CS271 Syllabus - Spring 2022

Course Name: Computer Architecture and Assembly Language
Course Number: CS271
Credits: 4
Instructor name: Justin Goins
Instructor email: justin.goins@oregonstate.edu
See the Course Syllabus Page (in Canvas) for information on TAs or Office Hours.

_____

1. Course Description

1.1. Introduction to functional organization and operation of digital computers. Coverage of assembly language; addressing, stacks, argument passing, arithmetic operations, decisions, macros, modularization, linkers and debuggers.

2. Prerequisites

2.1.	Enforced Prerequisites:	CS 161
2.2.	Other Prerequisites:	CS 225 or MTH 231
2.3.	Courses that require this class as a prerequisite:	CS 344

3. Communication

3.1. Any email sent to the instructor about this course **must** originate with an OSU supplied email account and contain the tag **[CS271]** at the beginning of the subject. Failure to comply with this will result in delayed (or possibly nonexistent) response to your email.

3.2. Office hours will be held online using the Zoom lecture room. See the Canvas website for access information.

4. Technical Assistance

4.1. If you experience computer difficulties, need help downloading a browser or plug-in, assistance logging into the course, or if you experience any errors or problems while in your online course, contact the OSU Help Desk for assistance. You can call (541) 737-3474 (USA), email osuhelpdesk@oregonstate.edu or visit the OSU Computer Helpdesk online.

5. Learning Resources

5.1. This course does not have a required textbook. Canvas contains access information to the learning resources that you will need for this course.

6. Canvas and Lectures

6.1. This course will be delivered primarily via a combination of lectures and Canvas content. We will meet on campus for lectures, in-class demonstrations, and exams. Learning materials such as the syllabus, weekly content, assignments, projects, and quizzes and available within Canvas.

6.2. Canvas is optimized for the most recent versions of most popular browsers. If your browser of choice is an out-of-date version, you should update for use with Canvas, especially for quizzes and exams. If you are having browser troubles, seek out the Technical Assistance described above.

6.3. When I have general announcements for the class I will post them on Canvas. It is your responsibility to keep up with announcements and messages posted on Canvas.

7. Course Learning Outcomes

7.1. Access and interpret binary data stored in memory.

7.2. Illustrate the Instruction Execution Cycle.

7.3. Create and analyze well-modularized assembly language programs utilizing decision, repetition, and procedure structures.

7.4. Utilize a debugger to identify and correct bugs in assembly language programs.

7.5. Illustrate the system stack as it is used for procedure calls and parameter passing.

7.6. Illustrate the primary components of a modern computer architecture, and explain their function.

8. Evaluation of Student Performance

Syllabus Quiz	1%	Open Resource, ∞ attempts, unlocks first module
Project #0	2%	Initial IDE Configuration, must configure your own setup
Project #1	6%	Your own work only
Project #2	3%	Debugging Lab, may brainstorm together
Project #3	6%	Your own work only
Project #4	6%	Your own work only
Project #5	8%	Your own work only
Project #5	10%	Your own work only
Module Summary Exercises (x10)	20% (2% each)	Open Resource, 2 attempts, 24 hour limit
Quizzes (x2)	10% (5% each)	Open Notes, 1 attempt, 1 hour limit
Midterm	12%	110 min limit, on-campus, notesheet only
Final	16%	110 min limit, on-campus, notesheet only

8.1. The grading scale is as follows:

93+	A
90 – 92.99	A-
87 – 89.99	B+
83 – 86.99	B
80 – 82.99	B-
77 – 79.99	C+
73 – 76.99	C
70 – 72.99	C-
67 – 69.99	D+
63 – 66.99	D
60 – 62.99	D-
0 – 59.99	F

8.2. Other important grading criteria:

8.2.1. **All programming projects must be submitted in order to pass the course.** Students missing programming projects at the end of the term will receive an F grade.

8.2.2. If you wind up with a grade average of 89.99%, you should expect a B+ in the class.

8.2.3. The intention is to give you a lot of practice. By getting the practice, you perform better and more quickly on the exams (and the programming projects).

9. Module Summary Exercises

9.1. The weekly summaries are open exploration, open Internet, and open lecture. You can use just about anything (including classmates) while taking a weekly summary.

