IS 230 – Computer Programming I
Course Syllabus and Calendar – Spring 2001

Professor Don Colton

Brigham Young University—Hawaii Campus

1 Brief Overview

Computer programming is one of those exciting, high-tech jobs that can make you a ton of money. It also puts you in control of the destinies of thousands (when was the last time you were affected by a computer somewhere?). It leverages your own abilities by cloning a part of you to run on possibly many computers all hours of the day and night. While you sleep or play with your kids.

This course and its successor (IS 231) teach you to program well enough that you can easily learn any language employers want, now or in the future. And, if you like doing programming, that means a well-paying job.

1.1 The Course

- **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. ISBN 0-13-482282-X.
- **Class Time:** MWF 11:00 AM – 12:50 PM
- **Final Exam:** on or before Jun 15, 2001
- **Classroom:** GCB 140

1.2 The Instructor

- **Instructor (me):** Don Colton
- **My email:** don@colton.byuh.edu
- **My Office:** GCB 130 B, Phone: 293-3478
- **T.A. Hours:** Daily 8–10 PM
- **T.A. Room:** GCB 140

1.3 Grading

Your grade is based on points. 1060 points are possible, including 1000 assigned points and 60 extra credit points. The grading will be as follows:

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

In Fall 2000, the 74 grades were: `A` 48 (65%), `A-` 5, `B+` 4, `B` 2, `B-` 2, `C+` 2, `C-` 1, `F` 1, `UW` 9 (12%). My goal is that you will master the course material. If you do, you will probably earn an "A." Grading is discussed further below.

1.4 Office Hours

Office hours are currently MWF 1–2 and TTh 10–11 (except devotionals and such things). Updated office hours are posted outside my office door. Students for whom the posted hours are not convenient can contact me by email to make an appointment.

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"

1.5 Students with Special Needs

If you require accommodation for special learning needs or physical impairments, please see me as soon as possible.

1.6 Subject to Change

It is possible but very unlikely that I will depart in some way from this syllabus. If any of my changes seems unfair to you, let me know. I will try to correct it.

2 Early Completion Option

This course is being structured into a framework called "year round, open entry, open exit." Year round would mean offered all the time, even between semesters. Open entry would mean 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.

3 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.** 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 (even without graduating).

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 you résumé.

4 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. This seems to be true for about 80% of my students. We start from the very beginning in that regard. You must, however, be willing to work hard, two to three hours per class session.

5 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.

attendance	100 pts
reading (textbook)	100 pts
dem programming labs	150 pts
pgm programming labs	350 pts
quizzes (bubble sheet)	100 pts
quizzes (programming)	100 pts
final exam	100 pts
total points possible *	1000 pts

* I typically provide additional extra credit problems.

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. Remember also that computer labs and testing centers sometimes shut down early on the last day. The final exam has a separate deadline.

Incomplete and UW: If you quit working in the class and do not get at least 600 points (a passing grade), generally I will 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 vanishingly 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. I will not babysit you through the completion of an Incomplete, but I will assist you when you ask for help.

6 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.

Attendance: Each day of class I take roll. This serves several purposes, including helping me to learn your names. Students who arrive on time receive 5 points attendance credit for that day. I take roll again at 15 minutes into the hour for 4 points (80%) credit, and at the end of class for 3 points (60%) credit. If you come in late and want credit for attending, you are responsible for making sure I notice you.

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. Most IS students seem to like it.

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. It is a vital skill for IT professionals. That is why you signed up (or were forced to sign up). You want to learn to program. You will program. You may even like it.

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 apprears 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. Don't cheat. Just drop the class instead.

Bubble Tests: There are eleven quizzes and one final exam in this class. Most 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 make available 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 sudden illness, discuss it with me.)

Program Tests: A few quizzes will be given in class. These will require you to write by hand a program, or maybe several.

7 Lectures

Attendance: In the past attendance has been optional. This term I am making it required. I will physically take role each day.

