CS 695 Introduction to Programming
Fall 2014 Syllabus and Information

Table of Contents

General Information

Contact Your Instructor

Name: Jeff Kinne
Email: jkinne@cs.indstate.edu
Phone: 812-237-2136
Office: Root Hall, room A-129

Lecture, Exam, Office Hours

Lecture: Tuesdays and Thursdays from 12:30-1:45pm in Root Hall, room A-017.

Exam: Our final exam slot is Tuesday, Dec 9 from 1-2:50pm.

Instructor Office Hours: I generally in his office and available most MWF's from about 8:30am-4pm. My official office hours are Wednesdays 9:30-11:30am. My office is A-129 Root Hall.

Unix Lab: The regular hours that the lab will be open will be posted to **here on the department's website**.

Website: this google doc, or find a link from kinnejeff.com

Prerequisites

CS Master's student within 1 semester of graduating. That is, CS Master's student in their final year of studies.

Required text

None.

Course Announcements

Announcements regarding the course will be made both during class and via email to your @sycamores.indstate.edu email address. You should regularly check this email account or have it forwarded to an account that you check regularly. You can set the account to forward by logging into your indstate.edu email from Internet Explorer (the "light" version of the webmail client that opens up from Firefox or Chrome does not give the option to forward email).

Classroom conduct

You may not use cell phones, iPods/music players, etc. during class. You should be civil and respectful to both the instructor and your classmates, and you should arrive to class a few minutes before the scheduled lecture so you are ready for lecture to begin on time. You may use your computer during class if you are using it to follow along with the examples that are being discussed. You may not check email, facebook, work on other courses, etc. during class.

Course Description

This course serves as the "capstone/research/etc." course for the computer science master's degree. Students normally take CS 695 in their final semester before graduating. The main goal of the course is that each student partake an independent project - including learning new material, presenting the material to the class, writing a research paper on their project, etc.

During the first part of the course we will review/learn javascript/html/css for making dynamic web pages. Each student will make a website that includes a working demonstration of the algorithm(s) they study. We also learn Latex typesetting, which is a markup language used to produce scholarly publications in computer science.

Course Outline

We will begin the course with a review of html/css/javascript. Students will write simple programs and games in javascript. We then cover Latex, and finally introduce the projects that students will study the rest of the semester.

Grading and Assignments

The students of this course have the following responsibilities: read assigned readings before lecture, attend lecture, complete homework assignments, take in-class quizzes, take exams, and complete a project. The final grade consists of:

- **Project Paper: 30%** of the final grade.
- **Project Website: 30%** of the final grade.
- **Project Presentation: 30%** of the final grade.
- **Class Attendance/Participation: 10% total**. Attendance will be taken at the beginning of each class. Half of your attendance/participation score will consist solely of whether you were present when attendance was taken each day - the total number of days present divided by the number of lectures in the semester. The other half of your attendance/participation grade will be assigned at the end of the semester based on how attentive you were in class throughout the semester.
    - Bonus for regularly checking in with me ...

Expected Amount of Work

My expectation is that an average student will spend about 6-9 hours OUTSIDE of class each week (that is in addition to class time) WORKING PRODUCTIVELY/EFFICIENTLY (not just starring at the computer) to complete their coursework for this class. Some students may spend less time than this, and some students will spend more.

Grade Cutoffs

Around halfway through the semester we will have a project checkin so that I can give you an estimated grade of how you are doing on the project so far. Initially, I will assign the following grades: 93-100 A, 90-93 A-, 87-90 B+, 83-87 B, 80-83 B-, 77-80 C+, 73-77 C, 70-73 C-, 67-70 D+, 63-67 D, 60-63 D-, 0-60 F. I may adjust these slightly so that the letter grades have the following meaning.

[I expect that roughly ⅓ of the class wll get A+/A/A-, and hopefully most of the rest get at least a B-. Some students will certainly get a C. I hope nobody gets an F, but it is possible.]

A+/A
Full understanding of the algorithm(s) you studied, including learning new material not covered in any of your courses. Well-designed website that gives a correct demonstration of the algorithm(s). Well-organized and thorough paper. Presentation demonstrates that you have a

good understanding of the project and have done all the work on your own.

B+/A-
One of the main components (website, paper, presentation) is not as good.

