# IS 131 – Applications Program Development I
# Course Syllabus and Calendar – Winter 1999

*Instructor: Don Colton*

Brigham Young University—Hawaii Campus

## Abstract

- **Course Number:** IS 131
- **Title:** Applications Program Development 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. Sequential file processing. (Prerequisites: IS 190, 190L, and Math 100.)
- **Textbook:** *C for Business Programming*, by: John C. Molluzzo. (Cost: $58.55 at the BYUH bookstore, $55.00 plus 3.95 shipping at Amazon.com)
- **Sec 1 Class Time:** MWF 7:00–7:50 AM
- **Sec 1 Final Exam:** Wed Apr 21 7:00–10:00 AM
- **Sec 2 Class Time:** MWF 4:00–4:50 PM
- **Sec 2 Final Exam:** Wed Apr 21 7:00–10:00 PM
- **Classroom:** GCB 150
- **Instructor (me):** Don Colton
- **My email:** don@colton.byuh.edu
- **My Office:** GCB 130 B, Phone: 293-3478
- **My Office Hours:** MWF 10–11
- **Teaching Assistant:** Mac
- **T.A. Hours:** MTWThF 7pm-9pm GCB 150

## 1 Why Take This Course?

IS professionals must understand programming, even though many of them do not do it on a daily basis. I believe the focus of an IS professional's life has shifted from the early days of programming in COBOL and RPG to today's Internet environment where many 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, and I believe there will continue to be many IS jobs that require programming as a routine part of their workday. (CGI scripting and automation of web pages come to mind.)

The purpose of this class is to understand computers and programming well enough to be able to quickly learn any of the broad range of languages that now exist or will be invented in the foreseeable future. The foundation of most modern languages is ALGOL, and the most popular and respected language of that class currently is C.

**You will learn C moderately well.** With the skill at C learned in this class, you will be able to continue learning C, or pick up C++, JAVA, PERL, COBOL, RPG, BASIC, or any of the other languages that are likely to be encountered in IS settings. You will know the fundamentals of computer programming.

Additionally, the operating system in which a language is used can be very important. Today's client-side world seems dominated by Microsoft Windows, but there is strong server-side pressure from Linux, a free software UNIX. UNIX and Windows are the two operating environments that seem to me most likely to dominate the IS computing world in the next decade and beyond. Therefore, this class will also introduce 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.). In company with this knowledge, you will learn to operate the most prominent free-software text editor, EMACS.

We will also become familiar with the underlying hardware of the computer. We will study the reasons why base two is the natural counting system for computers, and we will learn some of the machine language in which computers do their thinking.

At the end of this course, you should feel comfortable listing C, UNIX (Linux), and EMACS among your skill set on you résumé.

## 2 Prerequisites

There is no need for prior programming experience. We will start from the very beginning in that regard. You must, however, be willing to work hard, five to ten hours per week.

I do expect that you can add and subtract reliably in base 10 without the aid of a calculator. I expect that you can read and hear English with good comprehension. I expect that you can be on time to class and can manage your time well enough to get the work done. I expect that your handwriting is legible to me. I expect that you can type on a computer.

## 3 Attendance

6% of your grade is based on your attendance. I give two attendance points per class session. The first is given precisely at the starting time of class (by my watch, which is generally within a second or so of the Department of Commerce's Atomic Clock in Boulder, Colorado). The second is given about midway through the class. There are 41 class sessions, for a total of 82 points. For full credit you need all 82 points.

Due to INS (immigration) and VA (veterans) requirements the Vice President for Student Life is notified whenever a student misses four consecutive class periods.

## 4 Lecture Style

My most important goal is that you develop intuition about the subject matter, and to get unstuck if you have become stuck. Accordingly, I devote as much time as necessary to the answering of your questions, especially when those questions seem to be of general interest to the others in the class. (Questions of narrow interest may be deferred to my office.) Otherwise, the time is generally spent in discussion of some topic or other that is closely related to the material designated on the course calendar. I may pose a problem to the class, and moderate as we work through it together. This can serve as a jump-start for understanding and mastering new material. My method of teaching is based on the view that learning is a shared activity between the teacher and the student, and that learning proceeds most quickly when interaction occurs.

## 5 Work Load

I mentioned five to ten hours of work per week. In the United States, the expectation for accredited university-level course work is that there be an average of three hours of work per week for every hour of credit awarded. In a lecture class this means one hour in class and two hours outside of class. It is my goal to keep the work load for an average member of the class at these levels.

