Course Syllabus for CIS 130 – Introduction to Programming – Spring 2010

Basic Course Information:

Instructor:	Sharon Tuttle		
Lecture time and location:	Wednesday	1:00 – 2:50 pm	BSS 313
Lab time and location:	Monday	1:00 – 2:50 pm	BSS 313
Instructor's office:	BSS 322		
Instructor's e-mail:	st10@humboldt.edu	(or sharon.tuttle@humboldt.edu)	
Instructor's office phone:	(707) 826-3381		
Instructor's office hours:	Tuesday, Thursday	1:00 – 3:00 pm	
	Friday	1:00 – 2:00 pm	
	or by appointment		
Course public web page:	follow link from:	http://www.humboldt.edu/~st10	
	Or reach via link from course Moodle site		

Course Description:

[from the HSU catalog] Problem decomposition, algorithm design, modularity, cohesion, coupling, control structures, simple data structures, testing, and error detection approaches and documentation.

This course is an introduction to programming that happens to use the Scheme and C++ programming languages. There will be a strong emphasis on problem-solving and on good program design.

Students are expected to come into this course already-comfortable using basic computer applications. No prior programming knowledge is assumed or required.

Course Objectives:

After successfully completing this course, students should be able to:*

- Design, implement, test, debug, and use functions, following a standard design recipe.
- Analyze and explain the behavior of simple programs involving the fundamental programming

* some of these are adapted from the ACM Computer Science Curriculum 2001, http://www.sigcse.org/resources/cs-2001

structures of sequence, condition, iteration, and procedure.

- Modify and expand short programs that involve standard conditional and iterative control structures and auxiliary functions.
- Design, implement, test, and debug functions that use each of the following fundamental programming constructs: basic computation, simple I/O, and standard conditional and iterative structures.
- Choose appropriate conditional and iteration constructs for a given programming task.
- Apply the techniques of structured (functional) decomposition to break a program into smaller pieces (or, better yet, to design it from smaller components to begin with!)
- Describe the mechanics of parameter passing.
- Discuss the importance of algorithms in the problem-solving process.
- Identify the necessary properties of good algorithms.
- Create algorithms for solving simple problems.
- Use pseudocode or a programming language to design, implement, test, and debug algorithms for solving simple problems.
- Describe strategies that are useful in debugging.
- Describe common applications for primitive data types, arrays, and strings.

Course Prerequisites:

Math code 40 and general familiarity with basic computer applications.

Required Course Text, Materials, etc.:

- Turning Point RF Response Clicker, available at the campus bookstore, or a Responseware license, available on-line (address to be provided).
- Links to on-line required readings will be assigned and provided.

Recommended Course Text:

"Problem Solving with C++: the Object of Programming", Seventh edition, by Savitch, published by Addison-Wesley.

(Note, however, that this book is being recommended as a C++ reference for students who would like such a reference – no required reading assignments will be made from this text. Also note that you will not need the CD that comes with this text.)

Course Software:

For the first part of the semester, you are expected to use subsets of the Scheme programming language using the **DrScheme** programming environment. This software is available from http://www.drscheme.org, and has versions for Mac OS X, Linux, and Windows.

For the remainder of the semester, you are expected to use the GNU C++ compiler installed on nrs-labs.humboldt.edu. This, too, is free software, although we initially will be using it in conjunction with

some customized C++ tools on nrs-labs. But, this GNU C++ compiler is, for example, included in Dev-C++, a free C++ environment for Windows available in the BSS 313, BSS 317, and FOR 204A labs and available for download (for Windows) from http://www.bloodshed.net/. (Mac OS X users can download Xcode from the Apple Developers Connection, which includes this same GNU C++ compiler, if their computer doesn't already have the GNU C++ compiler. Mac OS X, Linux, and Windows users can also download the C++ version of Eclipse from http://www.eclipse.org .)

So, if you were to develop preliminary versions of your C++ coding in Dev-C++ or on a Mac or under Linux, such code ought to run on nrs-labs as well -- but it is your responsibility to verify that this is the case for your code before submitting it.

