CS 201 – Computer Programming II
Course Syllabus and Calendar – Spring 2003

Professor Don Colton

Brigham Young University Hawaii

1 Course Overview

This class teaches you to write programs in Perl for the Web using SQL databases at an intermediate level of programming proficiency.

Perl is a popular scripting language. It is very powerful and broadly available. It is even free. Amazon.com, one of the most famous web sites in history, was originally written in perl. (For speed, many parts of amazon.com have been rewritten in faster languages.) Perl makes things easy for the programmer, at the expense of being slower than C, C++, and other compiled languages.

The **Web** means your favorite browser, such as Microsoft Internet Explorer, Netscape Navigator, Mozilla, Opera, or even Lynx. You will create web pages dynamically, using forms, checkboxes, radio buttons, fill in the blanks, and press "submit." You will learn to create those web pages, and respond to the data entered into them. We will use CGI, the common gateway interface, as our processing method.

SQL, structured query language, is the standard for database access. You will be introduced to the basic commands for storing data, retrieving data, and updating data. We will use MySQL, a powerful and inexpensive implementation of SQL, as our database tool.

It's all free. We will not be learning proprietary and expensive languages and interfaces. Instead we will learn things that you can take with you right now and use on your next job without needing the chief financial officer to write a big check to get you started. You can run these tools at home or any place in the world (local laws allowing).

But what if your next job requires you to use Microsoft SQL? Or Microsoft ASP? No problem. The tools and design methods you learn will prepare you to understand and work with specialized tools that may be in use elsewhere. Those of you that drive "stick" will realize that it is easy to drive a car with an automatic transmission. Those of you that learned on an "automatic" will find out that it is difficult to drive "stick." We are teaching you to drive "stick."

Programming proficiency means that you and get something done without a lot of help. If you always need help for everything, you are a drag on your organization. You need to be a contributor. In this class you need to be able to write decent code on the final exam.

By the time you finish this course, you will be programming web pages. Not just HTML, but Forms and server-side CGI, relational data base entry, maybe even games. You will have some marketable skills. If you are a Math major or an Information Systems major, completion of this course will constitute "programming proficiency" for your major.

In the case of **Information Systems** majors, IS is not all about programming, but those without programming skills have limited mobility in the after-graduation job market.

In the case of **Mathematics Education** majors, you may be the most computer-savvy person at your school, and may get the opportunity to introduce students to computer programming.

In the case of **Mathematics** majors, some real programming ability is assumed in today's job market.

In the case of **Computer Science** majors, or other students who want more than an intermedate level of programming proficiency, there is yet more to learn. This course is a stepping stone to Computer Programming III (CS 202), where you will apply your 201 skills to Object-Oriented programming in a language like Java or C++.

Prerequisites: The formal prerequisite is CS 101 (Computer Programming I). At a minimum, I assume that you have written some programs. You probably know how to do formatted printing. You can use **if**, **else**, **while**, **do while**, **for**, and subroutines. I assume that you have some skill, but you may not be ready to sit in an interview and claim that you are a programmer.

Alternately, you can take the class if you have already completed some Calculus. This is because your level of mathematical maturity will allow you to learn the CS 101 material during our review (the first three weeks of class).

If you do not have prior Calculus or programming experience, you should confer with the instructor to make sure this is the right class for you.

1.1 The Course

- **Course Number:** CS 201
- **Title:** Computer Programming II
- **Course Description:** Review of CS 101. Problem solving, stacks, queues, hash tables, mathematical analysis of algorithms, regular expressions. Web programming including CGI and database. (Prerequisites: CS 101 and Math 110; or Math 112; or Math 119.)
- **Textbook (highly recommended):** *Perl by Example*, by: Ellie Quigley. ISBN 0-13-028251. SRP $44.99.
- Class Time: MWF 1:00–2:50 PM
  Final Exam: Wed 18 Jun, 1:00–2:50 PM
  Classroom: GCB 140

