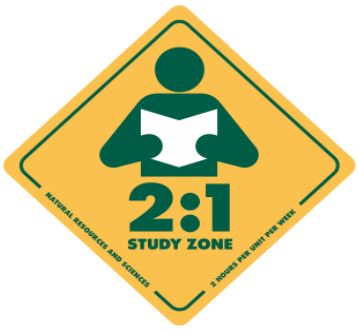

Course Syllabus for CS 111
Computer Science Foundations 1
Fall 2016

Basic Course Information:

Instructor:	Sharon Tuttle	
Lecture times and location:	Tuesday, Thursday	1:00 pm - 2:20 pm SH 108
Lab times and locations:	Section 11: Friday Section 12: Friday	11:00 am - 12:50 pm BSS 313 1:00 pm - 2:50 pm BSS 313
Instructor's office:	BSS 322	
Instructor's e-mail:	st10@humboldt.edu or sharon.tuttle@humboldt.edu or smtuttle@humboldt.edu	
Instructor's office phone:	(707) 826-3381	
Instructor's office hours:	Tuesday, Thursday	3:30 pm - 5:00 pm
	Wednesday	5:00 pm - 6:00 pm
	Friday	4:00 pm - 5:00 pm
	or by appointment	
Course public web page:	follow link from: http://users.humboldt.edu/smtuttle/ or follow link from course Moodle site	

Course Description:

[from the HSU catalog:] Introductory programming covering problem decomposition, control structures, simple data structures, testing, and documentation. Students design and implement a number of programs.

In this course, you will begin to explore the art and science of problem solving using the computer as a tool. Course topics will include problem solving, simple expressions and compound expressions, types of data, syntax and semantics, function application, design, and definition, introduction to software testing, Boolean operations, Boolean functions, conditional expressions, input-process-output, and the basic structures of computing: sequence, conditional, iteration, and procedure.

The problem solving techniques learned in this course will lead to a study of object-oriented programming in CS

112 - Computer Science Foundations 2.

Students are expected to come into this course already comfortable with basic computer use. **No prior programming knowledge is assumed or required.** In this particular offering of this course, we will start the semester programming in the Racket language, and then transition to programming in C++.

Course Co-requisite:

Math 113 - College Algebra OR Math 115 - Algebra & Elementary Functions, or instructor approval.

Notice that this is a **CO-REQUISITE** for CS 111. That means Math 113 or Math 115 can be taken either before OR *at the same time* as CS 111.

Course Objectives:

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

- Design, implement, test, and debug programs that use each of the following fundamental programming constructs: basic computation, standard conditional and iterative structures, and the definition of functions.
- Analyze the behavior of simple programs involving fundamental programming constructs.
- Choose appropriate conditional and iterative constructs for a programming task.
- Apply the techniques of structured (functional) decomposition to break a program into smaller pieces -- or, better yet, originally design it using such smaller pieces.
- Describe strategies that are useful in testing and debugging.
- Write clear comments that communicate to the reader what a function expects and what it produces.

CS Program Learning Outcomes that this course addresses:

This course addresses departmental learning outcomes of:

- Computational Thinking
- Technical Writing
- Communicating and Collaborating

This course addresses computational thinking at an introductory level, introducing fundamental computing concepts. It addresses technical writing at an introductory level via program documentation and coding standards that stress reusable code, and it addresses communicating and collaborating at an introductory level via experience pair-programming in course lab sessions.

HSU Learning Outcomes that this course addresses:

This course explicitly contributes to students' acquisition of skills and knowledge relevant to HSU Learning Outcomes:

HSU graduates will have demonstrated:

- Effective communication through written and oral modes.
- Critical and creative thinking skills in acquiring a broad base of knowledge and applying it to complex issues.
- Competence in a major area of study.
- Appreciation for and understanding of an expanded world perspective by engaging respectfully with a diverse range of individuals, communities, and viewpoints.

---

* Some of these are adapted from the ACM Computer Science Curriculum 2001, available from link at: http://www.acm.org/education/curricula-recommendations

HSU graduates will be prepared to:

- Succeed in their chosen careers.