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

8 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@is230.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 "xxx" to GradeBot, the subject line is "Subject: is230 xxx". If you are having problems with extra stuff appearing after your program (such as an advertisement for juno 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.

9 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.

10 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 properly. 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 you 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.

11 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.

date	day	num	topics, activities
04/25	Wed	1	Orientation, dem1-1, tc1
04/27	Fri	2	ch1, char, dem1-2, tc2, tc3
04/30	Mon	3	ch2, int, pp2-3, pp2-11
05/02	Wed	4	ch3, double, pp3-2
05/04	Fri	5	ch4, ++, pp4-7
05/07	Mon	6	appb, ch7, if, and, or
05/09	Wed	7	In-Class Quiz: flow charting
05/11	Fri	8	ch5, while
05/14	Mon	9	tc5, tc6
05/16	Wed	10	ch6, for, tc7
05/18	Fri	11	ch8, tc8, if
05/21	Mon	12	
05/23	Wed	13	ch11, ch12
05/25	Fri	14	In-Class Quiz: programming
05/28	Mon	-	Memorial Day
05/30	Wed	15	ch9
06/01	Fri	16	
06/04	Mon	17	ch10
06/06	Wed	18	
06/08	Fri	19	
06/11	Mon	20	printf
06/13	Wed	21	
06/15	Fri	22	In-Class Final: programming

You will encounter problems that would be difficult to solve by yourself, so I am available (especially in class) 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.