Throughout the semester, you will be making some use of the UNIX operating system (since that's what nrs-labs.humboldt.edu uses). Instruction on how to do so will be provided. Note that you may access nrs-labs.humboldt.edu by using ssh (secure shell) and sftp (secure ftp); ssh may be downloaded for free from:

https://experts.humboldt.edu/ditss/download/

...although in Fall 2008 some students had better luck downloading it from:

http://www.colorado.edu/its/docs/authenticate/printouts/win_ssh.html

...which also happens to include a nice illustrated tutorial as well.

Note, also, that ssh and sftp are installed on redwood and sorrel, so if you can reach one of them, you can generally also reach nrs-labs from them.

Clickers:

I am trying something new in this course this semester. We will be using Turning Technologies student response clickers (or Responseware) in lecture and in lab. There is significant literature indicating that using clickers may increase student engagement and success in learning course material. Students purchase these clickers (or buy a Responseware license); clickers can be returned at the end of the semester for a refund of part of the purchase price. Students register their clickers by entering the large number (consisting of 6 characters/digits) on back of the clicker at a special address that I will provide, and then bring them to every class session. (Students using Responseware sign in during each class session, using a special code that I will project at the beginning of that class session.)

These clickers will be used for in-class questions, which will be interspersed during the lecture and lab sessions (although probably with more questions during lecture sessions, since lab sessions may also include lab exercises). The system will record the overall class response percentages as well as keep track of the individual answers. Students will receive 2 points for a correct answer, 1 point for an incorrect answer, and no points for no answer, but with a maximum semester clicker-questions grade of 120. Thus you will be rewarded for regular attendance. If you miss a class session, you miss the in-lecture or in-lab questions and cannot make them up. However, there will be at least 65 questions asked over the course of the semester so there is opportunity for extra credit (up to a maximum clicker grade of 120) (or to make up for a day that you are out due to illness, although note that you are still responsible for finding out what you missed on such days).

(If you forget your clicker for a class session, then **up to 5 times** you may e-mail me your clicker answers for that day, using the Subject: **CIS 130 Clicker Answers for <date>**, with a 2-point penalty.)

The idea is that the clicker questions will provide you with some indications of how I think and some of the things that I think are important. Thus, they should help prepare you for the course exams (although exams will include questions that go more in depth than clicker questions can...!) Additionally, the questions will enable you to get some immediate feedback on how you are doing, whether you are understanding things, whether you need to pay more attention to readings, etc. It may even help me know what I need to explain better.

I will run a test of the system during the first two weeks of class and will begin the questions that "count" during the 3rd week of classes. Therefore you must purchase the clicker and register it as soon as possible. If there is an issue with this, see me immediately.

Please keep in mind that this in an experiment -- I will be working out how to effectively use clickers during the semester for this course.

Make sure that you bring your clicker to every class session (lecture and lab).

Grading Breakdown:

If you are a Computer Information Systems (CIS) major, it is **important** that you note that **you must earn at least a C in CIS 130** for this course to count towards your major.

Your semester grade will be determined by the percentage of points that you earn, **subject to some minimum requirements**. Here is the mapping of percentage to grade, and those minimum requirements:

Homeworks assignments:		25.0%	
Lab exercises:		12.5%	
Clicker questions:		12.5%	
Exams:	**Exam #1:**	15.0%	
	Exam #2:	15.0%	
	Final Exam:	20.0%	MONDAY, May 10th, 12:40 – 2:30 pm, in BSS 313

Grading Scale:

1. To earn a grade of **C or better** in this course, the following three requirements must **all** be met:
   - your overall semester average must **equal or exceed 72.5%** - this is to show a reasonable level of overall mastery of the course material.
   - the **average** of your Exam #1, Exam #2, and Final Exam grades must **equal or exceed 60%** - this is to show that you understand at least a minimal reasonable level of the most important course concepts.
   - the **average** of your Homework assignments must **equal or exceed 60%** - because this is a programming course, but programming acumen is not tested as effectively on exams, this is to show at least a minimal level of programming competence and experience in addition to course concept mastery. Also, past experience has shown that students who do not put a solid effort into the course programming assignments do not do well on the course exams.
2. If **all three** requirements above are **not** met, then your semester grade will be **either C-** or the letter grade computed according to the mapping given below, **whichever is lower**.