Required Course Materials:

- "How to Design Programs", Second Edition, Felleisen, Findler, Flatt, and Krishnamurthi, currently only available on-line, at:

  http://www.ccs.neu.edu/home/matthias/HtDP2e/

- Turning Account License used with TurningPoint RF Response Clicker or ResponseWare
- Any additional required readings will be made available either on-line, or via resources available through the HSU Library such as the ACM Digital Library and Safari TechBooks Online.

Recommended Course Text:

- "Problem Solving with C++", Savitch, Addison-Wesley
  - Note that this is being included as a recommended text just for C++ language reference only, for students who would like such a reference. No assignments or required readings will come from this text.
  - The current edition of this is the 9th edition, but for this course's purposes, the 7th and 8th editions are also fine. (It is possible that earlier might also be, but I am not as certain of this.)

Course Software:

For the first part of the semester, you are expected to use subsets of the Racket programming language using the DrRacket programming environment. This software is available from:

http://www.racket-lang.org/

...and has versions for Windows, Linux, and Mac OS X; it should also be available in BSS 313.

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. There are also different C++ environments that happen to use this compiler, as we will discuss later in the semester.

(So, if you were to develop preliminary versions of your C++ coding in one of these C++ environments, 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. Note that you may access `nrs-labs.humboldt.edu` by using the programs `ssh` (secure shell) and `sftp` (secure ftp); we will walk through how this can be done during an early class lab session. `ssh` and `sftp` are already available in on-campus labs, and you should be able to download one of several versions for free for your outside-lab use.

Campus labs have PuTTY installed to provide a GUI implementation of `ssh` and WinSCP installed to provide a GUI implementation of `sftp`. Also, command-line versions of `ssh` and `sftp` (usable from the Mac OS X Terminal command-line) are installed already as part of Mac OS X.

Clickers:

We will be using Turning Technologies student response clickers or ResponseWare in class. There is significant literature indicating that using clickers may increase student engagement and success in learning.

Students purchase this clicker and a license, or they use this license with the ResponseWare application on a mobile device. You then bring your clicker or mobile device to every class meeting (lectures and labs).

This class will be using Moodle this semester; I will be letting you know how to register so that your clicker answers receive credit.

These clickers will be used for in-class questions, which will be interspersed within class meetings. These will usually be given in a **think-pair-share** fashion, in which you answer a question first individually, and then discuss your answer with another student, discussing why you think your answer is correct; if they gave a different answer, you try to persuade them that yours is the correct answer, and then either of you can change your answer if you wish. The response system will record the overall class response percentages as well as keep track of individual answers.

Typically, you will receive:

- **1.5 points** for a correct answer,
- **0.75 points** for an incorrect answer, and
- **0 points** for no answer,
- but with a **maximum-possible** semester clicker-questions grade of **120**.
- (There may be some no-point questions from time-to-time as well -- such questions will be noted if/when they come up.)

Thus you will be rewarded for regular attendance and participation. If you miss a class session, you miss that day's clicker questions and cannot make them up (except for extraordinary circumstances). However, there will be a sufficient number of questions asked to allow for the possibility of extra credit (up to a **maximum-possible** 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 or mobile device for a class meeting, then **up to 5 times** you may still receive some clicker credit, **usually minus a 1.5-point penalty**, by e-mailing me your clicker answers for that day, **by 11:59 pm on that day**, using a Subject: line of: `Subject: CS 111 Clicker Answers for <date>`. Later e-mails, or e-mails without the proper `Subject:` line, might not be accepted for credit.

The idea is that the clicker questions will help you to see if you are starting to understand concepts being discussed; sometimes they will also provide review of concepts discussed previously. Clicker questions are typically quite different from exam questions (since clicker questions are typically multiple-choice questions, while exam questions will rarely be multiple-choice). They still enable you to get some immediate feedback regarding whether you are grasping course concepts, whether you need to pay more attention to course discussions and/or readings, etc. They may even help me to know what concepts might need more explanation in-class.

