IS 231 – Computer Programming II
Course Syllabus and Calendar – Winter 2002

Professor John Waite

Brigham Young University—Hawaii Campus

1 Brief Overview

Companies everywhere are eager to get onto the World Wide Web. They can choose to be present (but not special) with off-the-shelf software, or they can grab for market share with a more exciting and satisfying user experience, using home-grown or customized software. Programmers have become as vital as accountants in this new world order.

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. Perl (which is in the same language family as C, C++, and Java) is probably the most popular language on the server side of the web.

1.1 The Course

- **Course Number:** IS 231
- **Title:** Computer Programming II
- **Updated Course Description:** Emphasis on web programming (CGI, sockets), problem solving, stacks, queues, associative arrays, regular expressions, data manipulation, and simple algorithm analysis. Review of looping and precedence. (Prerequisite: IS 230 or equivalent.)
- **Required Textbook:** *Elements of Programming with Perl*, by: Andrew L. Johnson. ISBN 1-884777-80-5
- **Class Time:** MWF 7:00–7:50 AM
- **Final Exam:** Mon 22 Apr, 7:00–10:00 AM
- **Classroom:** GCB 140

1.2 The Instructor

- **Instructor (me):** John Waite
- **My email:** waitej@cs.byuh.edu
- **My Office:** GCB 101 (CS Lab)
- **Teaching Assistant:** ????
- **T.A. Hours:** Mon–Thu, 7–11 PM
- **T.A. Location:** GCB 101 (CS Lab)

1.3 Grading

Your final grade will be the **lower** of your total-points grade and your final-exam grade.

Your total-points grade is based on 1000 points of assigned work. Some extra credit points are also available. The total-points 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

To get an overall final grade above a C, you must also pass the final exam (five to ten programming problems, ten to twenty points each) with a sufficient score. The final-exam grading will be as follows:

93+	A	90–92	A-	87–89	B+
83–86	B	80–82	B-	77–79	C+

Grading is discussed further below.

1.4 Office Hours

As a first resort, you should see the TA for assistance. If you do need to contact me, email me to set up an appointment.

1.5 Special Needs

Brigham Young University–Hawai'i is committed to providing a working and learning atmosphere, which reasonably accommodates qualified persons with disabilities. If you have any disability that may impair your ability to complete this course successfully, please contact the students with Special Need Coordinator, Leilani A'una at 293-3518. Reasonable academic accommodations are reviewed for all students who have qualified documented disabilities. If you need assistance or if you feel you have been unlawfully discriminated against on the basis of disability, you may seek resolution through established grievance policy and procedures. You should contact the Human Resource Services at 780-8875.

1.6 Preventing Sexual Harassment

Title IX of the education amendments of 1972 prohibits sex discrimination against any participant in an educational program or activity that receives federal funds, including Federal loans and grants. Title IX also covers student-to-student sexual harassment. If you encounter unlawful sexual harassment or gender-based discrimination, please contact the Human Resource Services at 780-8875 (24 hours).

1.7 Subject to Change

It is possible that I will revise some aspects of the course as we go along. Any changes I make are likely to be to your advantage. If any of my changes seems unfair to you, let me know. I will try to correct it.

2 Now, About the 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. You will have programming proficiency in Perl.

We will develop your programming skills by completing projects in areas that support electronic commerce on the web. We will develop your knowledge of a number of additional topics that you are likely to encounter in 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 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 Perl, UNIX, and EMACS among your skills on you résumé.

2.1 What is the Course Like?

Much like IS 230, you will write programs that are graded by my robotic grader, GradeBot. Class time will be devoted to understanding basic concepts of computer programming as applied to the World Wide Web.

2.2 Prerequisites

The prerequisite is IS 230 (Computer Programming I). I assume that you have written some programs. You know how to do formatted printing, and use **if**, **else**, **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.

3 Grading

Your grade is earned by getting points for completing labs, readings, and tests. Once your computer account is set up, progress reports are available to you by computer at any time.

att	attendance	120 pts
pgm	programming labs	460 pts
qic	in-class quizes/final	140 pts
qtc	testing center quizes	160 pts
read	readings	120 pts
tot	total points possible *	1000 pts

* Extra credit is also available.

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 (usually one point) each day until it reaches maybe 60% of its original value. It then remains near 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.

The in-class quizes and the final exam are programming tests. You will be asked to write several fairly simple programs. The tests must be taken in class at the designated time. This is because anyone with advance knowledge of the test could memorize answers easily, so I must make sure no one has advance knowledge. If you must take the test at some other time, I must create a totally separate test for you.

Incomplete and UW: If you quit working in the class before achieving a passing grade, I will probably give you a "UW" grade instead of an "F."

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 time line for completion, and you get the necessary paperwork filled out.

4 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 a week. Maybe less. But you are not a professional yet. The work is difficult mainly 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 later in the semester. You should feel a great sense of achievement.