1.2 The Instructor

- **Instructor (me):** Don Colton
- **My email:** don@colton.byuh.edu
- **My Office:** GCB 130 B
- **Office Hours:** MWF 11 AM to 1 PM
- **Office Hours:** TTh 1 PM to 3 PM

1.3 The Tutors

- **T.A. Hours:** Mon–Thu, 3 PM to 11:30 PM **
- ** May stay open til 1 AM based on demand.
- **T.A. Hours:** Fri, 3 PM to 8 PM
- **T.A. Hours:** Sat, 3 PM to 6 PM
- **T.A. Location:** GCB 101 (CS Lab)

1.4 Office Hours

My office hours are shown above. You can contact me by email to make an appointment at another time. I also have an **open-door policy:** If my door is open (even just a bit) feel free to knock and come in.

1.5 Grading Overview

Your final grade will be computed using two methods, and the highest grade will be yours. Your total-points grade is based on 1000 points of assigned work. Some extra credit points may be 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

Method 1: Normal This method is based to a larger extent on your activity in the class. Major points are earned by attendance.

att	attendance	160 pts
pgm	programming labs	450 pts
qtc	testing center quizes	150 pts
qic	in-class quizes	120 pts
final	final exam (in class)	120 pts
tot	total points assigned	1000 pts

Method 2: Genius This method is suited to those geniuses that do really well on the labs and the final, and find class boring.

pgm	programming labs	500 pts
qtc	testing center quizes	100 pts
qic	in-class quizes	150 pts
final	final exam (in class)	250 pts
tot	total points assigned	1000 pts

In All Cases: You must achieve a sufficient score on the final exam, as shown in this table. Your final grade will be the **lower** of your total-points grade (above) and the grade in this table based on your final-exam percentage. Notice that these percentages are substantially lower than the point scales shown above.

83+	A	80–82	A-	77–79	B+
73–76	B	70–72	B-	67–69	C+
60–66	C	50–59	C-	40–49	D+
30–39	D	20–29	D-	0–19	F

Grading is discussed further below.

1.6 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 Course Calendar

First Quarter: During the first ten hours of class, we will review everything you should have learned in your previous programming classes. We will show you how to do these same things using Perl. There is a program due every day. We end with the first midterm exam.

Second Quarter: During the second ten hours of class, we focus on CGI and the web. There is a program due about every two days, but the programs are a bit harder. We learn some regular expression processing. We end with the second midterm exam.

Third Quarter: During the third ten hours of class, we focus on DBI, the database interface, using SQL. We store, retrieve, and update information in databases. We create new tables. We end with the third midterm exam.

Fourth Quarter: During the last ten hours of class, we deal with programming complexity and simple software engineering. We study "big oh" analysis. We end with the final exam.

Generally the lectures and discussion in class will follow the due dates for the various assignments (shown below). In-class tests will take one hour each (except the final).

```
Apr 30: hello      hello world
May 01: celsius    convert Fahrenheit
May 02: tc1        precedence simple
May 03: starbox    draw a box of stars
May 05: phonecard  phone card comparison
May 06: tc2        precedence mixed
May 07: lessthan   how many were less
May 08: perfect    perfect, excess, defic
May 09: argv       command line access
May 09: 2dice      win rates for 2 dice
May 09: 3dice      win rates for 3 dice
May 09: qic1       quiz in class 1

May 14: cgi0       cgi nuts and bolts
May 16: pick       pick a number
May 19: unlucky    Avoid unlucky numbers
May 19: tc4        regular expressions
May 21: cgi1a      cgi input parsing (simple)
May 21: cgi1b      cgi input parsing (complex)
May 21: qic2       quiz in class 2

May 28: dbcon      DBI connect, select
May 30: dbselect   DBI show tables, select
Jun 02: tc3        big oh analysis (simple)
Jun 04: dbupdate   DBI inventory
Jun 04: qic3       quiz in class 3

Jun 06: tc5        big oh analysis
Jun 09: checkwr1   check w blah blah
Jun 11: checkwr2   check w small nums
Jun 13: checkwr3   check w big nums
Jun 17: tc6        comprehensive exam
Jun 18: final      final in class
```