I hope to run tests of the system during the first week's class meetings, and hope to begin asking questions that "count" during the second week's class meetings. Therefore, you must register and purchase your clicker and/or license as soon as possible. If there is an issue with this, contact me immediately.

Finally, please note that use of another CS 111 student's clicker, or having someone else use your clicker in a CS 111 class session, or otherwise having anyone but yourself answering a clicker question on your behalf -- that is, pretending that someone is in class who actually is not -- is considered to be **cheating**, with the same policies applying as would be the case if you turned in someone else's work as your own or permitted someone else to copy your work. Please **ASK ME** if you are not sure what I mean by this.

Grading Breakdown:

If you are a Computer Science (CS) major, note that you must earn at least a **C-** in CS 111 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 are the grade percentages, followed by those minimum requirements:

Homework assignments:	25%
Lab exercises:	10%

Clicker questions:		10%	
Lab quizzes:		10%	
Exams:	**Exam 1:**	12.5%	
	Exam 2:	12.5%	
	Final Exam:	20.0%	Thursday, December 15, **12:40**-2:30 pm, SH 108

Grade Requirements:

- To earn a grade of **C- or better** in this course, the following three requirements must **all** be met:
  1. your overall semester average must **equal or exceed 70%** - this is to show a reasonable level of overall mastery of the course material.
  2. 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.
  3. the **average** of your Homework assignments must **equal or exceed 60%** - because there are hands-on skills that are part of this course that are not tested as effectively on exams, this is to show at least a minimal level of programming experience in addition to course concept mastery. Also, part experience has shown that, in general, students who do not put a solid effort into course homework assignments do not do well on course exams.
- If **all three** requirements above are **not** met, then your semester grade will be **either D+** 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 **D+** for their 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 their semester grade. You are expected to ASK ME if this aspect of the grading policy is not clear to you.)
- Including the three requirements noted above, your semester grade will be computed according to the mapping given below:

Overall Percentage (based on the given weights)	**Exams Average**	**Homework 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
>= 70 and < 73	>= 60	>= 60	C-
>= 70	< 60	any	D+
>= 70	any	< 60	D+
>= 67 and < 70	any	any	D+
>= 60 and < 67	any	any	D

Overall Percentage (based on the given weights)	Exams Average	Homework Average	Letter Grade
< 60	any	any	F

Final Exam:

Again, the Final Exam for this course is scheduled for **Thursday, December 15, 12:40 - 2:30 pm**, in **SH 108** (unless I announce otherwise). Note this time and date **BEFORE** making your end-of-semester travel plans.

Additional Grading-Related Policies:

- Clicker questions will be given during most lectures and labs; graded lab exercises will be given during most lab sessions. I also plan to experiment this semester with short lab quizzes, to be given during the first ten minutes of many of the lab sessions.
  - The **two lowest lab exercise grades** will be dropped from your grade.
  - The **two lowest lab quiz grades** will be dropped from your grade.
  - Between the ample quantity of clicker questions, the dropped lab exercise grades, and the dropped lab quiz 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** to determine what that material is).
- Note: **NO** homework grades are dropped; **ALL** homework grades count toward your homework average. Every homework includes important practice of course fundamentals.

Course Expectations:

First: 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 12 hours a week spent outside of class for this 4-credit course**.

However, you should be aware that:

- You can only learn programming by practicing it. Practicing programming as much as possible helps!
  - This can include playing around with in-class examples, experimenting to see if something you are curious about really works like you think, and so on.
  - Think of a musical instrument -- you have to practice to master playing a guitar, violin, trumpet, drums, etc. You can't master it by just reading about the instrument. Think, also, of sports skills such as pitching, putting, etc. -- again, repetition and practice is required to hone such skills.
- Writing programs can be a notorious time eater. Occasionally, a problem with code can potentially take large amounts of time to locate and fix (especially if you don't ask for help!).
  - Starting **early** enough so that you have time to ask me questions when you run into problems can help with this!
  - Why spend 4 hours struggling with a frustrating roadblock the night before the assignment is due, when you can spend 10 minutes composing an e-mail early in the week, work on other problems while waiting for the answer, and then get a reply that makes everything clearer as soon as you read it?
- I cannot emphasize this enough: you need to start homework assignments as soon as they are made available, to submit homework parts throughout the week, and to ask questions as you have them!
  - This is not a course that you can work on just one day a week, nor is it a course that you can take a week off from and expect to do well.
- The course will intensify as the semester progresses – as you are able to do more, you will be expected to do

more. Also, later concepts are built upon earlier concepts as the course progresses. If you ask me as soon as you realize that some concept is not clear to you, that can help keep you from falling behind.