12 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: a02       thru Apr 27 (Fri)   5 pts
  2: a03       thru Apr 30 (Mon)   5 pts
  3: dem1-1    thru Apr 30 (Mon)  10 pts
  4: tc1       thru Apr 30 (Mon)  10 pts
  5: ch1       thru May 01 (Tue)   7 pts
  6: dem1-2    thru May 01 (Tue)  10 pts
  7: tc2       thru May 01 (Tue)  10 pts
  8: a04       thru May 02 (Wed)   5 pts
  9: pp1-6     thru May 02 (Wed)  15 pts
 10: tc3       thru May 02 (Wed)  10 pts
 11: ch2       thru May 03 (Thu)   7 pts
 12: dem2-1    thru May 03 (Thu)  10 pts
 13: a05       thru May 04 (Fri)   5 pts
 14: pp2-11    thru May 04 (Fri)  15 pts
 15: pp2-3     thru May 04 (Fri)  15 pts
 16: ch3       thru May 05 (Sat)   7 pts
 17: dem3-1    thru May 05 (Sat)  10 pts
 18: a06       thru May 07 (Mon)   5 pts
 19: pp3-2     thru May 07 (Mon)  15 pts
 20: pp3-3     thru May 08 (Tue)  15 pts
 21: a07       thru May 09 (Wed)   5 pts
 22: ch4       thru May 09 (Wed)   7 pts
 23: ic1       thru May 09 (Wed)  20 pts
 24: tc4       thru May 09 (Wed)  15 pts
 25: pp4-7     thru May 10 (Thu)  25 pts
 26: a08       thru May 11 (Fri)   5 pts
 27: appb      thru May 11 (Fri)   8 pts
 28: ch7       thru May 12 (Sat)   8 pts
 29: dem7-1    thru May 12 (Sat)  10 pts
 30: a09       thru May 14 (Mon)   5 pts
 31: pp7-7     thru May 14 (Mon)  15 pts
 32: pp7-8     thru May 15 (Tue)  15 pts
 33: tc5       thru May 15 (Tue)  10 pts
 34: a10       thru May 16 (Wed)   5 pts
 35: ch5       thru May 16 (Wed)   8 pts
 36: dem5-1    thru May 16 (Wed)  10 pts
 37: prob5-1   thru May 17 (Thu)  10 pts
 38: a11       thru May 18 (Fri)   5 pts
 39: pp5-9     thru May 18 (Fri)  15 pts
 40: tc6       thru May 18 (Fri)  10 pts
 41: pp5-3     thru May 19 (Sat)  15 pts
 42: a12       thru May 21 (Mon)   5 pts
 43: ch6       thru May 21 (Mon)   8 pts
 44: prob6-1   thru May 21 (Mon)  10 pts
 45: tc7       thru May 21 (Mon)  10 pts
 46: pp6-1     thru May 22 (Tue)  10 pts
 47: pp6-6     thru May 22 (Tue)  10 pts
* 48: etut      thru May 22 (Tue)  15 pts
 49: a13       thru May 23 (Wed)   5 pts
 50: ch8       thru May 23 (Wed)   8 pts
 51: prob8-1   thru May 23 (Wed)  10 pts
 52: tc8       thru May 23 (Wed)  10 pts
 53: dem8-1    thru May 24 (Thu)  10 pts
 54: pp8-1     thru May 24 (Thu)  10 pts
 55: a14       thru May 25 (Fri)   5 pts
 56: ic2       thru May 25 (Fri)  30 pts
 57: pp8-9     thru May 25 (Fri)  15 pts
 58: ch11      thru May 26 (Sat)   8 pts
 59: dem11-7   thru May 29 (Tue)  10 pts
 60: a15       thru May 30 (Wed)   5 pts
 61: pp11-5    thru May 30 (Wed)  15 pts
 62: pp11-7    thru May 31 (Thu)  15 pts
 63: pp11-8    thru May 31 (Thu)  10 pts
 64: a16       thru Jun 01 (Fri)   5 pts
 65: ch12      thru Jun 01 (Fri)   8 pts
 66: dem12-5   thru Jun 01 (Fri)  10 pts
 67: dem12-6   thru Jun 02 (Sat)  10 pts
 68: pp12-1    thru Jun 02 (Sat)  15 pts
 69: a17       thru Jun 04 (Mon)   5 pts
 70: pp12-11   thru Jun 04 (Mon)  15 pts
 71: ch9       thru Jun 05 (Tue)   8 pts
 72: dem9-2    thru Jun 05 (Tue)  10 pts
 73: a18       thru Jun 06 (Wed)   5 pts
 74: dem9-3    thru Jun 06 (Wed)  10 pts
 75: dem9-5    thru Jun 06 (Wed)  10 pts
 76: pp9-9     thru Jun 07 (Thu)  15 pts
 77: prob9-1   thru Jun 07 (Thu)  10 pts
 78: a19       thru Jun 08 (Fri)   5 pts
 79: ch10      thru Jun 08 (Fri)   8 pts
 80: prob10-1  thru Jun 08 (Fri)  10 pts
 81: pp10-1    thru Jun 09 (Sat)  15 pts
 82: pp10-4    thru Jun 09 (Sat)  15 pts
 83: a20       thru Jun 11 (Mon)   5 pts
 84: a21       thru Jun 13 (Wed)   5 pts
* 85: pp2-8     thru Jun 15 (Fri)   5 pts
* 86: pp2-9     thru Jun 15 (Fri)   5 pts
* 87: pp3-1     thru Jun 15 (Fri)   5 pts
* 88: pp3-7     thru Jun 15 (Fri)   5 pts
* 89: pp5-1     thru Jun 15 (Fri)   5 pts
* 90: pp6-12    thru Jun 15 (Fri)   5 pts
* 91: pp8-7     thru Jun 15 (Fri)   5 pts
* 92: pp10-12   thru Jun 15 (Fri)   5 pts
* 93: pp12-12   thru Jun 15 (Fri)   5 pts
* 94: phonecard thru Jun 15 (Fri)   5 pts
 95: tc9       thru Jun 15 (Fri)  15 pts
 96: final-ic  thru Jun 15 (Fri)  50 pts
 97: final-tc  thru Jun 15 (Fri) 100 pts
```