9.2. Weekly summaries have a time limit of 24 hours (so I don't end up with partial tests).

9.3. You will be able to take each weekly summary at most twice. The recorded score will be the higher of the two scores you receive.

9.4. Weekly summaries cannot be taken after the due date.

9.5. Weekly summaries are present to help you pull together the material from the week. This will help you with the quizzes, the exam, and (importantly) the programming projects. The syllabus quiz counts as a weekly summary.

10. Quizzes

10.1. There are 2 quizzes in this course. Quizzes are open note, open Internet, and open exploration. You can use just about anything **except** your fellow students while taking a quiz. Quizzes are not proctored. You will be able to **take each quiz once**.

10.2. Quizzes cannot be taken after the due date.

10.3. Quizzes are taken on Canvas and become available on Thursday of the week they are due.

10.4. **Quizzes in this class are timed**. You won't be able to exceed the time limit on the quizzes. It is not the intent of the quizzes to be time pressured, but to ensure that you can pace yourself and do the work without relying excessively on external help.

11. Exams

11.1. The midterm and final exam are **open notesheet**.(8.5" x 11" sheet of paper, front and back)

11.2. The exams in this course will be taken on-campus during scheduled lecture times.

11.3. You are allowed to use a calculator, scratch paper, and your notes.

11.4. You are **not** allowed to utilize anything else during an exam.

11.5. Makeup Exams

11.5.1. Makeup exams will be given only for missed exams excused in **well in advance** by the instructor. Excused absences will not be given for airline reservations, routine illness (colds, flu, stomach aches), or other common ailments. Excused absences will generally not be given after the absence has occurred, except under very unusual circumstances.

11.6. Exam Time Limits

11.6.1. Exams in this class are timed (you are allowed to use the entire lecture window).

12. Incompletes

12.1. Incomplete (I) grades will be granted only in emergency cases (usually only for a death in the family, major illness or injury, or birth of your child), and only if the student has a passing score at the time of the request. If you are having any difficulty that might prevent you completing the coursework, please don't wait until the end of the term… let me know right away.

12.2. Completion of an incomplete (I) grade will require **additional** work from you. You won't simply have an opportunity to do the work late; you'll do more, possibly a lot more.

13. Homework (aka Programming Projects)

13.1. The programming projects (homework) are a significant portion of this class and are the most common place for students to struggle. Several things about this class may be new to you.

13.1.1. You may not have used Visual Studio before.

13.1.2. Intel x86 Assembler code will be new.

13.1.3. Programming at the assembler level is very different from using higher level languages.

13.1.4. Stepping through the assembler code in the debugger will be new.

13.2. All programming projects must be submitted by 11:59pm on the due date to Canvas.

13.3. If you wish to petition a grade, you must do so within one week of receipt of the grade, by email to your instructor (or the person who graded the work).

13.4. Late Projects have exactly two (2) days from the due date, no more, to be submitted. Since programming assignments are normally due on a Sunday, 2 days late makes that Tuesday. **Late work is penalized 15% per day.** Any programming project submitted more than 2 days after the due date will automatically receive a grade of zero (0). Don't make the mistake of submitting your assignment late just trying and get the last few points by making it perfect. Perfection is the enemy of done. You want to be done.

13.5. You have the right to make use of two grace days for submission of programming projects, used in increments of one day. The grace days allow you to have an un-penalized late assignment by up to two days or 2 assignments up to 1 day each.

13.5.1. The use of grace days does not extend the last day on which you can submit an assignment to be graded, it is still a maximum of 2 days past the assignment due date.

13.5.2. Grace days don't change the assignment due date, they only change deductions for late. If you use 2 grace days and submit the assignment 3 days late, you've used your grace days and received a 0 on the assignment.

13.5.3. I encourage you to not use up your grace days early in the term. Programming assignments get harder as the as the term progresses. You'd hate to waste grace days on early and easy assignments when the assignments get harder. Start your programming assignments as soon as possible. Do not wait until the last weekend to begin them.

13.5.4. Don't be lulled into over-confidence from early assignments only to be surprised by later assignments. Watching the clock tick past midnight for an assignment that feels far from working is not enjoyable. It causes stress, and stress is bad.

13.6. All source files (.asm files), must include a comment block at the top that contains the following information. Neglecting this information will result in a point deduction. It is easy to do, so please just do it.