- 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; likewise, they will typically **not** be extended because of hardware or network failure. (Admittedly, campus failures might affect deadlines. But don't assume so until you have heard from me definitively.) You need to keep backups of your files at all times, and need to plan your schedule to be able to work on on-campus computers as necessary.
- If you have not completed an assignment by the deadline, **your best choice is to submit whatever you have managed to do by then**, as partial credit is your friend, to carefully study the posted example solution as soon as it is available, to ask me about anything there that is still unclear, and to get a good **early** start on the next homework.

A successful student in this class will:

- Attend every lecture and lab, clicker at the ready.
- Participate in class (discussing clicker answers with other students, asking questions, paying attention, taking notes, being an attentive partner when pair-programming in lab).
- Complete reading assignments in a timely fashion.
- Practice and "play around" with posted examples.
- Ask specific questions -- in class, in lab, in office-hours, and in e-mail.
- Read through each homework assignment as soon as it is posted.
- Start working on each homework as soon as it is posted.
- E-mail the instructor with specific homework-related questions starting soon after it is posted, both to clarify what a question is asking for and when hitting roadblocks (being sure to include **BOTH** the code involved **AND** any error messages or descriptions of bizarre behavior).
- **Follow the design recipe for all programs.** We will be discussing what this means throughout the semester.
- Always submit SOMETHING for an assignment, even if it is not complete.
  - Note that significant credit is given for following the design recipe steps, to encourage good programming habits.
  - Also, I believe in partial credit on homeworks, believing that if you have at least started working on a problem, the posted example solution will be more helpful/understandable than if you have not.
- Compare their homework solutions to posted example solutions when they become available.
- Study with others for exams, and practice explaining concepts to one another.
- Attempt every exam problem, and carefully study over exams when they are returned.
- Practice programming as much as possible.

Academic Honesty:

Students are responsible for knowing policy regarding academic honesty. For more information, visit:

https://www2.humboldt.edu/studentrights/academic-honesty

Observe that among the actions that are unacceptable are submitting another's program, code, or file as your own and failing to quote material (that includes algorithms, code, and comments, too!) taken from another person's work. (Note that copying another student's comments is also unacceptable.)

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 then each pair-programmed file turned in will include both of the names of the students who worked on it as a pair.)

(**Important aside**: pair programming specifically means that two people sit at one computer, with one typing while the other says what to type. Both people are actively involved in the programming process. Pair-programming is **not** two people working at two computers, each doing different parts of the work individually. Pair-programming is also not one person doing all the work while the other does nothing or does something else. If pair-programming is ever explicitly permitted, then you are expected to actually pair-program any files you do not complete on your own.)

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

For homework assignments (that are not explicitly specified as permitting 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, in general, students may not work together to complete homework assignments, one student should not instruct another in how to write the code for a homework assignment, and **any type of copying or modifying of another person's computer files, OR of providing computer files to another, related to homework assignments 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.