3 Now, About the Course

I assume that you want the ability to use programming in your future career. At the successful conclusion of this course, you will have programming proficiency in Perl. In today's job market (2002) and economy, this proficiency is not enough. There are many people qualified and experienced and out of work, and this raises the bar for finding a job. But if you want to do your own programming and not be at the mercy of others, this course will prepare you.

We will develop your programming skills by completing projects in areas that support electronic commerce on the web. We will develop your knowledge of some 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 Linux. Linux and Windows are the two operating environments that I believe will dominate the computing world in the next decade and beyond. Therefore, this class utilizes Linux 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, Linux, and EMACS among your skills on you résumé.

4 Course Content

The CS 201 course covers the following CC2001 Knowledge Units. These are defined in Computing Curricula 2001, a joint project of IEEE-CS and ACM. The IEEE Computer Society and the Association for Computing Machinery are the two major professional societies in computer science.

Much of this material is covered in more than one course.

PF1. Fundamental programming constructs

We review material that was introduced in CS 101.
- Basic syntax, semantics of a higher-level language
- Variables, types, expressions, and assignment
- Simple I/O
- Conditional and iterative control structures
- Functions and parameter passing
- Structured decomposition

PF2. Algorithms and problem-solving

- Problem-solving strategies
- The role of algorithms in problem-solving process
- Implementation strategies for algorithms
- Debugging strategies
- The concept and properties of algorithms

PF4. Recursion

We introduce material covered more fully in CS 202.
- The concept of recursion
- Recursive mathematical functions
- Simple recursive procedures
- Divide-and-conquer strategies

AL1. Basic algorithmic analysis

We introduce material covered more fully in CS 301.
- Asymptotic analysis of upper complexity bounds
- Differences among best, average, and worst case
- Big O, little o, Ω (omega), and Θ (theta) notation
- Standard complexity classes (linear, log, exponential)

NC4. The web as an example of client-server computing

- Web technologies
  - Server-side programs
  - Common gateway interface (CGI) programs
  - Client-side scripts
- Characteristics of web servers
  - Handling permissions
  - File management
  - Capabilities of common server architectures
- Role of client computers
- Nature of the client-server relationship
- Web protocols
- Tools for web site creation and management
- Developing Internet information servers
- Publishing information and applications

IM5. Database query languages

We introduct material covered more fully in IS 351.
- Overview of database languages
- SQL (query formulation, update sublanguage)
- Embedding queries in a procedural language

5 Grading

Your grade is earned by getting points for things like completing labs and tests. Progress reports are available to you by computer at any time.

Attendance: I take roll in this class. Attendance counts for 16% of your final grade. Typically attendance is worth 3 or 4 points per day. I take 4-point roll at the start of class. I take 3-point roll about 10 minutes into class. If you come later than that make 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.

In class 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 reading. Our class time will be focused on things you need to do the nearby assignments, or on explaining things that may not be sufficiently clear from the reading.

Reading: I am using *Elements of Programming with Perl* by Johnson. Student response to this book has been quite good, and it is in its third edition, suggesting that it has sold fairly well. I continue to be 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.

Labs: 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.

Much like CS 101, 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.

Some of the programs will be graded by demonstration in class. In such cases you will have more freedom in the user interface. Programs graded by GradeBot allow freedom in the methods used to construct the answer, but they do not allow freedom in how the answer is presented: the user interface is specified for you.

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 web, "turnin," or email.

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.

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.

Tests: There are several 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, plus the final. 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.

Deadlines: Each assignment has a deadline. You can see these deadlines by asking GradeBot. 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.

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.

6 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.

7 Special Needs

Brigham Young University Hawaii 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.

8 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).