B-/B
At least one of the components (website, paper, presentation) is pretty good while the others need more work.

C/C+
You have done the project but none of the components is up to standards.

F
You cheated by taking code without citing it. Or you have not really done the work. Or you completely skipped doing one of the required components.

Academic Integrity

Please follow these guidelines to avoid problems with academic misconduct in this course:

- **Note on sources:** if you use some other source, the web or whatever, you better cite it! Not doing so is plagiarism.

- **Projects:** You should not copy from the internet or anywhere else. The project should be your own work. It will be fairly obvious to me if you do copy code from the internet, and the consequences will be at the least a 0 on the project.

If cheating is observed, you will at the least receive a 0 for the assignment (and may receive an F for the course), and I will file a Notification of Academic Integrity Violation Report with Student Judicial Programs, as required by the university's policy on Academic Integrity. A student who is caught cheating twice (whether in a single course or different courses) is likely to be brought before the All-University Court hearing panel, which can impose sanctions up to and including suspension/expulsion. See the Student Code of Conduct and Academic Integrity Resources for more information.

Please ask the instructor if you have doubts about what is considered cheating in this course.

Special Needs

If you have special needs for the classroom environment, homeworks, or quizzes, please inform the instructor during the first week of classes. If you have any such needs, you should go to the Student Academic Services Center to coordinate this. See Student Academic Services Center - Disabled Student Services for more information.

Assignments

Assignments will be posted to this document and announced in class. Things will be listed here most recent first.

Presentation …

- **Your intended audience is CS students who have never heard about your project. I want to give you A's and B's.**
- Check https://docs.google.com/spreadsheets/d/14Timm9pnPvWtlqFz3dv62xfZrk2HbwBIOu8icy3LQB0/edit?usp=sharing for time, date, location
- Either have 15 or 20 minutes. People that are doing same topic can coordinate, or not. If you are coordinating, I still have to grade each of you. So make sure each of you demonstrates that you know stuff, have done stuff, etc.
- Your 15-20 minutes includes time for me to ask you questions and make suggestions. Actually, I'll probably steal about 30% of your time.
- I'll try to have at least one other professor at each presentation.
- You are all invited to each other's talks as well. People that are doing the same topic should be at each other's talks.
- Grade…
    - 30% of final grade is presentation.
    - I will fill out https://docs.google.com/document/d/1HZsHTyY1hi4AQ1VdADdL7tpaxd32mwiDEJ9P58QLklE/edit#heading=h.txe5g6bythpu form and use that as basis for grade.
    - In particular, looking at…
        - 20% basic programming and data structures,
        - 20% algorithms and analysis,
        - 20% overall ability to manage the project (did you get done what I asked you to, and did you get done timely fashion),
        - 20% how independently were you able to work,
        - 20% presentation skills (how clearly do you explain things)
    - Note that how you do on the presentation influences your grade on the other things as well. So do well.
- Basic order of doing things…
    - .25 min - My name is, and I'm doing …
    - .25 min - Go over to your website, and start …
    - Sometime in the presentation you want to …
        - .25 min - Point out the link to your paper for further information.
        - 30-70% of the time - Use visual aids of some sort

        - For people doing basic algorithms, show your javascript algorithm.
        - For people doing drupal, demo-ing your site and how it managed.
        - For people doing bio, some graphics and what they mean, or something.
        - Etc.
        - You can use PPT if you want. Or not.
    - 30-70% of the time - Show some code that you have done - show the overall structure of the program. For example, you show main (or html page for javascript programs), and you have your code nicely split into functions so main only takes up 1 or 2 screens.
    - Say something about efficiency, running time, etc. - that should be while you're doing the above.
    - What's left to do, what would be next, what did you learn, what would you different next time.

Final Paper…

- **Your intended audience is CS students who have never heard about your project. I want to give you A's and B's.**
- 30% of total grade.
- Due the end of exam week.
- Also use https://docs.google.com/document/d/1HZsHTyY1hi4AQ1VdADdL7tpaxd32mwiDEJ9P58QLklE/edit#heading=h.txe5g6bythpu as a guide.
- 20% basic programming and data structures - this has to do with your actual code.
- 20% algorithms and analysis - from the paper itself
- 20% overall ability to manage the project (did you get done what I asked you to, and did you get done timely fashion) - paper and code combined.
- 20% how independently were you able to work
- 20% writing skills - paper
- Note: you get 0 if you don't have a paper written in latex compiled into a pdf. Leave your latex files somewhere in your account so I can look at them if I want to.