   (That is, if a student had an overall semester average of 74% but a Homeworks average of 55%, that student would receive a **C-** for his/her semester grade; if a student had a Homeworks average of 61% and an Exams average of 71%, but an overall semester average of 65%. then that student would receive a **D** for his/her semester grade. You are expected to ASK ME if this aspect of the grading policy is not clear to you.)
3. If all three requirements above are met, then your semester grade will be computed according to the mapping given below:

Overall Percentage (based on the given weights)	Exams Average	Homeworks Average	Letter Grade
>= 93	>= 60	>= 60	A
>= 90 and < 93	>= 60	>= 60	A-
>= 87 and < 90	>= 60	>= 60	B+
>= 83 and < 87	>= 60	>= 60	B
>= 80 and < 83	>= 60	>= 60	B-
>= 77 and < 80	>= 60	>= 60	C+
>= 73 and < 77	>= 60	>= 60	C
>= 73	< 60	any	C-
>= 73	any	< 60	C-
>= 70 and < 73	any	any	C-
>= 67 and < 70	any	any	D+
>= 60 and < 67	any	any	D
< 60	any	any	F

Final Exam:

Again, the Final Exam for this course is scheduled for MONDAY, May 10th, 12:40 – 2:30 pm, in BSS 313 (unless I announce otherwise). Note this time and date BEFORE making your end-of-semester travel plans.

Students with Disabilities:

Persons who wish to request disability-related accommodations should contact the **Student Disability Resource Center** in the basement of the Library, **826-4678 (voice)** or **826-5392 (TDD)**. Some accommodations may take up to several weeks to arrange. You can reach the Student Disability Resource Center's web site at http://www.humboldt.edu/~sdrc/. Please note that some accommodations may take up to several weeks to arrange; also, if you are eligible for such accommodations, please contact me as soon as possible to discuss them.

Add/Drop Policy:

Students are responsible for knowing the University policy, procedures, and schedule for dropping or adding classes. You can find these on the web at

http://www.humboldt.edu/~reg/regulations/schedadjust.html

Note that the CSU (and thus HSU) policies on withdrawing from and retaking courses changed as of Fall 2009:

4. Students may withdraw from no more than 18 semester-units (between census and the final 20% of instruction, with a serious and compelling reason).
5. Students may repeat courses only if they earned grades lower than a C.

6. Students may repeat up to 16 semester-units with grade forgiveness.
7. Students may repeat up to an additional 12 semester-units with grades averaged.

Be careful – as of Fall 2009, HSU is being much more strict about what constitutes a "serious and compelling reason".

The census date for Spring 2010 (before which you can drop without a W, and without it counting toward your 18 semester-units drop limit) is: **Monday, February 15th**.

The last date for Spring 2010 to drop with a W on your transcript, with a serious and compelling reason, and subject to the 18 semester-unit drop limit, is: **Friday, April 16th**.

If you do drop the course, note that it is **your responsibility** to complete and submit the appropriate paperwork.

Incompletes:

Incompletes are rarely given and only in the case of a true emergency. They certainly are not appropriate for students who find they have fallen behind on assignments, missed a test, or taken on too much academic, work, or family responsibilities. For these situations, dropping the course would be appropriate (subject to the University policy for dropping courses).

Time Expectations:

Remember the general rule of thumb for college-level courses --- to be successful in a course, you should plan to spend at least 3 hours outside of class for each 1 hour of college course credit. That implies an estimate of at least 9 hours a week spent outside of class for this 3-credit course.

