline an item is worth a certain number of points (100%). After the deadline, it is worth somewhat less each day until it reaches 60% of its original value. It then remains at the 60% level until the last day of class. All work must be completed by the end of the last day of class. The final exam has a separate deadline.

Incomplete and UW: If you quit working in the class before achieving a passing grade, I will probably give you a “UW” grade. In addition to saying that you failed the class, a UW also tells people that you didn’t seriously attempt the class; you just gave up.

I do not give “I” grades (incompletes) except in unusual circumstances. In my experience only a small fraction of incompletes are ever completed. I will consider giving you an incomplete if you request it, seem to have

a good reason, have a pretty solid timeline for completion, and you get the necessary paperwork filled out.

5 Work (No Pain, No Gain)

Most of your time will be spent writing programs. I estimate that a good student programmer could complete all of these assignments in about a month, working quarter time (as you should be). A professional could probably do most of them in one or two nights.

Since I do not assume you are a good programmer when you start, or even when you finish, I allow 3.5 months. Be aware that the work is only difficult because it is unfamiliar. Our task is to make it familiar, and therefore easy. You will find that assignments you did in three or four hours early in the semester can be done in just a few minutes late in the semester. You should feel a great sense of achievement.

If it takes you longer than others, remember that we do not all start with the same skills. About 10% of the students fail the class and take it again. This is not shameful, although it is tedious if you are one of those students.

Reading: The book is written for a typical student with no programming background. It will be too easy for many of you, and too difficult for some of you.

To get reading credit, you must let your sight rest on each of the words in the assignment, and you must try to understand what is being said. If you can speed-read some or all of it with reasonable comprehension, that is acceptable too.

Programming Labs: The key to this course is programming. That is the purpose of the class. That is your reason for being here. That is why you signed up. You want to learn to program. You will program. If you don’t want to program, why are you taking this class?

You will write about fifty simple programs, and test and submit them for grading. Each program must run perfectly (more on that below) before it will be accepted. Most students will submit a program five or more times before it is accepted. The overall average time spent fixing and resubmitting programs appears to be about 30 minutes per program.

Cheating: For some there will be a strong temptation to cheat by copying someone else’s program. The rationale is that since this class is so difficult, everybody else must be cheating too. Uh huh. This is not good for you. If you cheat in this way, what will you learn? You will learn that you cannot program. You will learn that you are a cheater. It will destroy your life. Count on it. Don’t cheat. Just drop the class instead.

Tests: There are eleven quizzes and one final exam in this class. All of them (including the final) are given at

the testing center using bubble sheets. You can complete the tests as soon as you want. I allow unlimited time and scratch paper, but no books, no notes, and no calculators. For each test, I will give you a sample test (with answers) that you can use as a study guide. You only get one chance to take each test. (If you feel there is some special reason you should get another chance, such as illness, discuss it with me.)

6 Lectures

See the Course Calendar at the end of this document for topics to be covered each day.

Attendance: The on-line **status** report gives you a timeline for your progress through the tests and lab assignments. If you are making progress, you are counted present whether you actually come into the classroom or not. If you stop making progress, and have not earned a passing grade, you will be counted absent. If you have not been making progress and do not have a passing grade by the end of the semester (or term), you will receive a grade of “UW” (unofficial withdrawl) instead of an “F.”

Due to INS (immigration) and VA (veterans) requirements the Vice President for Student Life is notified whenever a student misses four consecutive class days. In the context of this class, that happens when you are not actively working on assignments for a period of ten calendar days.

7 GradeBot (Yes Drill Sergeant Sir!)

GradeBot is my robotic program grader. It (he?) is available 24 hours a day, seven days a week, to grade and return your lab assignments. This is done via email.

I provide you with a computer account on the is230.byuh.edu UNIX host. This account gives you access to a UNIX system, software (including compilers and assemblers), email, and some storage. Most of you will use this account to do all the lab work in this class. See me if you need any help getting set up.

GradeBot is correct and authoritative. It is your boss. It is your client. It is your Drill Sergeant. There is always a particular correct behavior that it wants. You must make your program behave in exactly the way that GradeBot is requiring. This may involve changing the wording of your prompts and/or the spacing and wording of your output. It will not significantly alter the difficulty of the problem.

To submit a program to GradeBot, send it by email to <gradebot@gradebot.byuh.edu>. You can do this from almost anywhere on the Internet. The basic subject line for this class is “Subject: is230”. That will

get you a **status** report telling you everything you have completed, everything that is still due (and when), and what grade you have earned or are likely to get. To submit an assignment “x” to GradeBot, the subject line is “Subject: is230 x”. If you are having problems with extra stuff appearing after your program (such as an advertisement for junos or hotmail), you can put a “BEGIN” line before your program and an “END” line after it. GradeBot does not understand attachments; your program must be in the body of your message. Do not use any special encoding, such as HTML or MIME.