Reading: This is the first semester that I am using *Elements of Programming with Perl* by Johnson. I have read portions of the book and it seems quite good. I also received a very strong positive recommendation from someone I know. For these reasons I have adopted the book as the text for this class. I am especially interested in your feedback as students. I want to know whether you like this book.

Programming Perl by Wall, Christiansen, and Schwartz is a much more detailed treatment of the Perl Language. Larry Wall is the primary creator of the language, so this book is very authoritative and much more complete. It is also quite well written. If your programming skills are more advanced, you may want to look at this book. I believe it is available in the BYUH bookstore.

I do give credit for reading in the books. To honestly get reading credit, you must (at a minimum) let your sight rest on each of the words in the assignment, let the word pass through your mind, and 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 that tells me "I read chapter 1, Introduction, of Elements of Programming with Perl by Johnson." 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 what the program says about you.

Labs: The key to a programming course is programming. (Duh.) You will complete a number of programs. Programs are graded by GradeBot (see below). Each must run perfectly before it will be accepted. Half of the points for your grade come from these labs.

Tests: There are six tests given in the testing center using bubble sheets, including one that is comprehensive and is due the last day of class. 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, a sample test is available through GradeBot for you to 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.)

There are three tests given in class. These are programming tests where you will be asked to write some particular program without the aid of notes of a computer: written by hand, graded by hand. Time is limited. Scratch paper is provided. No books, no notes, and no calculators. No sample tests are provided.

5 Lectures

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, the GradeBot routines to grade them, and the schedule of readings. I also prepare myself to answer whatever questions you can think of.

I follow a general "got questions?" teaching philosophy. It leaves the responsibility for learning with the people that are supposed to learn: the students. (I cannot learn for you.) Canned lectures can be fun and exciting, but frequently the relevant material is in the assigned reading. Our class time will be focused on things you need to do the nearby assignments, or on explaining things that were not sufficiently clear from the reading.

Attendance: I take roll in this class. Attendance counts for 10% of your final grade. Typically attendance is worth 3 points per day. I take 3-point roll at the start of class. I take 2-point roll about 10 minutes into class. If you come later than that, you can get one point by making sure I notice you in class (maybe right after class). Missing and unnoticed persons get zeros.

Due to INS (immigration) and VA (veterans) requirements the Vice President for Student Life is supposed to be notified whenever a student misses four consecutive class days. I try to do this.

6 GradeBot

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

I enable you to have a computer account on the (gradebot.byuh.edu) Linux host. This account gives you access to a UNIX system, software (including compilers and assemblers), email, and some storage (type "quota" to see how much). You can also put up your own web page and CGI scripts. 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.

For grading, 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 demands. You must make your program behave in exactly the way that GradeBot is requiring (including spelling errors, if any). Be sure to look at a sample "conversation" with GradeBot before you start writing your program.

To submit a program to GradeBot, send it by email to `<gradebot@gradebot.byuh.edu>` (or just `gradebot` if you are already logged in there). 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 "xxx" to GradeBot, the subject line is "Subject: is231 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. Currently 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 or Rich Text.

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.

7 Lab Submission Rules

Cheating has never been a problem in this class, but there are rules. I am unhappy when I see cheating in any class. These may be 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. I consider any such behavior to be plagiarism and an honor code violation. I want you to learn, but not do things that might let you complete the assignments without learning.

Open-Neighbor versus Copying: All labs 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. Never 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.

8 Programming Labs

The purpose of lab work 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.

9 Course Calendar

Generally the lectures and discussion in class will follow the due dates for the various assignments (shown below). In-class tests will generally occur at the start of class.