However, you should be warned that:

- Programming courses can be notorious time eaters – occasionally, a problem with your program will take large amounts of time to locate and fix. Starting early enough so that you have time to ask me questions when you run into problems can help with this.
- You can only learn programming by practicing it, and it takes some much longer than others to master it. Practicing programming as much as possible helps.
- The course will intensify as the semester progresses – as you are able to do more, you will be expected to do more.
- Homework deadlines will not be extended because you waited too late to start or because you did not allocate enough time before the deadline to work on it; you will simply have to submit whatever you have managed to do by the deadline.

Academic Honesty:

Students are responsible for knowing policy regarding academic honesty:

http://studentaffairs.humboldt.edu/judicial/academic_honesty.php

Observe that among the actions that are unacceptable are submitting another's program, code, or file as your own and failing to quote material taken from another person's written work.

All course work is to be the work of each student, **individually**, **unless** it is **explicitly** stated otherwise at the beginning of that course work's description. Except for explicit exceptions, this is **not** a group or team programming course. When group work is explicitly permitted, the names of all students involved must be included on the work submitted. (For example, when you use **pair programming** in lab, the

lab exercise will specify that, and the files you turn in will include both of the names of the students who worked as a pair.)

(When an assignment does specify that it is acceptable to work in pairs or groups, make sure that you don't get into the situation where you are merely watching someone else learn.)

For individual work (that is not specified as pair-programming), students may discuss general approaches **as long as no one involved in the discussion is writing anything down or typing anything during such discussions**. Students may also help one another in determining causes of program bugs, or in determining the meaning of compiler error messages. However, students may not work together to complete homeworks, one student should not instruct another in how to write the code for a homework, and **any type of copying or modifying of another person's computer files, OR of providing computer files to another, related to homeworks is definitely over the line, and never justified**. This applies to copying of documentation and comments as well as to copying of program code.

Note that it is **your** responsibility to ensure that course assignment files are read-protected. If you are careless about this, and someone else copies your work, you will share the penalty. (In particular, be very careful about leaving work on shared network drives in campus labs, or in UNIX/LINUX directories that are not read-protected.)

Learning takes hard work; when students turn in others' work as their own, it is a slap in the face to those seriously interested in learning. Not turning in an assignment results in no credit for that assignment, of course, but that is an honest grade. Work that violates the course honesty policy deserves a lower grade than that, and therefore the course policy is that work violating this policy will receive **negative** credit. A person providing a file for copying receives the same **negative** credit as the copier. Repeat offenses will be handled according to University policies.

Additional Course Policies:

- You are expected to read this syllabus and be prepared to sign a statement that says you have received and understand these policies.
- You are encouraged to ask me questions in class, in office hours, and by e-mail. The most successful students are those who are not afraid to ask questions early and often (I will gently let you know if you are overdoing it), who do the assigned reading, who attend lecture and lab regularly, who start assignments early, and who practice course concepts as much as possible.
  - That said, I am expecting that you will ask specific questions – overly vague or broad questions are problematic. (For example, an example of a specific question is, “When I try to compile function X, I receive the following error message: … and my code is attached to the end of this e-mail. Can you point me in the right direction about what is wrong?” An example of an overly vague or broad question is: “Here's my code. Is it right?”
- I generally check my e-mail (st10@humboldt.edu or sharon.tuttle@humboldt.edu) at least once a day on weekdays. Include **CIS 130** in the Subject: line for expedited handling.
- You should monitor your e-mail for course-related messages. The University provides a means for you to specify your preferred e-mail address, so if you wish to receive e-mail into an account other than the one HSU provides, change your preferred e-mail address in both Banner (the new Student Center?) and Moodle accordingly. Course-related messages from me will include **CIS 130** in the Subject: line.
- If you would like me to e-mail certain course grades to you during the semester, then you must give me permission in writing on the course information form.
- Regular attendance at lecture and lab sessions is expected. If you should happen to miss a lecture or a

lab, then you are responsible for finding out what you missed. "I wasn't there that time" is never an acceptable excuse. Lecture notes are not posted, although many of the projected examples will be made available on the course web site. Clicker questions and graded lab exercises missed **cannot** be made up later.