Asking Questions/Getting Help:

- Sending questions by e-mail can be a very effective way to ask for help.
  - Include `CS 111` along with the subject of your e-mail in the `Subject:` line of any class-related e-mail that you send me. This will help your e-mail be more recognizable as a class-related message, and will make it less likely that I will accidentally overlook it.
  - ALSO include a descriptive subject along with the `CS 111` in that `Subject:` line -- this also increases the chances that I will notice and reply to your question more promptly. (In particular, do not just reply to a class e-mail message I have sent previously, and do not simply leave the `Subject:` line blank!)
  - That said, if I have not replied to your e-mail within 24 hours, please re-send it, just in case I did overlook it somehow.
  - You are expected to sign each e-mail you send me with your name -- sometimes the sender's identity is not obvious from one's e-mail address, especially for an off-campus e-mail address.
  - Also, DON'T INCLUDE the word "`password`" in your e-mails to me -- `pwd` is a handy abbreviation to use instead -- because, due to phishing scams, HSU's spam filtering does not seem to like e-mails with that word in it! (Odd, but this was definitely the case in Spring 2010...)
- I try to check my e-mail (`st10@humboldt.edu` or `sharon.tuttle@humboldt.edu` or `smtuttle@humboldt.edu`) about once a day on weekdays, and about once over each weekend. This is another reason to start assignments early, so that you have time to receive a reply to any questions that might arise.
- 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 homeworks promptly after they are made available from the public course web site, and who practice course concepts as much as possible.

- It is better to ask a question sooner than later -- for example, it is better to send an e-mail with a specific question as soon as you think of it than it is to wait a day or two until the next class meeting or office hour. If you wait to ask such questions, you may not have time to complete the assignment.
- It is perfectly reasonable if you send me a question and then end up finding out the answer yourself before you receive my answer; likewise, it is not a problem if you end up sending me several questions in separate e-mails (as you work on different parts of a homework while awaiting earlier answers).

- 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 run my function: (paste in the function's code), I receive the following error message: (paste in the error message) 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 program. Is it right?")

Additional Coursework-Related Policies:

- You should not expect to be able to finish course assignments during the lab sessions -- although you may *occasionally* get some lab time to work on course assignments, typical lab sessions will include a lab exercise that is to be completed in lab. Even when you finish the lab exercise early, it will still be the case that, like any college-level course, you should expect to put in a significant amount of time outside of scheduled class meetings (lectures and labs) doing the assigned reading, working on course assignments, and practicing concepts discussed.
- Each assignment must be submitted as is specified on its handout to be accepted for credit. This may vary for different assignments. Often, parts of assignments will be submitted using a special tool on nrs-labs.
  - That said, if for some reason you cannot properly submit some parts of an assignment as specified by the deadline, e-mail those parts **before** the deadline and **then** submit them properly as soon as you are able.

    (The e-mailed parts will show that you completed those parts by the deadline even though they were officially submitted later.)
- Each assignment will be clearly marked with one or more due dates (a single assignment could have multiple parts with multiple due dates).
  - **In general, no homework problems will be accepted late. If you wish to receive any credit for an assignment or a homework problem, 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 an assignment for the last minute and there are technical problems, you still must turn in whatever you have by the deadline. As with any work done on computer, make frequent back-ups of your files!

    (If there are unusual/extenuating circumstances such that you think there should be an exception to the above for you for a particular homework problem, you must e-mail or see me as soon as possible explaining why. Note that you help your case if you can show that you have been working on the homework throughout the week -- and not just at the last minute -- by having submitted parts of the homework **throughout** the week.)
  - You may submit **multiple versions** of assignment files and homework problems before the deadline; I will grade the latest pre-deadline submission unless you inform me otherwise. This is to encourage you to turn assignment parts in early (since you will know that you can always turn in an improved version if further inspiration strikes). You also don't have to worry about forgetting to submit something that has already been submitted!
  - If for any reason you cannot submit course work using the submission tool on nrs-labs (or as specified by the

assignment), e-mail me your homework files as attachments **before** the deadline, and **then** submit the files using the submission tool (or other specified means) as soon as you are able.

(The e-mailed files will establish that these files were completed by the deadline even though they were officially submitted later.)

