IS 231 – Computer Programming II
Course Syllabus and Calendar – Winter 2000

Professor Don Colton

Brigham Young University—Hawaii Campus

Abstract

- **Course Number:** IS 231
- **Title:** Computer Programming II
- **Course Description:** Emphasis on structured methodology of program design, development, testing, implementation, and documentation, including arrays, records, and pointers. (Prerequisite: IS 230.)
- **Textbook 1:** *C for Business Programming*, by: John C. Molluzzo.
- **Textbook 2:** *The C Programming Language*, by: Brian W. Kernighan and Dennis M. Ritchie.
- **Classroom:** GCB 140
- **Class Time:** MWF 10:00–10:50 AM
- **Final Exam:** 21 Apr 11:00–14:00
- **Instructor (me):** Don Colton
- **My email:** don@colton.byuh.edu
- **My Office:** GCB 130 B, Phone: 293-3478
- **My Office Hours:** Daily 1–2

1 Early Completion Option

I am restructuring this course 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 wouldn't need to wait for the last day of class.

I am still working on all these aspects. My hope is that before the course is too far spent, you will indeed be able to take the final early. Right now I cannot promise anything, but that is my goal.

2 Why Take This Course?

I assume that you want a programming job, or at least the ability to use programming in your future job. Some of you may be ready right now. At the successful conclusion of this course, many of you will be qualified to start work in most entry-level IS programming jobs.

We will develop your programming skills by completing projects in several areas, chosen based upon my experience as an IS programmer at Texas Instruments and as a consultant in Boston, and my years of teaching experience. We will use string functions to extract information from log files and printf to create reports for tracking and management. We will also organize and access complex data in memory and in files. We will develop your knowledge of a number of smaller topics that you will encounter in programming. We will improve your programming speed by holding some in-class contests.

This course and its predecessor (IS 230) 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. 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.

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 utilizes UNIX to a modest degree. You will know 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 know how 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.

3 Prerequisites

The prerequisite is IS 230 (Computer Programming I). I assume that you have written some programs. You

know how to use **printf**, **if**, **while**, **do while**, **for**, and subroutines. I assume that you have some skill, but you are not ready to sit in an interview and claim that you are a programmer.

We will use the C programming language and the UNIX operating system in this class. If your background does not include these, you will probably survive, but you may need to spend intensive time up front, the first week or so, to catch up.

4 Grading

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

dem programming labs	4	60 pts
pgm programming labs	25	465 pts
readings (usually chapters)	13	135 pts
tests	5	150 pts
final exam	1	150 pts
in-class contests	6	90 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 am not sure how much time it would take a good student programmer to complete all of these assignments. A professional could probably do all of them in less than a week.

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

Reading: Molluzzo 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. We cover the last three chapters in this class. The first twelve chapters are covered in IS 230.

Kernighan and Ritchie is about 200 pages long, divided into eight chapters. It is compact and excellent. It gets straight to the point, gives one good example (maybe two), and moves on to the next topic. Study it carefully. Note the questions that you have; we can discuss your questions in class.

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.

As you complete each assigned reading mentioned on the course calendar, notify me by submitting a program (cute, huh?) that tells me "I read chapter 1: A Tutorial Introduction of The C Programming Language (the White Book) by Kernighan and Ritchie." Or something like that. Email GradeBot (see below) for authoritative details. When you submit such a program, you are asserting that you have in fact done that reading.

Labs: The key to a programming course is programming. (Duh.) You will complete a number of programs, including six contest programs that are done in class. Programs are graded by GradeBot (see below). Each must run perfectly before it will be accepted.

Tests: There are five tests and one final exam in this class. All of them (except maybe the final) are given at the testing center. Some use 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 (usually 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 Calendars later in this document for topics to be covered each day. We will generally keep pace with the assignments.

Those who took IS 230 (IS 131) from me will wonder whether IS 231 will be the same. As you will recall, in that class each day I asked "Do you have any questions?" and then simply (or elaborately) answered them. Because students moved at vastly different paces, based on their vastly different backgrounds and aptitudes, the discussion was not always meaningful to everyone. Some were behind. Some were ahead. Some stopped attending because they got so far ahead. That was okay. In IS 231 (this class) I expect the differences will be much less. We are all "up to speed." We will have more that we can talk about together. It is more likely that the discussion will be meaningful to everyone.

I still like that general "got questions?" philosophy. It puts the responsibility for learning on the ones that are supposed to learn: the students. (I cannot learn for you.) Lectures can be fun and exciting, but often I have found myself simply presenting material that was already presented in the reading, which discouraged students from even doing the reading. This was not the best use of limited class time.

You do your part by reading, attempting the labs, deciding what questions to ask in class, and bravely asking them. I prepare the overall calendar, the syllabus, the list of assignments (and the GradeBot routines to grade them), and schedule the readings. I also prepare myself to answer whatever questions you can think of.

Attendance: Some may think by my policy in the following paragraph that I encourage absenteeism. I do not. I do encourage you to use your time wisely. For most of you, that means being present in class.

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, whether you actually come into the classroom or not. 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 approximately ten calendar days.

7 GradeBot

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: is231". With "Subject: is231 status" you get 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: is231 x". 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.

8 Lab Submission Rules

I do not expect that cheating will be a problem, but I have rules anyway. I am unhappy when I see cheating in any 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 (except contests on the first day of the contest) are "open-neighbor" in the sense that you can **confer** with other people. 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 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: 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 collaborating group makes a copy and submits it. In IS 230 I allow and encourage collaboration. In IS 231 collaboration is not allowed.