- Clicker questions will be given during most lectures and labs (although probably more clicker questions will be asked during lecture sessions); graded lab exercises will be given during most lab sessions. The two lowest lab exercise grades will be dropped from your grade. Between the ample quantity of clicker questions and the dropped lab exercise grades, then, you can be absent several times from non-exam lecture or lab sessions without direct penalty, for whatever reason (although you are, of course, still responsible for the material covered on those days, and it is your responsibility for determining what that material is).
- Note: **NO** homework grades are dropped; **ALL** homework grades count toward your homework average. Every homework includes important practice of programming fundamentals.
- When reading assignments are given, you are expected to prepare (read and study) assigned readings before class and to participate in class discussions. Projected examples will be utilized frequently during discussion. You should understand that there may be material in the reading that will not be discussed in lecture/lab, and material in the lectures/labs that may not be found in the reading. You are responsible for both.
- As previously mentioned, during lab sessions, there may be lab exercises due during that lab session. Once a lab's lab exercise is complete, the remaining lab time should be used to continue work on the current course homework assignment, to practice course concepts, and/or to ask questions about course-related topics.
- You should not expect to be able to finish homework assignments during the lab sessions--- like any college-level course, you should expect to put in a (potentially) large amount of time outside of scheduled class meetings (lectures and labs) doing the assigned reading, working on homework assignments, and practicing concepts discussed.
- Each homework must be submitted as is specified on the homework handout in order to be accepted for credit. This may vary for different homeworks. Often, parts of homeworks will be submitted using a special tool on nrs-labs. Code that does not run in DrScheme or on nrs-labs will not receive credit; remember that it is your responsibility to verify that your code runs in DrScheme or on nrs-labs for each homework before submitting its code, regardless of where you developed that code.
- Each homework will be clearly marked with one or more due dates (a single homework could have multiple parts with multiple due dates).
  - **No homework will be accepted late. If you wish to receive any credit for a homework, then you must turn in whatever you have done, even if it is incomplete, by the deadline. Partial credit is usually preferable to no credit.** Note that "the computer/network/etc. going down" is no excuse --- if you leave a homework for the last minute and there are technical problems, you still must turn in whatever you have by the deadline.
  - You may submit multiple versions of homework files before the deadline; I will grade the latest pre-deadline submission unless you inform me otherwise. This is to encourage you to turn homework parts in early (since you will know that you can always turn in an improved version if further inspiration strikes).
- The tool that you will be using to submit homework parts results in a file that serves as your "receipt" for having submitted items. You are expected to retain these "receipt" files at least until a grade has been posted to the course Moodle site for that homework. If there is a system glitch or other hardware/software/network problem, you may be asked to make me a copy of one or more receipt files; if you do not have them, then you will not receive credit for the files involved. These receipt

files are for your protection!