Final Website

- **Your intended audience is CS students who have never heard about your project. I want to give you A's and B's.**
- Due end of exam week. But should be already nice looking, having information, when you do your presentation.
- Required stuff:
    - link to pdf file.
    - basic information about yourself.

    - some graphics and/or code and/or text describing your project/algorithm.
    - Don't have to have all your code available on the website, do have to have all your code on the server.
- Grading…
    - 33% Has required information.
    - 33% Looks nice, is easy to use.
    - 34% Clarity of information, good presentation of information.

Final Project To-Do List

These are things you MUST do. Suggested finished-by dates should be followed so that you stay on track.

1. Finish the part of HW1 where you create a public_html directory in your CS account, create a index.html or index.php file in the public_html directory, and set the permissions with chmod to make sure the page can be viewed on the web at http://cs.indstate.edu/~yourusername/
2. Make your index page include basic information about you - your name, say that you're at ISU and what degree you're working on, and basic description of your interests. Include a few favorite links. Include a section about your project; for now just have a sentence or two about which project option looks the most interesting to you, or if you really wanted to work on something else.
3. On your index page, link to another web page that is a page demonstrating an algorithm in javascript. You can copy one of the examples from class for this, but the page running the algorithm needs to be inside of your account. You should make it look nicer than the examples from class.
4. On your index page, include a link to a javascript/html page that does an animation of some kind. Basically that means having a canvas element, and then javascript to do an animation on the canvas. Make it pretty simple so that the code is short and easy to understand. Include comments in the html and javascript explaining how it works.
5. Copy paper.tex, paper-bib.bib into your directory and start putting your information into latex… Put a preliminary version of your pdf in your public_html directory with a link from your webpage. You get the paper.tex and paper-bib.bib at http://cs.indstate.edu/~jkinne/cs695-f2014/ or at ~jkinne/public_html/cs695-f2014 when you're logged in to the CS server.
6.

Reading/Viewing Assignments

Reading/viewing assignments are listed according to their DUE DATE. That means you should have that reading or video viewing done before class on that day. Note - these are not graded.

- 8/26 - maybe look at http://mathcs.indstate.edu/dept/acm/pconn.php#summer2014
- 8/26 - W3Schools Javascript Tutorial

- 8/26 - Week 9 of CS 50 Harvard course

Programming Assignments

Note - these are not graded. But you should do them to make sure you are learning what you need to learn.

HW 2 - javascript something

- Game - Crosses and Naughts, X's and O's, Tic Tac Toe
    - including computer.
- Game - Reversi
- search.html…
    - copy/paste into your directory and make sure it works
    - When the number is found, put a message at the top of the page that says "found it". In the html, add a tag where the message will go, and give an id so you can get to it later. In the javascript, when the number if found, use the document.getElementById and .innerHTML to update that tag.
    - Add a tag in the html to show the total number of steps so far, and keep that current.
    - Add in a demo for binary search… Add a button for "Binary search step". When they click that, first sort the list. Make the middle red, make the top or bottom gray (whichever is eliminated). Keep track of which part of the list is being looked at…
- sort.html/sort.js
    - Fix the way it switches colors and bold to look better…
- mst.html, graph.js
    - copy/paste into your directory, make sure it still works
    - look at the code, understand it.
    - make it so right-clicking on a vertex deletes it.
    - make it so clicking on a vertex, and then right-clicking on the next vertex from an edge deletes the edge.
    - change the way the graph is displayed to make it nicer/different in some way
    - add an ability to load a graph from either a file or an input box, including different edge weights than the default euclidean distance.
    - ~~fix the problem with the scrolling - already fixed in the latest code.~~
    - user options - add checkboxes to turn on/off showing the distance, or the vertex labels.
    - other little stuff - play around
    - animation/timer - make all the vertices move slowly
    - put in another graph algorithm!
        - test if the graph is connected - dfs/bfs
        - harder - chromatic number, minimal TSP

    - test if there are any triangles.
    - shortest path

HW 1 - public_html directory and index.html in your permanent account

- Task: If you don't have a permanent account on CS, ask for one. Login and issue the following commands
  cd ~
  mkdir public_html
  chmod a+rx public_html
  cd public_html
  pico index.html
  Then in your browser, type in the address
  http://cs.indstate.edu/~yourusername/index.html

- And then put html code into the index.html. In the index.html file, make a personal website for yourself that says who you are, that you are a master's student finishing up at ISU, some of your favorite links, and some topics you might be interested in for your project. Keep in mind these websites are public and will show up in google searches soon.