13.6.1. Your name.

13.6.2. Your OSU email address.

13.6.3. The class number, and section (CS271-001).

13.6.4. The assignment number.

13.6.5. Assignment due date.

13.7. Don't miss submitting a programming assignment. You are much better off to submit a partially functional assignment than to not submit anything for an assignment. As stated above, you must submit **all** programming projects for the class in order to pass the class.

13.8. Your programming assignments must run in Visual Studio to be graded. If your assignment does not run in Visual Studio, then you will get a zero for a grade. Running under some other assembler or emulator in addition to VS is fine, but it must still run under Visual Studio.

13.9. You must submit all your assignments through Canvas.

13.9.1. Submit your work for each assignment as a **single asm** file through Canvas. You should not need to submit any additional files for a programming assignment. If you use an external library other than the Irvine32 library, your code will fail to assemble in

the standard Visual Studio environment that we use to grade the assignment. That means you will be disappointed with your grade. We expect you to make use of the Irvine32 library and no other libraries in your code. If you need to comment on your code, place your comments into the asm file.

13.9.2. You can submit your assignments more than once through Canvas. Each will be time stamped. We will grade only the last one submitted.

13.10. All work must be done individually unless specifically allowed to work in groups. The homework programming projects are **not** group projects. You must do your own programming projects. You may share pseudo code and ideas about how to solve or approach problems, but write your own code. If you are getting odd assembler messages, you can share the snippet of code that is producing the message; you don't need to share the entire file.

13.10.1. There are some very good tools for detecting when students are copying code from each other. I have used those tools. I have found students who copied code. No one enjoyed it, and I'm sick of failing students for it. Programs **must** be done individually.

13.10.2. Copying code is a violation of the student conduct policy. If you are caught copying code from others, you will receive a grade of 0 on the assignment. Even if the code comes from a previous student's work, it is still copying.

13.10.3. **We reserve the right to ask you to explain a complicated piece of code**. If you cannot explain your own code to us, you will receive a grade of 0 on the assignment and I will initiate an investigation into possible academic misconduct.

13.11. If you are struggling on a programming assignment, the first thing you should do is make sure you have read the provided material for the week (and prior weeks) and watched the lectures (review the notes you've taken).

13.11.1. The lectures often include examples of those same topics.

13.11.2. Sending your entire source code to the instructor or TA with a note saying "Something is wrong, can you fix it?" is unlikely going to get you the response you want. The instructor and TA are not debuggers.

13.11.3. Run the program in the Visual Studio debugger yourself. The way to get better at debugging code is to use the debugger.

13.11.4. Make sure you read the entire assignment. There can be some really useful information in all that text.

13.12. We will be using Visual Studio as the development environment for this class, specifically MASM (Microsoft Macro Assembler). You can use most versions of Visual Studio after 2010, but I recommend Enterprise 2019. If you use some other assembler (e.g. NASM), your code likely will not assemble and you'll lose most/all of your points. If you don't already have Visual Studio, you can get it (free for student use) through Microsoft Azure (See the Tools tab inside the Canvas Syllabus page for instructions).

13.13. Feedback for your programming assignments will be given through Canvas. Once a programming assignment is graded, the rubric results will be available from the Grades tab in Canvas. Please take the time to read this. It is one of the important ways to learn in the class.

13.14. This is important feedback. You don't want to repeat this sort of error on following assignments. The rubric will also identify by whom you assignment was graded, making it much easier to contact her/him if you have questions about your grade.

13.15. If you are unable to locate the feedback on your assignments, ask a TA to guide you to it.

13.16. It is your responsibility to keep up with your assignment/exam/quiz/summary grades and initiate contact if you have a question.

14. Keys to Success

14.1. This class requires a keen attention to detail. Particularly when you are working with an unfamiliar x86 instruction (and they all start as unfamiliar). Some of the keys to success are:

14.1.1. Attend lectures and interact with others.
14.1.2. Read the weekly modules (and take notes).
14.1.3. Complete the weekly summary exercises.
14.1.4. Start the programming assignments early and complete them on time.
14.1.5. Don't get discouraged if your code initially does not execute correctly; many problems are simple to fix and it just takes time to isolate and identify the problem.
14.1.6. Although mentioned in the course lectures, this cannot be emphasized enough, **learn to use the Visual Studio debugger**. It's much faster to troubleshoot a problem while using the debugger, so take the time to understand how it works.
14.1.7. Assemble your code often. Writing a few lines of code and then double-check it to make sure that it assembles correctly. This is especially important when you are first learning to program in assembly. By assembling your code frequently you can locate mistakes more quickly and have a better idea of where a problem originates.