- The nrs-labs tool that you will be using to submit some assignments 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 assignment. 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!
- It is nearly impossible to write unambiguous specifications. If you have questions about "what she means", get them resolved very early in the development cycle by **asking**.
- There is more to computer commands, expressions, statements, functions, and files than simply whether they "run" or not...
  - Part of your grade will be determined by how well your work meets the written requirements. Work that you turn in is expected to meet handout specifications precisely; when one eventually works within a team on large projects, following the specifications precisely is vital, and can mean the difference between a working product and one that just sits there.
  - Note that work may be graded on **style** as well as on whether it runs properly and whether it precisely meets the homework specifications and requirements. Discussions on style will be ongoing throughout the semester.
  - Likewise, because you will be learning good problem-solving practices in this course as well as programming syntax, part of your homework grade will be based on whether you are following these practices (following the design recipe, including required documentation and tests, etc.) A program that runs but omits these parts may lose substantial credit.
- Some course work may be graded simply 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.

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 (**if** that is still possible according to the University policies for dropping courses).

Additional Course Policies:

- You are expected to read this syllabus and be prepared to verify in a required Moodle activity that you have received it, have read it, and understand its contents.
- Exam dates are given in the course schedule below. Make-up exams are only possible by special prior arrangement or because of a valid medical excuse.
- 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 HSU's Account Settings and Moodle accordingly. Course-related messages from me will include `CS 111` in the `Subject:` line.
- You are expected to check the public course web page and the course Moodle site regularly -- course handouts, homework assignments, examples from lectures and labs, 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.

- 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.
- 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 and lab notes are not posted, although many of the projected examples will be made available on the public course web site. Clicker questions, lab quizzes, and graded lab exercises missed **cannot** be made up later (except for extraordinary circumstances).
- As previously mentioned, during lab sessions, there will typically 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, to practice course concepts, and/or to ask questions about course-related topics.

Campus policies:

The following URL leads to useful links regarding HSU policies, procedures, and resources:

http://www2.humboldt.edu/academicprograms/syllabus-addendum-campus-resources-policies

The following are just a few of the links available from this site.

Students with Disabilities:

Persons who wish to request disability-related accommodations should contact the **Student Disability Resource Center** in the Learning Commons, Lower Library, **826-4678 (voice)** or **826-5392 (TDD)**. You can reach the Student Disability Resource Center's web site at:

http://www2.humboldt.edu/disability/welcome

Please note that some accommodations may take up to several weeks to arrange. 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://pine.humboldt.edu/registrar/students/regulations/schedadjust.html

You can find the University policies for repeating classes at:

http://pine.humboldt.edu/registrar/students/regulations/repeat.html

NOTE THAT THE ADD/DROP DEADLINE IS:

******* SEPTEMBER 5, 2016 *******

...WHICH IS THE DEADLINE TO ADD OR DROP CLASSES WITHOUT A SERIOUS AND COMPELLING REASON. And, please note: it is the **Registrar's Office** that determines what constitutes a "serious and compelling reason".

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

Attendance and disruptive behavior:

Students are responsible for knowing policy regarding attendance and disruptive behavior:

https://www2.humboldt.edu/studentrights/attendance-behavior

- **Late arrival to class:** Please attempt to come to class on time, with your headphones/earbuds/etc. put away and your cell phones/tablets/pads/gadgets/etc. 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, inappropriate or distracting laptop/tablet/smartphone/gadget use, 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://www2.humboldt.edu/businessservices/sites/default/files/images/Emergency-Procedures_1.pdf

...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: August 23, 25, 26

- Reading: from "How to Design Programs", 2nd Edition (HtDP/2e): Prologue: How to Program, and Section I - Fixed Size Data - Chapter 1: Arithmetic
- Topics: Introduction to course; simple and compound expressions; data types (number, boolean, string, image); syntax and semantics
- **Homework 1 out**

Week 2: August 30, September 1, 2

- Reading: HtDP/2e: Section I - Chapter 2: Functions and Programs, and Chapter 3: How to Design Programs
- Topics: Introduction to designing your own functions and to the program design recipe; parameter variables
- **Homework 1 due**, **Homework 2 out**

Week 3: September 6, 8, 9

- (Monday, September 5 - Labor Day Holiday) - does not affect CS 111 this semester
- **FYI: NOTE: Last day to drop a course without a W, without a serious and compelling reason, and without it counting toward your 18 semester-units drop limit is Monday, September 5.**
- Reading: HtDP/2e: Section I - Chapter 2: Functions and Programs, and Chapter 3: How to Design Programs (continued)
- Topics: Introduction to designing your own functions and to the program design recipe, continued
- **Homework 2 due, Homework 3 out**