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 ppX-X). Source code for examples is given right in the textbook. For real work you must invent it yourself.

Example 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. 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.

Real 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.

Real 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 customer sees the program up and running are the other requirements (the details) discovered. Build your programs in such a way that these modifications do not cost you much lost time.

10 Contests

Six labs are designated as contests. I will disclose information about each of those labs at the start of class on the day the lab is due. (This means the labs cannot be completed in advance.) The goal is to improve your programming speed by forcing you to think and program in a higher-stress setting, such as you may encounter in some job interviews. (Yes, I have been asked to write sample programs during job interviews, and I have asked applicants to do the same when I was interviewing.) Although my goal is that you finish the lab in class before an hour is up, I do allow you to complete the lab later that same day for full credit.

11 Assignment Calendar

The names and dates on this list are not guaranteed, but they are approximately correct. You should run a GradeBot status report to find the authoritative, correct due dates for you. The wording in this list is condensed to make it fit the space available.

```
 1: hello    thru Jan 05: 15 pts (Jan 06: 14)
 2: kr1      thru Jan 06: 10 pts (Jan 07:  9)
 3: pp3-3    thru Jan 07: 15 pts (Jan 08: 14)
 4: kr2      thru Jan 10: 10 pts (Jan 11:  9)
 5: pp7-15   thru Jan 11: 15 pts (Jan 12: 14)
 6: gdb      thru Jan 12: 10 pts (Jan 13:  9)
 7: pp5-2    thru Jan 13: 15 pts (Jan 14: 14)
 8: kr3      thru Jan 14: 10 pts (Jan 15:  9)
 9: sumProd  thru Jan 15: 15 pts (Jan 18: 14)
10: q01      thru Jan 18: 30 pts (Jan 19: 29)
11: con1     thru Jan 19: 15 pts (Jan 20: 14)
12: pp8-7    thru Jan 22: 15 pts (Jan 24: 14)
13: pp11-8   thru Jan 25: 15 pts (Jan 26: 14)
14: pp12-12  thru Jan 29: 15 pts (Jan 31: 14)
15: kr4      thru Jan 31: 10 pts (Feb 01:  9)
16: pp10-4   thru Feb 02: 15 pts (Feb 03: 14)
17: jm13     thru Feb 03: 10 pts (Feb 04:  9)
18: dem13-5  thru Feb 04: 15 pts (Feb 05: 14)
19: q02      thru Feb 05: 30 pts (Feb 07: 29)
20: con2     thru Feb 07: 15 pts (Feb 08: 14)
21: dem13-10 thru Feb 08: 15 pts (Feb 09: 14)
22: kr5      thru Feb 09: 10 pts (Feb 10:  9)
23: pp13-5   thru Feb 10: 15 pts (Feb 11: 14)
24: pp13-13  thru Feb 12: 15 pts (Feb 14: 14)
25: jm14     thru Feb 15: 10 pts (Feb 16:  9)
26: dem14-3  thru Feb 15: 15 pts (Feb 16: 14)
27: kr6      thru Feb 17: 10 pts (Feb 18:  9)
28: dem14-4  thru Feb 17: 15 pts (Feb 18: 14)
29: q03      thru Feb 18: 30 pts (Feb 19: 29)
30: con3     thru Feb 22: 15 pts (Feb 23: 14)
31: pp14-7   thru Feb 23: 15 pts (Feb 24: 14)
32: pp14-11  thru Feb 24: 15 pts (Feb 25: 14)
33: jm15     thru Feb 25: 10 pts (Feb 26:  9)
34: pp15-7   thru Feb 28: 15 pts (Feb 29: 14)
35: kr7      thru Feb 29: 10 pts (Mar 01:  9)
36: kr8      thru Mar 01: 10 pts (Mar 02:  9)
37: pp15-9   thru Mar 03: 15 pts (Mar 04: 14)
38: etut     thru Mar 04: 15 pts (Mar 06: 14)
39: con4     thru Mar 06: 15 pts (Mar 07: 14)
40: pp15-10  thru Mar 08: 15 pts (Mar 09: 14)
41: pp15-11  thru Mar 09: 15 pts (Mar 10: 14)
42: name     thru Mar 10: 15 pts (Mar 11: 14)
43: crypt    thru Mar 11: 15 pts (Mar 13: 14)
44: q04      thru Mar 13: 30 pts (Mar 14: 29)
45: con5     thru Mar 15: 15 pts (Mar 16: 14)
46: web0     thru Mar 18: 30 pts (Mar 20: 29)
47: web1     thru Mar 21: 30 pts (Mar 22: 29)
48: web2     thru Mar 24: 30 pts (Mar 25: 29)
49: q05      thru Mar 28: 30 pts (Mar 29: 29)
50: con6     thru Mar 29: 15 pts (Mar 30: 14)
51: web3     thru Apr 03: 30 pts (Apr 04: 29)
52: zoo1     thru Apr 08: 30 pts (Apr 10: 29)
53: zoo2     thru Apr 14: 30 pts (not later)
54: final    thru Apr 21: 150 pts (not later)
```

12 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.

13 Subject to Change

It is very likely that I will make a number of small changes. If any of my changes seems unfair to you, let me know. I will try to correct it.

IS 231 Course Calendar

The first day of class is an orientation to the class. I demonstrate (using an overhead projector) how to do the lab work.

After that, I come prepared to talk about the current assignments, but my theory of learning is that you know how to proceed. Indeed, I have discovered in past semesters that some 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 am available to help you. To get that help someone 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
- 21 Apr, 11–2 PM, in-class final