If you discover a case where you believe that GradeBot is wrong, tell me about it. If you found an error in GradeBot, I generally reward you with some extra credit. Otherwise, you must assume GradeBot is right.

8 Lab Submission Rules

I am unhappy when I see cheating in this class. Often these are cases where one student gives a copy of their completed program to another student, and the second student keys it in, possibly with minor changes, such as changing the names of variables. In worse cases, the second student uses cut-and-paste to copy the program, or sections of it. In almost every case, the second student does not understand how the program works, or why the program says what it says.

There are several rules that I use in this class. **These rules apply to the programs you submit to GradeBot.** They are designed to allow you to learn, but to prevent you from doing things that might let you pass the class without learning. Violation of any of these rules is typically regarded as a violation of the BYUH honor code. You will receive a score of zero for any such assignment, and it cannot be made up. Repeated violations may lead to failing the class. Please be careful what you submit.

The Keystroke Rule: Every keystroke in every lab you submit must come **from your own fingertips.** (If you are handicapped in some way that makes typing difficult or impossible for you, check with me. We can make a special exception for you if necessary.) You can re-use code that you wrote in a prior assignment (or in a prior class or in a prior job). You are forbidden to submit any code that was not typed by you yourself. You are permitted to copy things (particularly text strings) that GradeBot sends you in response to your submission.

The Open-Neighbor Rule: All labs are “open-neighbor” in the sense that you can **confer** with other students and lab assistants. You can read their code (if they let you). You can show your code to them. You can talk about your code, your approach, your difficulties, and your ideas. You can draw pictures and

make analogies and ask the TA or me (even me) questions. You can use their ideas. However, **you cannot make a copy of their code or submit their code to GradeBot, even if you first modify it.**

Never let another student take, borrow, or keep a copy of any program you wrote for this class. You can look at it **together**. If it is printed, please look at it away from any computers. If it is online, look at it on the author's own screen. Do not bring up a window on the second student's screen so they can look at the first student's program. You can talk about what the program does, and why it is that way. Do NOT leave them with a copy of your program.

If you receive a copy of a program from someone, and use it as the basis for the program you are submitting, you are cheating.

The Possession Rule: Except for the textbook, or handouts from me, you are not allowed to possess a copy of any lab program written by someone else until **after** you have earned credit for that lab yourself. If you ever obtain any such copies, you must permanently dispose of them or give them back BEFORE you work on your program again.

The Collaboration Rule: A small group of people who have not completed a particular lab assignment may collaborate (work together). Collaborate means you actively work out the solution together, maybe using one person's login account, and once the program is right, everybody else in the small group makes a copy and submits it. I require that every collaborative submission to GradeBot includes a comment near the top saying "The joint authors are: " and then listing by name (first name, last name, and login name) all the collaborators. Otherwise your submission will probably be regarded as cheating.

The purpose of collaboration must be to learn and understand. It must **not** be to merely get the work done. You must not submit as a collaboration anything that you cannot totally explain to me by yourself.

Once the group has gotten a version of the program accepted, no new people are allowed to join or use a copy of that program. It is impossible to collaborate on a program that is already finished. If you were not actively involved in creating the original work, you are **not** a collaborator.

The Looking Rule: Except for looking at the textbook, or things sent to you by GradeBot, or handouts from me, you are not allowed to look at your own code and somebody else's code at the same time, until after you get credit for that lab.

The Challenge Rule: If I think that you may have violated these rules on some particular assignment, I

will ask you (by email or in person) to state that you followed these rules. If I don't hear back from you, I will assume that you cheated and set your grade to zero for that assignment.

9 Types of Programming Labs

There are two types of programming labs: examples (named in the book demX-X or probX-X) and real work (named pgmX-X). Source code for examples is given right in the textbook. For real work you must invent it yourself.

dem Labs: The purpose of example labs is to encourage you to key in a fully-operational program and make it work. Why would it not work? There may be some small errors in the original program. Perhaps you will make a few typographical errors as you key it in. After submitting it for grading, you may want to "play" with the example program, changing various things to see what effect they have. In the end, you will learn good programming style and you should remember programming concepts better because you have worked through a detailed example.