Important Links

Javascript

- CS 50: Intro to CS I at Harvard. Parts at the end give an introduction to php, javascript
- W3Schools Javascript Tutorial
- Programming at Khan Academy - learn javascript

Php

Software and CS Server

- Download Putty. For those using Windows at home, download and install Putty. Then watch the first two videos on this youtube playlist to show you how to use it.
- Getting Started with the CS server - links and videos to help you get started with connecting to the CS server.
- Linux Console Tutorial. A detailed tutorial on how to use the Linux console (command line). Not everything it discusses is available to you (e.g. permissions), but it offers a broad spectrum of what can be done.

Other Programming

- Scratch Programming - all online, visual, no syntax errors, share with friends!
- MIT Android App Inventor - online, visual, create apps for android.
- Code.org - links to other beginning programming tutorials, many suitable for kids. The organization is pushing teaching programming/CS in K-12 schools.
- Udacity Online Courses - including introduction to CS in Python, intro to Java

Project Topics

These are the possible project topics so far. A few more will still probably be added.

Bioinformatics

- I am meeting with a new bioinformatics professor (Yongsheng Bai) Wednesday morning to discuss the possibility of one or more of you working on something related to his research. I'll report back on Thursday…
- Yes, it is a thing, it looks interesting…
- Yongsheng Bai, http://thoth.indstate.edu/ybai2/
- Take his code and convert to C/C++ (from Perl)
- Make the algorithms better ...
- Do some optimization …
- Search on bioinformatics algorithms, rna sequencing, …
- Data set
    - /u1/junk/RSW_check_test/
    - 2 readable files - 1 is input to SQL statement, 1 is correct output
    - 1 unreadable file - full input to SQL statement
    - What is in those files…
        - I have put three files under /net/common/ybai/RSW_check_test/ on our
        - bioinformatics server. You could help to copy them over to CS machines.
        - Jeff, a small test file named ³HWI-EAS159:6:7:895:305.read² and its output
        - file ³HWI-EAS159:6:7:895:305.read.inter_yb² could be used to check
        - students' code accuracy. The file ³RSW_test.txt² could be used for the
        - actual run.
        - The headers in the file are listed as:
        - ReadName SplitDirection SplitLength Strand Chromosome MappedLocation SplitReadSequence SplitReadQuality AdditionalMappablePlaces
    - Input to step 1, or something…., from http://thoth.indstate.edu/ybai2/RSW/RSW_steps.pdf
        - I have put mm9 reference genome + normal splice junctions under:
          /u1/junk//RSW_check_test/mm9
          Those are mm9sp35.fa and mm9sp79.fa
        - The original raw read files can be found and downloaded under
          http://www.ncbi.nlm.nih.gov/sra/?term=GSE54631
          And a few of those have been downloaded and put under
          .../RSW_check_test/mm9/
          SRR1157324.sra is the first link from there.
    - http://en.wikipedia.org/wiki/String_searching_algorithm

    - http://en.wikipedia.org/wiki/Sequence_alignment
    - http://en.wikipedia.org/wiki/Alternative_splicing
    -