- The successful student in this course will start homeworks **promptly** after they are made available from the course web page. You are **encouraged** to send me e-mail with specific questions you have as you work on these homeworks; if you wait until class meetings or even office hours to ask such questions, you may not have time to complete the homework.
  - It is not a problem if you send me a question and then end up answering it yourself before you receive my answer; likewise, it is not a problem if you end up sending me several questions (as you work on different parts of a homework while awaiting earlier answers).
- It's nearly impossible to write unambiguous program specifications. If you have questions about "what she means", get them resolved very early in the development cycle by **asking**.
  - Part of your grade will be determined by how well your programs meets the written requirements. Programs are expected to meet homework handout specifications precisely; when one eventually works within a team on large software projects, following the specifications precisely is vital, and can mean the difference between a working product and one that just sits there.
  - Note that programs may be graded on **style** as well as on whether they run properly and whether they precisely meet the homework specifications and requirements. Discussions on programming style will be ongoing throughout the semester.
  - Some course work may be graded based on whether it has been **attempted** (the instructor's decision is final as to whether this is the case) --- other course work may be graded for correctness, style, and whether it meets specifications. You will not know in advance which will be the case.
- Exam dates are given in the course schedule below. No make-up exams will be given, except by special prior arrangement. Note that the Testing Center's services were severely curtailed in Fall 2003, due to budget cuts.
- You are expected to check the public course web page and the course Moodle site regularly --- course handouts, homework assignments, example code from lectures, and possibly more will be posted to the public course web page, and grades will be posted to the course Moodle site. You are expected to monitor your posted grades and let me know about any discrepancies.
- **Attendance and disruptive behavior:** Students are responsible for knowing policy regarding attendance and disruptive behavior: http://studentaffairs.humboldt.edu/judicial/attendance_behavior.php
- **Late arrival to class:** Please attempt to come to class on time, with your headphones put away and your cell phones turned off. If you must arrive late or leave early, please do so with the least possible distraction to other students. If your late/early habits become disruptive, you may be asked to leave the class permanently.
- **Class disruption:** University policy requires that instructors eliminate disruptions to the educational process. Distractions such as excess talking, ringing cell phones, working on assignments for other classes, demonstrations of affection, packing of books early, loud music leaking from headphones, chronic late arrivals or early departures, excessive comings and goings or other behaviors that disrupt the class are not acceptable. Students indulging in such behaviors will first be warned before being required to leave the class permanently.
- **Emergency Evacuation:** Please review the evacuation plan for the classroom (posted on the orange signs), and review the campus Emergency Preparedness web site at: http://studentaffairs.humboldt.edu/emergencyops/campus_emergency_preparedness.php for information on campus Emergency Procedures. During an emergency, information regarding campus conditions can be found at **826-INFO** or http://www.humboldt.edu/emergency .

Tentative Course Schedule: (subject to change!)

Week 1: January 20

- Topics: Introduction to course (no class Monday, January 18: Martin Luther King, Jr., holiday)

Week 2: January 25

- Topics: simple and compound expressions; data types (number, boolean, string, image); syntax and semantics
- Wednesday, January 27 – NO CLASS, Instructor furlough day
- HW #1 out

Week 3: February 1, 3

- Topics: introduction to functions and to the program design recipe; parameter variables
- HW #1 due. HW #2 out

Week 4: February 8, 10

- Topics: boolean functions, relational expressions, simple conditional statements, conditional functions
- HW #2 due, HW #3 out

Week 5: February 15, 17

- Topics: composing auxiliary functions and definitions into a program
- Monday, February 15 – NO CLASS, Instructor furlough day
- HW #3 due

Week 6: February 22, 24

- Monday, February 22: review for Exam #1
- Wednesday, February 24: **EXAM #1**

Week 7: March 1, 3

- Topics: Introduction to C++
- HW #4 out

Week 8: March 8, 10

- Topics: conditional statements in C++
- HW #4 due, HW #5 out

HSU Spring Break: March 15-19

Week 9: March 22, 24

- Topics: local variables; assignment; cout (screen output), cin (interactive input); void functions; definition of a C++ program; main function; compiling/linking functions to create a C++ program
- HW #5 due, HW #6 out

Week 10: March 29, 31

- Monday, March 29 – Review for Exam #2
- Wednesday, March 31 – NO CLASS, Cesar Chavez Holiday
- HW #6 due

Week 11: April 5, 7

- Monday, April 5 – NO CLASS, Instructor furlough day
- Wednesday, April 7 – **EXAM #2**

Week 12: April 12, 14

- Topics: mutation; C++'s += and ++ operators; scope; pass-by-value; intro to imperative repetition in C++ using while loops; intro to count-controlled loops
- HW #7 out

Week 13: April 19, 21

- Topics: introduction to arrays; intro to for-loops; examples of functions involving arrays
- HW #7 due; HW #8 out

Week 14: April 26, 28

- Topics: intro to sentinel-controlled loops; intro to file input/output; pass-by-reference
- HW #8 due; HW #9 out

Week 15: May 3, 5

- Topics: to be announced
- Wednesday, May 5: Review for Final Exam
- HW #9 due

Final Exam:

MONDAY, May 10th, 12:40 – 2:30 pm, in BSS 313 (unless I announce otherwise)