Reading is about 500 pages. The course calendar lists by chapter the topics that will be discussed. To maximize your understanding and your grade, you should read the relevant sections before attending class and then review them again after class. Even if we do not discuss all parts of a chapter in class, you are responsible for its content on the exams. You might want to skim back through them the day before the exam.

What is the hardest part of the class? We will write a number of simple programs, and test and submit them for grading. For some people this will take a LARGE amount of time. Since you are probably new to programming, you may need the full two hours per class to come up to speed on the whole challenge of running and debugging programs.

## 6 Grading

Grades will be computed on the basis of points earned on attendance, reading, labs, and tests. I expect there will be 1000 points that you can earn throughout the semester, so 10 points will probably be worth 1%. The weighting is as follows:

| | |
|---:|---|
| 6% | attendance |
| 17% | reading |
| 4% | résumé |
| 17% | programming (examples) |
| 26% | programming (real) |
| 10% | midterm |
| 20% | final exam |
| 100% | total |

**Extra Credit:** Some extra credit assignments are available during the semester. For example, there are six 10-point extra credit programming assignments listed on the course calendar. The maximum extra credit that can be earned is 10%, even though more (or less) than that *might* be offered. 10% is enough to raise a B to an A.

There is no required "homework" in this class. If I give any homework, it will be for extra credit.

There are no required quizzes in this class. If I give a quiz, it will be for extra credit.

**Reading:** The purpose of the reading is to introduce you to business programming concepts in an orderly manner. The book will be excellent for the majority of students. If you find the book too wordy then skim it. If you find it too terse, check the library for other books, talk with your fellow students, talk with me, or all of the above.

Each day in class when reading is assigned (see the course calendar) I will ask each student whether they have completed the reading assigned for that day. There are seventeen such days. If you say "yes," you get ten points. If you say "no," then I will try to find out if you read part of the assignment, and give partial credit according to the amount you read (e.g., for a 20-page reading assignment, about one point for every two pages completed).

**Programming Labs, in general:** All labs will be "open-neighbor" labs in the sense that you can **confer** with other students and lab assistants. Within limits you can show your code to others and look at the code others have written. You can also talk about your code, your approach, your difficulties, and your ideas. You can draw pictures and make analogies and ask me (even me) questions. Limits:

> **Except for looking at the textbook, you are forbidden to look at your own code and somebody else's code at the same time.** (In order to copy something, you must memorize it and type it in yourself.)

Every keystroke in every lab 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 insert any code that was not typed by you yourself.

To do otherwise will be regarded as an honor-code violation.

**GradeBot:** Labs are graded by computer.

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

**Programming Labs (examples):** The purpose of example labs is to key in a fully-operational program and make it work. Why would it not work? 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 should remember the concepts better because you have worked through a detailed example.

On the example labs, you are generally permitted to submit a different program than the one given 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.

**Late Example Labs:** Example labs are generally not accepted after their designated due dates, since there is really nothing challenging about the lab that might make it take a surprisingly long amount of time.

**Programming Labs (real):** The purpose of real-work labs (pgmX-X) 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 major needs to be an accomplished programmer. Labs reflect the true reality of a programmer's life. You should experience labs.

**Late Real Labs:** I do accept late lab work up until the last day of class (April 16), especially because labs are famous for taking longer than it looked like they might take. **Lab work is due by midnight of the day designated.** Late labs are penalized as follows:

| late | worth |
|---|---|
| 1 class | 80% |
| 2 classes | 60% |
| more | 40% |

**Programming Labs (extra credit):** Extra credit labs (pgmX-Xe) work just like real-work labs, except that late work is not accepted. You can get a perfect score without doing any of the extra credit labs. If you think you can do an extra credit lab, give it a shot. It may be the safety net you find you need to keep your "A" at the end of the semester.

**GradeBot:** GradeBot is my robotic program grader. It is possible that gradebot will make some major crazy mistake. If you find an example of this, bring it to me. I will generally reward you with some extra credit.

Aside from such mistakes, you should assume that gradebot is correct and authoritative, much like your future boss or client will be. The assignments given in the book are generally not precise enough to have only one correct answer. However, gradebot has a particular correct behavior that he (it?) is looking for. This is not unlike getting specifications from a boss or client, then building a complying program, and then being told that it is not quite exactly what the boss or client wants. Life is tough. So, what you do in a case like that (there will be many) is make your program behave in exactly the way that gradebot is requiring. Most likely this will involve changing the wording of your prompts and/or the spacing and wording of your output. It should not significantly alter the difficulty of the problem.

**Tests:** The primary purpose of tests (examinations) is to gauge student learning by measuring performance in a timed and supervised situation. Some memorization may be required. I give out preliminary copies of all tests, including the final exam, so you will have some idea of what to study.