Other Project Topics

- If there is some other topic you'd like to work on, then make a proposal by Sept 19.
- First, describe what you would like to do.
- Example: path finding algorithms.
    - Description of the problem - a path finding algorithm…
    - Some links showing example -
        - http://en.wikipedia.org/wiki/Pathfinding
        - http://en.wikipedia.org/wiki/A*_search_algorithm
    - Some recent research or papers related to path finding…
        - http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1315730&abstractAccess=no&userType=inst
        - http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4154833&abstractAccess=no&userType=inst
        - http://arc.aiaa.org/doi/abs/10.2514/1.8371?journalCode=jacic
        - You would pick a few to read and understand the basic idea, and what those people did that was new, and what the state of the art is…
        - Maybe a paper from
            - http://www.aaai.org/Conferences/AAAI/2014/aaai14accepts.php#HSO
            - Search for the title of the paper and you'll probably find a PDF, like http://www.cs.toronto.edu/~jdavies/bacchusAAAI14.pdf
    - Say you have a research paper that looks interesting…
        - YOu can look at the papers that paper references.
        - And search for the paper on google scholar and see if there are newer papers citing the one you're looking at.
    - So, basic understanding of what other people are doing.
    - Now, what do you want to do?
        - Recreate what they did.
        - Or, try something different/better.
- Second, look up a few research papers related to the topic.
    - One place to look is at CS conferences related to the topic. Look at http://en.wikipedia.org/wiki/List_of_computer_science_conferences and find a conference that is related. Search on google for that conference, and look at recent papers for some that are related.
    - Another place to look is doing a search at scholar.google.com.
    - Another place to look is in some textbook you have used, and look up the references in the back of the chapter or back of the book.
- Send the information to me in an email.

- Note that you can work on a project, but there must be a component of the project that is “research-y” and looking at the latest research. There also must be a component where you do some serious programming.

Search for Large Prime Numbers

- Building on work by Jeff Kinne, Geoff Exoo, and students who worked in the summer of 2013.
- For reference information see http://cs.indstate.edu/Summer/
- Also see http://primes.utm.edu/
- Current goal is beating the world record for “near repdigit” primes listed at http://primes.utm.edu/top20/page.php?id=15
    - Types of numbers to test, probably 9’s with a single 8 - more common than others...
- Basic strategy
    - Starting at some digit length, trial divide and Fermat test all numbers of the form 999999….989….999 - where the position of the 8 in the sequence is at most ⅔ from beginning. When one passes the Fermat test, confirm it is prime with PFGW.
- Tools to use
    - C++ programming - use gmp library for large integer arithmetic. Code up the trial divide and Fermat tests using gmp.
    - OpenPFGW - can be used to prove a number is prime if the number +/- 1 is mostly factored.
    - Microsoft Visual Studio - develop the program on Windows as well, possibly having it run as a system service so the lab computers around campus can be tasked to run the program while they are idle.
- Another direction to pursue…
    - Making GPU programs to do this stuff...
- Papers to read
    - References at the bottom of http://primes.utm.edu/top20/page.php?id=15
    - Look for newer literature as well.
-

Graph Ordering and Forbidden Induced Subgraphs

- Current work by Arash Rafiey, Jeff Kinne, Geoff Exoo on graph orderings and forbidden induces subgraphs.
- See http://cs.indstate.edu/~jkinne/cs695-f2014/forbid-4.html
- Papers to read
    - http://www.sfu.ca/~arashr/publication/ordering.pdf
    - Other newer/older papers on similar topics…
- Basic strategy

- Optimize the brute force program from forbid-4.html, write it also in C/C++ to make it faster, possibly make it run over all the CS computers as well.
- Use brute force program to look for patterns - certain types of sub-graphs that rule out a graph (e.g., C_5).
- Use brute force program to check all graphs that are small, or to pick random graphs from those that are larger. Get some statistics on probability a random graph is in this class or not (depending on the sparsity of the graph).
- Try to come up with a poly-time algorithm for the problem. Test the algorithm against brute force to make sure it is correct on small graphs. Prove the algorithm is correct.
- On the other side, try to prove the problem is NP-complete by reducing other known NP-complete problems to it.

Drupal CMS Administration

- The university is switching over to using drupal to run many of its web pages. This project would involve both becoming familiar with the system and installing it, and looking at recent literature on drupal.
- Basic strategy
  - Get drupal installed on your own computer, and go through tutorials on installing, configuring, and administering it. Keep notes in a document of important choices in the configuration/administration.
  - Work with Steve Baker on getting an installation put on one of the CS servers so you can test it out “live”. Possibly run the student ACM website or the CS group pages off of the drupal server.
  - Investigate latest research on efficiency, security, and usability of drupal.
  - Check out drupalgardens.com
  - Also, Chinmai and Ashok know something about this...
- Papers to read
  - TBD