On the example labs, you are permitted to submit a different program than the one shown in the textbook as long as it works the same. You can take this as a challenge to see if you can improve on the book version in various ways. Can you write the program in fewer lines? Can you organize it in a different way? But you can always fall back on the version in the book.

prob Labs: Prob labs are just like dem labs but they are generally longer and more complicated.

pgm Labs: The purpose of real-work labs is to experience programming and grow thereby. Programming can be an extreme joy, where time ceases to exist (e.g., hours pass quickly but you don't notice). It can be a great pleasure to cause a machine to produce reports and process data at your will. Or it can be a nightmare, where nothing seems to work right, and the most insignificant things turn out to have far too much significance, and you pull out great clumps of your hair and hit your head against the wall and you are glad that not every IS professional needs to be an accomplished programmer. Labs reflect the true reality of a programmer's life. You should experience labs.

Programming Labs, Approach: I believe in the successive refinement approach to programming because it keeps you from getting buried in details. In this approach, you (the programmer) are given general instructions for the behavior of the program. Once you

have the program running, additional details of the assignment will be made available to you. Generally these details include the exact wording of prompts, greater information about how to deal with error situations, or formatting requirements for your output.

This sequence is exactly like most programming jobs. Programmers are seldom if ever given a complete specification for any business program. Instead they are given a rough spec. Only after the program is up and running are they told the other requirements (the details) that apply. Build your programs in such a way that these modifications do not cost you much lost time.

10 Assignment Calendar

The dates on this list are not guaranteed. They are approximately correct. You should run a GradeBot status report to find the authoritative, correct due dates for you.

1: q01	thru Jan 08:	10 pt	(Jan 10: 9)
2: dem1-1	thru Jan 11:	10 pt	(Jan 12: 9)
3: q02	thru Jan 11:	10 pt	(Jan 12: 9)
4: ch1	thru Jan 13:	7 pt	(Jan 14: 6)
5: dem1-2	thru Jan 13:	10 pt	(Jan 14: 9)
6: q03	thru Jan 15:	10 pt	(Jan 18: 9)
7: pp1-6	thru Jan 15:	20 pt	(Jan 18: 19)
8: ch2	thru Jan 19:	7 pt	(Jan 20: 6)
9: dem2-1	thru Jan 19:	10 pt	(Jan 20: 9)
10: pp2-3	thru Jan 21:	20 pt	(Jan 22: 19)
11: pp2-11	thru Jan 24:	20 pt	(Jan 25: 19)
12: ch3	thru Jan 26:	7 pt	(Jan 27: 6)
13: dem3-1	thru Jan 26:	10 pt	(Jan 27: 9)
14: pp3-2	thru Jan 29:	20 pt	(Jan 31: 19)
15: pp3-3	thru Feb 01:	20 pt	(Feb 02: 19)
16: ch4	thru Feb 03:	7 pt	(Feb 04: 6)
17: q04	thru Feb 03:	30 pt	(Feb 04: 29)
18: pp4-7	thru Feb 05:	25 pt	(Feb 07: 24)
19: appb	thru Feb 08:	8 pt	(Feb 09: 7)
20: ch7	thru Feb 10:	8 pt	(Feb 11: 7)
21: dem7-1	thru Feb 10:	10 pt	(Feb 11: 9)
22: pp7-7	thru Feb 12:	20 pt	(Feb 14: 19)
23: pp7-8	thru Feb 15:	20 pt	(Feb 16: 19)
24: q05	thru Feb 15:	10 pt	(Feb 16: 9)
25: ch5	thru Feb 17:	8 pt	(Feb 18: 7)
26: dem5-1	thru Feb 17:	10 pt	(Feb 18: 9)
27: prob5-1	thru Feb 19:	15 pt	(Feb 22: 14)
28: q06	thru Feb 23:	10 pt	(Feb 24: 9)
29: pp5-9	thru Feb 23:	20 pt	(Feb 24: 19)
30: pp5-3	thru Feb 24:	20 pt	(Feb 25: 19)
31: q07	thru Feb 26:	10 pt	(Feb 28: 9)
32: ch6	thru Feb 26:	8 pt	(Feb 28: 7)
33: prob6-1	thru Feb 26:	10 pt	(Feb 28: 9)
34: pp6-1	thru Feb 29:	10 pt	(Mar 01: 9)
35: pp6-6	thru Feb 29:	15 pt	(Mar 01: 14)
36: q08	thru Mar 02:	10 pt	(Mar 03: 9)