It is understood that such a situation creates additional stress for many students. For this reason testing is only 30% of the grading process. Each test will receive a scaled (normalized) score and a letter grade indicating the final course grade that would be earned by consistent performance at the level reflected on that test.

**Testing Center:** Except the final exam, tests are generally given in the BYUH testing center. (In this class there will only be one such exam.) The day of the exam we will preview it in class. You can take the exam that same day or the next day. Generally I allow unlimited time but no books, notes, or calculators.

**Test Makeup:** Exams cannot be made up except when I approve it in unusual circumstances. This is *very* rare.

**Grading Scale:** I use the following grading scale, both for individual assignments and for the course as a whole.

| | | | | | |
|---|---|---|---|---|---|
| 93%+ | A | 90–92.9% | A- | 87–89.9% | B+ |
| 83–86.9% | B | 80–82.9% | B- | 77–79.9% | C+ |
| 73–76.9% | C | 70–72.9% | C- | 67–69.9% | D+ |
| 63–66.9% | D | 60–62.9% | D- | 0–59.9% | F |

I occasionally up-scale the scores on an assignment, exam, or whatever, when I feel the absolute numeric grading would have been too harsh. The transformation may even be non-linear, but in any case, absolute rank order is preserved and no score goes down.

I welcome the opportunity to discuss specific instances of grading with you, and to hear your requests for different grades than were initially assigned. In fact, I encourage it. Some very good learning occurs in these settings (for you **and** for me). Historically speaking, about half the time I find that I end up agreeing with the request. The best time to do this is right after class, or during my office hours if the case seems more complicated. As an alternative, you can submit your argument in writing, together with the original graded work.

Final grades are generally issued by email soon after the final exam, or in-person if I feel that some discussion might be beneficial. Students are invited to visit my office to claim any exams or homework that I am still holding, and to discuss their academic progress. Interim progress reports are issued to the students several times during the course, generally after the tests and before the final exam.

## 7 Office Hours

Office hours are posted outside my office door. Currently they are Daily 1–2. Office hours are subject to change, as I might discover the need to attend some meeting somewhere, or visit the men's room, or talk to someone in the computer lab.

Students for whom the posted hours are not convenient, or who just want a guaranteed appointment, can come by whenever my door is open (which is most of the time) or contact me by email to make an appointment.

My "open-door policy" is 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"

## 8 Communication by Email

When I want to say something, or when you want to say something, if we are not in the same room, my first choice is to do it by email. I far prefer it to telephone calls, for instance. When there is an announcement, I will generally tell you in class or send it to you by email. Such announcements might include clarifications on the lab assignments. You must maintain an email account and to provide me with a valid email address.

## 9 Computer Accounts

You do (or shall) have a computer account on the is131 UNIX host. This account gives you access to UNIX systems, software (including compilers and assemblers), email, and limited storage. You will use this IS account to do the lab work in this class. See the teaching assistant if you need any help getting set up.

Programming homework is submitted by email from the student to me by sending it to "gradebot," who grades and returns such homework to the sender by email. You will use your is131 account for this.

## 10 Subject to Change

It is very unlikely that I will make any major changes, but aside from course number, title, and description, I reserve the right to change anything in this syllabus including the grading policies and the course calendar. Important changes are generally communicated in class and by email to those affected. If my changes are unfair to you, let me know. I will try to fix it.

## 11 Résumé Assignment

You will write a résumé for an IS position. Describe the self you want to be, or would need to be, to get such a job. `http://www.surfhawaii.com/classifieds` might get you started. Graded half on format, half on content.

## 12 Course Calendar

The course calendar is subject to change by me at any time for any reason. To avoid disruption and unhappiness, I avoid changes as much as I can. Any substantial change will be communicated in class and by email.

# IS 131 Course Calendar — Winter 1999

Probably reliable but subject to change.