AI Leaf Identifier

- The idea is to develop a good AI machine learning algorithm for identifying trees based on a picture of their leaf and/or bark. Existing tree identification programs often ask the user questions (is the leaf shaped like __ or __, etc.). This program should be fully automatic.
- Basic strategy
  - Begin with literature review looking for best similar programs out there.
  - There are two different basic approaches I see. One is to use something like neural nets or support vector machines to do a “classification based on learning from examples” approach. Another is to do a graphical analysis of the leaf to break it up into its parts to automatically identify the structure and characteristics

of the leaf. Possibly 2 different groups of students could work on both approaches.
    - The goal would be to develop something that would be able to run on a phone. You can decide whether you'd focus on Android or iPhone. To get started, it may be easier to start with a version that runs on CS or your own computer.
    - Note - you can start with easier tasks. For example, distinguishing between just oak and maple. After that, maybe just the top 20 most common trees in Indiana.
    - Note - you can also give some probabilities as output - e.g., 70% chance it's an oak… Kind of a bayesian inference, expert system, kind of thing...
- Papers to read
    - See some of the papers listed at http://scholar.google.com/scholar?start=10&hl=en&as_sdt=800005&sciodt=0,15&cites=17525617266456647645&scipsc=
        - For example http://dl.acm.org/citation.cfm?id=2461520
        - http://dl.acm.org/citation.cfm?id=2403044
        - http://dl.acm.org/citation.cfm?id=2562082&CFID=558426797&CFTOKEN=91792773
        - ...

Course Schedule and Notes

Things will be listed here most recent first.

- 9/18
    - Bio talk 2pm or 2:30pm Monday. I'll send an announcement.
    - Let's make up some due dates…
        - Right now - project proposal on your cs website. I'll look at those next week and say yes or no. Convince me you've done some work and have a realistic chance of getting something done. Give links to things you've looked at. Give some descriptions that show some understanding.
            - And I'll categorize the project as either A, B, C - meaning that if you do a good job that is roughly the grade the project would be worth. So I may let you do a project that isn't very "research-y", but I'll tell you "well that's a B project, if that's okay with you then you can do it".
        - Mid-October, I give you a preliminary grade on the code.
        - Final code, website due Nov 14, before Thanksgiving.
        - Final paper, website everything due Dec 5.
- 9/16
    - I go around and you show me what you have so far for your proposal and/or website, etc.
    - Bioinformatics - I'll meet with him Monday and then I'll know what's going on. Next Tuesday we could talk about bioinformatics. If you're going to bioinformatics, you should start looking at stuff on your own.
- 9/11
    - Project proposal due by 9/19. Put the proposal as your index.html or index.php file in your public_html directory. You should pick one of the topics listed above, and provide some description of what you're thinking. Or, if you want to propose something different you have to talk to me about it first…
    - Bioinformatics - hopefully Dr. Bai will do a presentation for us, I'll let you know.
    - Drupal - there has to be some "research" part - look up papers about security, efficiency or something…
    - Algorithm X - it's fine to start with something you know well already, then you'll look at better/newer/different ways to solve the problem.
    - What if lots of people pick the same topic? I'll choose the best proposals and approve those, and suggest the others choose something else.
    - Note - you'll have to convince me that you're going to do something nice, that you've already started learning something, …
    - Project related to gaming?
    - Research/conferences related to Drupal?
    - Look at acm.org… dl.acm.org...

- 9/4
  - Things people say they're interested in…
    - AI, machine learning
    - Something new
    - graphics, games
    - database/warehousing
    - A few other topics...
  -
- 9/2
  - Graphs and MST
  - array in javascript
  - classes in javascript
  - canvases in javascript/html5
  - Make sure your index.html or index.php is updated with topics you are interested in for the project.
  - Note - console.log('hello...'); - messages show up in the javascript/web console if you have it open.
- 8/28
  - Next up - graphs...
- 8/26
  - Attendance in blackboard
  - Today's example: searching through an unsorted list
    - http://cs.indstate.edu/~jkinne/cs695-f2014/ then search.html
  -
- 8/21
  - Attendance
  - Javascript/php/html - starting Tuesday
  - Problems/algorithms
  - Latex
  - Projects
  - Grades

Email Log

If I remember, I'll copy/paste emails to the class here so you can refer back to them in case you accidentally deleted one.

-