10 Due Dates and Points

The names, dates, and points 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 08 (Tue)	10	pts
2:	aj1	thru Jan 09 (Wed)	5	pts
3:	aj2	thru Jan 09 (Wed)	5	pts
4:	aj3	thru Jan 10 (Thu)	7	pts
5:	a03	thru Jan 11 (Fri)	3	pts
6:	aj4	thru Jan 12 (Sat)	7	pts
7:	qprec	thru Jan 14 (Mon)	20	pts
8:	a04	thru Jan 14 (Mon)	3	pts
9:	aj5	thru Jan 15 (Tue)	6	pts
10:	sumProd	thru Jan 15 (Tue)	20	pts
11:	a05	thru Jan 16 (Wed)	3	pts
12:	a06	thru Jan 18 (Fri)	3	pts
13:	qloops	thru Jan 19 (Sat)	20	pts
14:	starbox	thru Jan 19 (Sat)	20	pts
15:	a07	thru Jan 23 (Wed)	3	pts
16:	lessthan	thru Jan 24 (Thu)	20	pts
17:	a08	thru Jan 25 (Fri)	3	pts
18:	qbigoh	thru Jan 25 (Fri)	20	pts
19:	cgi1	thru Jan 26 (Sat)	25	pts
20:	a09	thru Jan 28 (Mon)	3	pts
21:	aj6	thru Jan 28 (Mon)	6	pts
22:	a10	thru Jan 30 (Wed)	3	pts
23:	etut	thru Jan 30 (Wed)	20	pts
24:	qregex	thru Jan 30 (Wed)	20	pts
25:	cgi2	thru Jan 31 (Thu)	25	pts
26:	a11	thru Feb 01 (Fri)	3	pts
27:	a12	thru Feb 04 (Mon)	3	pts
28:	aj7	thru Feb 04 (Mon)	6	pts
29:	a13	thru Feb 06 (Wed)	3	pts
30:	aj8	thru Feb 06 (Wed)	6	pts
31:	a14	thru Feb 08 (Fri)	3	pts
32:	aj9	thru Feb 08 (Fri)	6	pts
33:	roman	thru Feb 11 (Mon)	25	pts
34:	a15	thru Feb 11 (Mon)	3	pts
35:	aj10	thru Feb 11 (Mon)	6	pts
36:	a16	thru Feb 13 (Wed)	3	pts
37:	a17	thru Feb 15 (Fri)	3	pts
38:	aj11	thru Feb 15 (Fri)	4	pts
39:	aj12	thru Feb 15 (Fri)	4	pts
40:	qic1	thru Feb 15 (Fri)	20	pts
41:	checkwr1	thru Feb 15 (Fri)	20	pts
42:	aj13	thru Feb 19 (Tue)	3	pts
43:	aj14	thru Feb 19 (Tue)	6	pts
44:	a18	thru Feb 20 (Wed)	3	pts
45:	checkwr2	thru Feb 20 (Wed)	20	pts
46:	aj15	thru Feb 21 (Thu)	4	pts
47:	aj16	thru Feb 21 (Thu)	3	pts
48:	a19	thru Feb 22 (Fri)	3	pts
49:	aj17	thru Feb 23 (Sat)	4	pts
50:	a20	thru Feb 25 (Mon)	3	pts
51:	checkwr3	thru Feb 25 (Mon)	25	pts
52:	aj18	thru Feb 26 (Tue)	7	pts
53:	a21	thru Feb 27 (Wed)	3	pts
54:	hfetch1	thru Feb 28 (Thu)	25	pts
55:	aj19	thru Feb 28 (Thu)	5	pts
56:	a22	thru Mar 01 (Fri)	3	pts
57:	qprintf	thru Mar 02 (Sat)	20	pts
58:	a23	thru Mar 04 (Mon)	3	pts
59:	hfetch2	thru Mar 04 (Mon)	25	pts
60:	a24	thru Mar 06 (Wed)	3	pts
61:	a25	thru Mar 08 (Fri)	3	pts
62:	href1	thru Mar 08 (Fri)	30	pts
63:	a26	thru Mar 11 (Mon)	3	pts
64:	a27	thru Mar 13 (Wed)	3	pts
65:	href2	thru Mar 13 (Wed)	30	pts
66:	a28	thru Mar 15 (Fri)	3	pts
67:	qic2	thru Mar 15 (Fri)	20	pts
68:	a29	thru Mar 18 (Mon)	3	pts
69:	href3	thru Mar 18 (Mon)	30	pts
70:	a30	thru Mar 20 (Wed)	3	pts
71:	a31	thru Mar 22 (Fri)	3	pts
72:	sitemap1	thru Mar 22 (Fri)	30	pts
73:	a32	thru Mar 25 (Mon)	3	pts
74:	sitemap2	thru Mar 27 (Wed)	30	pts
75:	a33	thru Mar 27 (Wed)	3	pts
76:	a34	thru Mar 29 (Fri)	3	pts
77:	a35	thru Apr 01 (Mon)	3	pts
78:	sitemap3	thru Apr 01 (Mon)	50	pts
79:	a36	thru Apr 03 (Wed)	3	pts
80:	a37	thru Apr 05 (Fri)	3	pts
81:	a38	thru Apr 08 (Mon)	3	pts
82:	a39	thru Apr 10 (Wed)	3	pts
83:	a40	thru Apr 12 (Fri)	3	pts
*84:	preced	thru Apr 13 (Sat)	30	pts
*85:	index	thru Apr 13 (Sat)	30	pts
*86:	nkrypto	thru Apr 13 (Sat)	20	pts
*87:	robot	thru Apr 13 (Sat)	40	pts
*88:	wcs1	thru Apr 13 (Sat)	15	pts
*89:	zoo1	thru Apr 13 (Sat)	40	pts
*90:	zoo2	thru Apr 13 (Sat)	50	pts
91:	a41	thru Apr 15 (Mon)	3	pts
92:	a42	thru Apr 17 (Wed)	3	pts
93:	comp	thru Apr 17 (Wed)	60	pts
94:	final	thru Apr 22 (Mon)	100	pts