| mtg | day | date | time | read | Topic | due (pts) |
|---|---|---|---|---|---|---|
| 1 | Wed | Jan 6 | 7am | | Syllabus, Getting a job in IS, résumé, telnet | |
| 2 | Fri | Jan 8 | 7am | A | Computers, base 2, ascii, hex | read A (10), résumé (40) |
| 3 | Mon | Jan 11 | 7am | 1 | Introduction, telnet, unix, emacs, gcc | read 1 (10) |
| 4 | Wed | Jan 13 | 7am | | unix, emacs, email, gcc | dem1-1 (10), dem1-2 (10) |
| 5 | Fri | Jan 15 | 7am | 2 | Integers | read 2 (10), pgm1-6 (10), dem2-1 (10) |
| | Mon | Jan 18 | | | **Holiday: Human Rights Day** | |
| 6 | Wed | Jan 20 | 7am | | Integers | pgm2-3 (10), pgm2-8e (10) |
| 7 | Fri | Jan 22 | 7am | | Integers | pgm2-9 (10), pgm2-11e (10) |
| 8 | Mon | Jan 25 | 7am | 3 | Real numbers | read 3 (10), dem3-1 (10) |
| 9 | Wed | Jan 27 | 7am | | Real numbers | pgm3-1 (10), pgm3-2 (10) |
| 10 | Fri | Jan 29 | 7am | | Real numbers | pgm3-7 (10), pgm3-9e (10) |
| 11 | Mon | Feb 1 | 7am | 4 | Arithmetic operators | read 4 (10), pgm4-7 (10) |
| 12 | Wed | Feb 3 | 7am | B | Program control, Structured programming | read B (10) |
| 13 | Fri | Feb 5 | 7am | 5 | Indefinite iteration (while) | read 5 (10), dem5-1 (10) |
| 14 | Mon | Feb 8 | 7am | | Indefinite iteration | prob5-1 (10), pgm5-1 (10) |
| 15 | Wed | Feb 10 | 7am | | Indefinite iteration | pgm5-3 (10), pgm5-9 (10) |
| 16 | Fri | Feb 12 | 7am | 6 | Definite iteration (for) | read 6 (10), pgm6-1 (10) |
| | Mon | Feb 15 | | | **Holiday: Presidents Day** | |
| 17 | Wed | Feb 17 | 7am | | Definite iteration | pgm6-6 (10), pgm6-12 (10) |
| 18 | Fri | Feb 19 | 7am | 7 | Program control (if) | read 7 (10), pgm7-3 (10) |
| 19 | Mon | Feb 22 | 7am | | Program control | pgm7-7 (10), pgm7-8 (10) |
| 20 | Wed | Feb 24 | 7am | 8 | Complex decisions | read 8 (10), pgm8-1 (10) |
| 21 | Fri | Feb 26 | 7am | | Complex decisions | pgm8-7 (10), pgm8-9 (10) |
| 22 | Mon | Mar 1 | 7am | | **Midterm** | **Midterm (100)** |
| 23 | Wed | Mar 3 | 7am | | Halloween Documents: www.opensource.org/halloween.html | read (10) |
| 24 | Fri | Mar 5 | 7am | 9 | Functions | read 9 (10), dem9-2 (10) |
| 25 | Mon | Mar 8 | 7am | | Functions | dem9-3 (10), dem9-5 (10) |
| 26 | Wed | Mar 10 | 7am | | Functions | prob9-1 (10), pgm9-9 (10) |
| 27 | Fri | Mar 12 | 7am | 10 | Modular design (and testing) | read 10 (10), prob10-1 (10) |
| 28 | Mon | Mar 15 | 7am | | Modular design | pgm10-1 (10), pgm10-12e (10) |
| 29 | Wed | Mar 17 | 7am | 11 | Arrays | read 11 (10), dem11-7 (10) |
| 30 | Fri | Mar 19 | 7am | | Arrays | pgm11-5 (10), pgm11-8e (10) |
| 31 | Mon | Mar 22 | 7am | 12 | Pointers and strings | read 12 (10), dem12-5 (10) |
| 32 | Wed | Mar 24 | 7am | | Pointers and strings | dem12-6 (10), pgm12-12 (10) |
| | Fri | Mar 26 | | | **Holiday: Kuhio Day** | |
| 33 | Mon | Mar 29 | 7am | 13 | Pointers, arrays, functions | read 13 (10), dem13-5 (10) |
| 34 | Wed | Mar 31 | 7am | | Pointers, arrays, functions | dem13-7 (10), dem13-12 (10) |
| 35 | Fri | Apr 2 | 7am | 15 | File processing | read 15 (10) |
| 36 | Mon | Apr 5 | 7am | | File processing | pgm15-2 (10) |
| 37 | Wed | Apr 7 | 7am | | File processing | pgm15-4 (10), pgm15-5e (10) |
| 38 | Fri | Apr 9 | 7am | | File processing | pgm15-7 (10) |
| 39 | Mon | Apr 12 | 7am | | tba | |
| 40 | Wed | Apr 14 | 7am | | Discussion: How to test and evaluate programs | |
| 41 | Fri | Apr 16 | 7am | | Review for Final, What is CS (vs IS)? | |
| | Wed | Apr 21 | 7am-10am | | **Final Exam, 3 hours, in the classroom** | **Final Exam (200)** |
| | Thu | Apr 22 | noon | | Final Grades by email (probably) | |