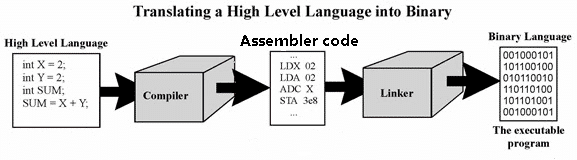
CS 255-1: Computer Organization and Assembly Programming



**Course Description and Objectives**

Syllabus Contents

Course description and objectives: . . 1

Course resources: . . . . . . . . . . . . . . . . . 1

Taking this course: . . . . . . . . . . . . . . . . . 2

Class policies . . . . . . . . . . . . . . . . . . . . . 2

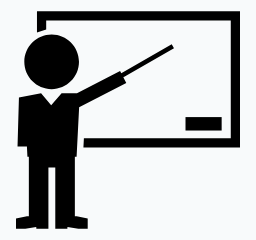
Grading . . . . . . . . . . . . . . . . . . . . . . . . . 2

Lecture format: . . . . . . . . . . . . . . . . . . . 3

Class schedule: . . . . . . . . . . . . . . 3

**This course is part of the core course requirement for the Computer Science BA/BS degree program. The pre-requisite for CS255 is CS170/CS171 where you have learned Java programming, the array, the linked list and the Binary Search Tree data structures used to store information efficiently. The goal of CS255 is to show you how a computer executes a computer program and how it processes information.**

**In CS255, you will first learn how a computer operates at the ``functional’’ level. The program statements that you learned to use in Java are high level operations. Computers operate at a much lower level: the operations that a computer execute are called ``assembler instructions’’. In order to understand how information are processed in a computer program, we must first learn how computers represent information internally. Then we study how a computer store and organize data in the memory and how it executes (1) assignment statements, (2) if-statements, (3) while-statements, (4) function calls and (5) recursions in term of assembler instructions. You will also learn how the array and the linked list data structure are stored inside the computer memory and how to access and manipulate these variables using assembler instructions.**



**Your Professor**

Dr. Shun Yan Cheung, PhD

Office: MSC Room W428, Phone: 404-727-3828

Email: [cheung@emory.edu](mailto:cheung@emory.edu) (preferred)

Office hours: MW 11-12, TuTh 3-4 and by appointment

Zoom office URL: see Canvas

**Course resources:**

|  |  |
| --- | --- |
| **Class website:** | **http://www.mathcs.emory.edu/~cheung/Courses/255/index.html** |
| **Text book: (online)** | **http://www.mathcs.emory.edu/~cheung/Courses/255/Syl-ARM/syl.html** |
| **Online Zoom lecture:** | **See Canvas for the URL** |
| **Online Zoom office:** | See Canvas for the URL |

**Class Policies**

**Taking this course**

Have you ever wonder what happens inside a computer program when you define a variable ? How can a computer drive a car ? Or how a computer program can make decisions using an if-statement ? Why does a parameter that is ``passed-by-value’’ to a function does not get updated by the function call ? What make recursion work ? All these questions and more will be answered in this course.

* Homeworks/projects and their deadlines are posted on the class website and they must be turned in prior to the dead line unless you have requested an extension.
* You can request an extension for homework/project for **upto 3** **times**. Extensions will always be granted and you will get at least 3 extra days when you receive an extension. You cannot request two extensions for the **same** homework/project.
* Requests for homework/project extension **must** be made ***before*** the assignment is due.
* "Free" extensions are allowed for illnesses, accidents or personal matters that require you to leave for home. “Free” extensions are not counted towards the maximum of 3 allowed extensions. In order to receive a “free” extension, I will need documentation, like a doctor's note. A "free" extension must requested **ASAP**.
* OAS students can make unlimited extension and receive extra time for exams when OAS status approved
* Homeworks/projects must be turned in according to the instructions given in each homework/project assignment write-up. I.e.: after completing the homework/project, you must follow the instructions in the homework/project write-up and turn the work in.
* I have online office hours (at the Zoom link given on Canvas). Office hours are not used as lecture time where I reteach a topic that I have covered in some lecture. It is time for you to **ask specific question(s)** about lecture materials that you have question(s) on or explore the material in more depth. Outside office hours, you can ask me questions through email – I will usually respond the same day.
* Abide to the Emory Honor Code: <http://catalog.college.emory.edu/academic/policies-regulations/honor-code.html>