14.2. When struggling with a homework assignment:

14.2.1. Create tests to isolate the problem.
14.2.2. Use the debugger (while running the test cases)
14.2.3. Look for/create a reply in Piazza.
14.2.4. Ask questions during office hours.
14.2.5. Email the instructor and/or TA

14.3. The best key to success in this class is to keep up. Don't let yourself fall behind.

14.4. Another valuable asset to have in this class (and the entire program) is a study group. I recommend posting on Piazza to get connected with some people near you, or at least who have similar schedules.

14.5. One of the skills you'll develop in this class is how to look for things in the resources listed above. You'll spend some quality time with your favorite Internet search engine. You'll learn how to wade through the chaff of Stack Overflow to find the relevant example from the hundreds of search hits. You'll probably be able to remember the page number of certain examples from the textbook. Some of this won't be fun, but you'll learn that you can learn it.

14.6. Make sure the code you submit actually assembles and links. If your code does not assemble (using MASM in Visual Studio), you will receive a zero (0) for that portion of the programming project. I'm not going to try and guess what portions of your code may be correct if it does not assemble. If you are unable to get your program to assemble correctly, comment out the portion of the code that causes the assembly process to fail. You are better off getting partial credit on a programming project than getting a zero for code that does not assemble.

15. Statement Regarding Students with Disabilities

15.1. Accommodations for students with disabilities are determined and approved by Disability Access Services (DAS). If you, as a student, believe you are eligible for accommodations but have not obtained approval please contact DAS immediately at 541-737-4098 or at http://ds.oregonstate.edu. DAS notifies students and faculty members of approved academic accommodations and coordinates implementation of those accommodations. While not required, students and faculty members are encouraged to discuss details of the implementation of individual accommodations.

16. Student Conduct

16.1. Student conduct expectations: https://beav.es/codeofconduct

17. Academic Honesty

17.1. Students are expected to do their own work. The only sources you're allowed to use code from are the explorations, and you must make a comment with the Module/Exploration. Direct use of any other resources is prohibited.

17.2. We use software designed to find similarities between programs. Each individual's program is compared to every other individual's program to find similarities. Please, do you own work.

17.3. Honesty is absolutely essential in order for learning to take place. It will form the foundation of your professional integrity in your career.

17.4. If you are having trouble with an assignment, you are encouraged to discuss it with other students, TAs, the instructor, or anyone else, but don't just have someone else tell you how to solve the problem! If other students ask you for help, don't just let them copy your work! It is possible to discuss problems without plagiarizing. One of the best methods of debugging is to explain your solution to someone else.

17.5. If you get help from, give help to, or work together with someone, you must (in the program header block) list that person as a collaborator and describe the help. Programs that are very similar will be subjected to review unless both programs indicate that they were produced collaboratively. We use plagiarism-detection software to check your code against the code from other students. It is quite sophisticated and can easily see through variable name changes and formatting differences.

17.6. If you are found in violation of any of the above policies, whether you are the giver or receiver of help, you will receive a zero on the assignment. An investigation will be initiated with the Office of Student Conduct. A first offense normally results in a zero grade on the assignment. Future offenses often include more severe repercussions.

18. Reach Out for Success

18.1. University students encounter setbacks from time to time. If you encounter difficulties and need assistance, it's important to reach out. Consider discussing the situation with an instructor or academic advisor.Learn about resources that assist with wellness and academic success at oregonstate.edu/ReachOut. If you are in immediate crisis, please contact the Crisis Text Line by texting OREGON to 741-741 or call the National Suicide Prevention Lifeline at 1-800-273-TALK (8255)

19. Student Bill of Rights

19.1. OSU has twelve established student rights. They include due process in all university disciplinary processes, an equal opportunity to learn, and grading in accordance with the course syllabus: https://asosu.oregonstate.edu/advocacy/rights

20. Academic Calendar

20.1. All students are subject to the registration and refund deadlines as stated in the Academic Calendar: https://registrar.oregonstate.edu/osu-academic-calendar