37: ch8	thru Mar 02:	8 pt	(Mar 03: 7)
38: prob8-1	thru Mar 02:	15 pt	(Mar 03: 14)
39: dem8-1	thru Mar 04:	10 pt	(Mar 06: 9)
40: pp8-1	thru Mar 04:	10 pt	(Mar 06: 9)
41: pp8-9	thru Mar 07:	20 pt	(Mar 08: 19)
42: ch11	thru Mar 09:	8 pt	(Mar 10: 7)
43: dem11-7	thru Mar 09:	10 pt	(Mar 10: 9)
44: pp11-5	thru Mar 11:	20 pt	(Mar 13: 19)
45: pp11-7	thru Mar 14:	20 pt	(Mar 15: 19)
46: pp11-8	thru Mar 14:	10 pt	(Mar 15: 9)
47: ch12	thru Mar 16:	8 pt	(Mar 17: 7)
48: dem12-5	thru Mar 16:	10 pt	(Mar 17: 9)
49: dem12-6	thru Mar 18:	10 pt	(Mar 20: 9)
50: pp12-1	thru Mar 18:	15 pt	(Mar 20: 14)
51: pp12-11	thru Mar 21:	20 pt	(Mar 22: 19)
52: ch9	thru Mar 23:	8 pt	(Mar 24: 7)
53: dem9-2	thru Mar 23:	10 pt	(Mar 24: 9)
54: dem9-3	thru Mar 25:	10 pt	(Mar 28: 9)
55: dem9-5	thru Mar 25:	10 pt	(Mar 28: 9)
56: prob9-1	thru Mar 29:	15 pt	(Mar 30: 14)
57: pp9-9	thru Mar 29:	20 pt	(Mar 30: 19)
58: ch10	thru Mar 31:	8 pt	(Apr 03: 7)
59: prob10-1	thru Mar 31:	15 pt	(Apr 03: 14)
60: pp10-1	thru Apr 04:	20 pt	(Apr 05: 19)
61: pp10-4	thru Apr 04:	15 pt	(Apr 05: 14)
62: etut	thru Apr 14:	15 pt	(not later)
63: pp2-8	thru Apr 14:	5 pt	(not later)
64: pp2-9	thru Apr 14:	5 pt	(not later)
65: pp3-1	thru Apr 14:	5 pt	(not later)
66: pp3-7	thru Apr 14:	5 pt	(not later)
67: pp5-1	thru Apr 14:	5 pt	(not later)
68: pp6-12	thru Apr 14:	5 pt	(not later)
69: pp8-7	thru Apr 14:	5 pt	(not later)
70: pp10-12	thru Apr 14:	5 pt	(not later)
71: pp12-12	thru Apr 14:	5 pt	(not later)
72: q09	thru Apr 14:	30 pt	(not later)
73: q10	thru Apr 14:	30 pt	(not later)
74: q11	thru Apr 14:	30 pt	(not later)
75: final	thru Apr 21:	100 pt	(not later)

11 Office Hours

Office hours are posted outside my office door. I also have an open-door policy, posted on my office door as follows: "If my door is open (even just a bit) feel free to knock and come in. – Bro. Colton" Students for whom the posted hours are not convenient, or who just want a guaranteed appointment, can contact me by email to make an appointment.

12 Course Evaluation

The course evaluation for IS 230 is given in the testing center. You will be required to complete the course evaluation right after you turn in the last quiz. (The

testing center people will request you to do it then.) Your honest feedback (a) hopefully makes us feel good, (b) shows us where the course and/or instruction are weak, and (c) helps us improve the course for future students.

13 Subject to Change

It is very unlikely that I will make any changes, but I might. If my changes seem unfair to you, let me know. I will try to fix it.

IS 230 Course Calendar

The first day of class is an orientation to the class. During the next few class periods I demonstrate (using an overhead projector) how to do the lab work. This continues until most or all students have completed the first two labs.

After that, my theory of learning is that you know how to proceed. Indeed, I have discovered in past semesters that on average the students are so confident of their abilities that they ignore me and rush ahead into the assignments, with a goal to get enough points for an A so they can quit and think about something else. Good.

You will encounter problems that would be difficult to solve by yourself, so I and the TA are available to help you. But to get that help you must ask a question or state a request. For example, “Brother Colton, how do you do problem 17-2?” Each class period I will come to the class room and respond to these questions and requests. In that way, you the students will determine the topics to be discussed each day, in response to the deadlines provided in your GradeBot status reports.

- 05 Jan: first day of class
- 17 Jan: Human Rights Holiday
- 28 Jan: no class – School of Business special event
- 21 Feb: Presidents Holiday
- 27 Mar: Kuhio Holiday
- 01 Apr: General Conference
- 14 Apr: last day assignments can be turned in
- 14 Apr: last testing center final
- 17 Apr, 3–6 PM, first in-class final
- 21 Apr, 7–10 AM, second in-class final