Week 4: September 13, 15, 16

- Reading: HtDP/2e: Section I - Chapter 4: Intervals, Enumerations, Itemizations
- Topics: Boolean functions, conditional expressions, conditional functions, enumerations, intervals, and

itemizations, adding the template step to the program design recipe

- **Homework 3 due, Homework 4 out**

Week 5: September 20, 22, 23

- Reading: HtDP/2e: Section II - Arbitrarily Large Data - Chapter 8: Lists, and Chapter 9: Designing with Self-Referential Data Definitions
- Topics: Introduction to lists
- **Homework 4 due, Homework 5 out**

Week 6: September 27, 29, 30

- Reading: HtDP/2e: Chapters 8 and 9, continued
- Topics: More on lists; simple examples of file input/output
- Friday, September 30 - Review for Exam 1
- **Homework 5 due**

Week 7: October 4

- **Tuesday, October 4 - Exam 1**
- Thursday, October 6, and Friday, October 7 - NO CLASS - instructor out-of-town

Week 8: October 11, 13, 14

- Reading: HtDP C++ Transition readings on Moodle: Sections 1, 2, 3
- Topics: Introduction to C++: simple expressions, compound expressions, data types, and functions
- **Homework 6 out**

Week 9: October 18, 20, 21

- **FYI: NOTE: Last day to change a registered class' grade option to CREDIT/NO CREDIT is Monday, October 17.**
  - (that said, also note that courses applying to your CS degree requirements must NOT be taken as credit/no credit -- they **must** be graded with a letter grade)
- Reading: HtDP C++ Transition readings on Moodle: Section 4
- Topics: Conditional statements in C++
- **Homework 6 due**, **Homework 7 out**

Week 10: October 25, 27, 28

- Reading: HtDP C++ Transition readings on Moodle: Sections 5.3, 5.4, 5.5, 5.6, and 5.7
- Topics: Introduction to `cout` (screen output) and `boolalpha`; definition of a C++ program; writing a `main` function; compiling and linking C++ functions to create a C++ program
- **Homework 7 due**, **Homework 8 out**

Week 11: November 1, 3, 4

- **FYI: NOTE: Last day to drop a course with a W, with an approved serious and compelling reason, and**

subject to your 18 semester-units drop limit is Monday, October 31.

- Reading: HtDP C++ Transition readings on Moodle: Sections 5.1, 5.2, 5.5
- Topics: Introduction to local variables; mutation; assignment statements; scope; introduction to `cin` (interactive input); intro to (mutation-based) repetition: count-controlled `while`-loops; adding pseudocode to the design recipe
- **Homework 8 due**, **Homework 9 out**

Week 12: November 8, 10

- **Friday, November 11 - Veterans Day - HSU Holiday - NO CLASS**
- Topics: Some convenient C++ operators: +=, -=, *=, /=, and the increment (++) and decrement (--) operators; pass-by-value; introduction to arrays and the `for`-loop
- **Homework 9 due**

Week 13: November 15, 17, 18

- Topics: Review for Exam 2; (after Exam 2) Introduction to `void` functions; other common `while`-loop styles; introduction to stream-based file input/output
- Tuesday, November 15 - Review for Exam 2
- **Thursday, November 17 - Exam 2**
- **Homework 10 out**

Fall Break - November 21 - 25

Week 14: November 29, December 1, 2

- Topics: Introduction to `void` functions; other common `while`-loop styles; introduction to stream-based file input/output (continued)
- **Homework 10 due, Homework 11 out**

Week 15: December 6, 8, 9

- Topics: "Packaging" a collection of C++ functions into a single `.cpp` source code file; introduction to pointers and dynamic memory allocation; pass-by-reference; review for Final Exam
- **Homework 11 due**

Final Exam:

THURSDAY, DECEMBER 15, 12:40 - 2:30 pm, in SH 108 (unless I announce otherwise)