**Grading:**

**Grade scale:**

Letter Percentage

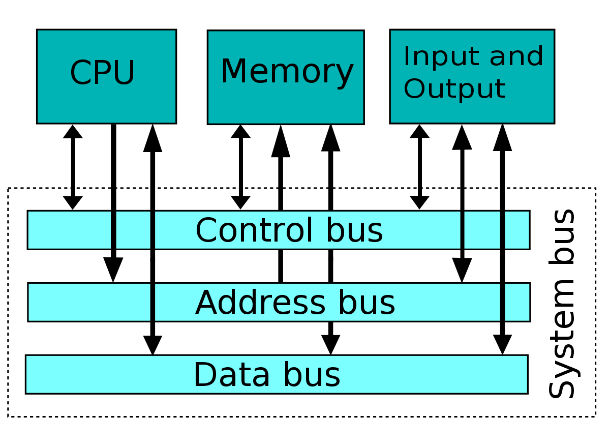
|  |  |
| --- | --- |
| A | 93.3 - 100 |
| A- | 90.0 – 93.2 |
| B+ | 86.6 – 89.9 |
| B | 83.3 – 86.5 |
| B- | 80.0 – 83.2 |
| C+ | 76.6 – 79.9 |
| C | 73.3 – 76.5 |
| C- | 70.0 – 73.2 |
| D+ | 65.0 – 69.9 |
| D | 60.0 – 64.9 |
| F | < 60.0 |

**Weight of assignments:**

* Homeworks and projects: 40%
* Midterm exam: 25%
* Final exam: 35%

**Missing exams:**

If you miss an (midterm or final) exam, you will **only** be allowed to do a "make up exam" with the permission the **Office of the Undergraduate Education** (i.e., the **dean**)



**Lecture format:**

|  |  |
| --- | --- |
| **How:** | **Synchronous online lectures using Zoom.**  **I will record each lecture and post the recording on Canvas – you can view the recording asynchronously later when available.** |
| **Zoom URL:** | **See Canvas for the URL** |
| **Netiquette of online Zoom lectures:** | **Mute** your microphone during the synchronous lecture, **except** when asking question. When you want to ask a question, unmute your microphone and then say: “Question” to interrupt the lecture. I will stop and let you ask your question. You must mute your microphone when you finish asking your question to minimize background noise. |

**Class Schedule (tentative)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Week** | **Lecture dates** | **Topics** | **What to read** | **To do assignment** |
| **1** | **1/26, 1/28** | **Intro to Computer Architecture + the Computer System** | **Chapters 1-3** | **Homework 1 (computer architecture)** |
| **2-3** | **2/2, 2/4, 2/9, 2/11** | **The programming process + storing different types of data in the computer** | **Chapters 4-6** | **Homework 2 (data representation in computer)** |
| **4** | **2/16, 2/18** | **Textual data and conversion between textual data and other types of data** | **Chapter 7** | **Project 3 (implement an ASCII data conversion algorithm)** |
| **5** | **2/23, 2/25** | **Representing computer instruction + Intro to (ARM) assembler programming** | **Chapter 8-9** |  |
| **6** | **3/2, 3/4** | **Defining program variables + working with simple program variables in assembler** | **Chapters 10-11** | **Project 4 (use simple variables in assembler program)** |
| **7** | **3/9, 3/11** | **Working with array and list data structures in assembler** | **Chapters 12-13** | **Project 5 (use structured variables in assembler program)** |
| **8** | **3/16, 3/18 (Delay assignment)** | **Working with if-statements and while-statements in assembler** | **Chapters 14-15** | **Project 6 (Write the GCD algorithm in assembler)** |
| **9** | **3/23, 3/25** | **Intro to subroutines (leaf) and passing parameters with registers** | **Chapters 16-17** | **Project 7 (Write the Bubblesort function in assembler)** |
| **10** | **3/30, 4/1** | **Passing parameters on stack + recursive subroutines** | **Chapters 17-18** | **Project 8 (Write a recursive function in assembler)** |
| **11** | **4/6, 4/8** | **Working with linked list using recursive functions + parameter passing methods (by reference)** | **Chapters 19-20** | **Project 9 (Write a recursive subroutine to insert into an ordered list in assembler)** |
| **12** | **4/13, 4/15** | **Intro to C programming, the C pre-processor, C data types, variable definitions, input/output, operators and statements** | **Chapters 21-25** |  |
| **13** | **4/20, 4/22** | **C data types, variable definitions, input/output, operators and statements, Arrays and function definitions/declaration and multi-files C programs** | **Chapters 22-30** | **Project 10 (C programming)** |
| **14** | **4/27, 4/29** | **C structs, scoping, pointers, pointer arithmetic and array access with pointer arithmetic** | **Chapters 31-32** |  |

