CS 136: Elementary Algorithm Design and Data Abstraction

Official calendar entry: This course builds on the techniques and patterns learned in CS 135 while making the transition to use of an imperative language. It introduces the design and analysis of algorithms, the management of information, and the programming mechanisms and methodologies required in implementations. Topics discussed include iterative and recursive sorting algorithms; lists, stacks, queues, trees, and their application; abstract data types and their implementations.

Course Staff – Overview

Instructors:

- Adrian Reetz
- Dave Tompkins

Other course personnel:

- ISAs (Instructional Support Assistants)
- IAs (Instructional Apprentices)
- ISC (Instructional Support Coordinator)

Announcements

All course announcements will be made on piazza.

Pinned piazza posts are *mandatory* reading, including official assignment and exam posts.

Main Topics & Themes

- imperative programming style
- elementary data structures & abstract data types
- modularization
- memory management & state
- introduction to algorithm design & efficiency
- designing “medium” sized, “real world” programs with I/O

Curriculum

Three of the most common programming paradigms are functional, imperative and object-oriented.

The first three CS courses at Waterloo use different paradigms to ensure you are “well rounded” for your upper year courses.

CS 135	$\Rightarrow$	CS 136	$\Rightarrow$	CS 246
functional		*imperative*		*object-oriented*

Each course incorporates a wide variety of CS topics and is **much more** than the paradigm taught.

Programming Languages

Most of this course is presented in the **C** programming language.

While time is spent learning some of the C syntax, this is not a “learn C” course.

We present C language features and syntax only as needed.

We occasionally use Racket to illustrate concepts and highlight the similarities (and differences) between the two languages.

What you learn in this course can be transferred to most languages.

CS 136: Remote Learning

	Traditional	**Remote Learning**
Content Delivery:	lectures	lecture notes (readings)
Content Enrichment:		online Q&A sessions supplemental videos
Demonstrations:	tutorials	tutorial videos
Self Assessment:	clickers	online quizzes
Applied Learning:	assignments	assignments
Formal Assessment:	exams	exams

Lecture Notes

These lecture notes will be the **primary source of information** and are available free online (at the course website).

It is your responsibility to thoroughly read these lecture notes.

There may be *(optional)* **supplemental videos** to explain some topics in more detail.

Several different styles of “boxes” are used in the lecture notes:

Important information appears in a thick box.

Comments and “asides” appear in a thinner box. Content that only appears in these “asides” will **not appear on exams**.

Additional **“advanced”** material appears in a “dashed” box.

The advanced material enhances your learning and may be discussed in class and appear on assignments, but you are **not responsible for this material on exams** unless your instructor explicitly states otherwise.

Q&A Sessions

Every week there will be scheduled Question & Answer (Q&A) sessions with the instructors to answer questions related to the content in these lecture notes.

Q&As are open to all students and can be attended by multiple (100+) students at a time.

Q&A sessions are held on Microsoft Teams.

Q&As are *not* for assignment-specific questions, which can be asked via piazza or during individual (one-on-one) office hours.

Even if you don't have a question, it might still be useful to participate in a Q&A:

- listening to your colleagues' questions allows you to compare your knowledge and assess your own progress
- listening to answers reinforces your knowledge

You must read through the scheduled lecture notes before participating in a Q&A.

Q&As are not a substitue for traditional lectures or for your own independent learning.

Tutorials

Tutorials showcase the process from a (simple) problem statement to a complete working solution.

We will post tutorial videos weekly to the course web site.

We strongly recommend following along with tutorials as they teach you how to approach, analyze, and solve programming problems. They help you getting into the “programmer’s mindset”.

Office Hours

Both ISAa and instructors will provide scheduled private individual (one-on-one) sessions every week.

Office hours are for all types of questions (*e.g.,* assignment and/or course content).

Office hours are held on Microsoft Teams.

Leave a message to get into the waiting queue; we will call you when it is your turn.

Schedule

Check the CS 136 web page and piazza regularly for the latest schedule.

In General, the planned weekly schedule is:

- **Monday:** New content available
- **Wednesday:** Tutorial available
- **Thursday:** Quiz Deadline
- **Friday:** Assignment released (due the following Friday)

Q&A sessions and office hours occur throughout the week.

Marking scheme

- 50% assignments (weekly)
- 15% quizzes (weekly)
- 10% midterm
- 25% final

Assignments

Assignments are *weekly* and each assignment is weighted equally (except A0).

- **read the assignment instructions carefully**
- **read the official piazza post frequently**
- **rules & requirements may change throughout the course**

A0 does not count toward your grade, **but must be completed** before you can receive any other assignment marks.

Assignment Collaboration

Assignment questions are individually colour-coded as either **Black** or **Gold** to indicate the level of collaboration permitted.

For **Black** questions, **moderate collaboration** is permitted.

For **Gold** questions, **no collaboration** is permitted.

Test cases and documentation are part of your assignment and follow the same collaboration rules.

Black Assignment Questions

For **Black** questions (moderate collaboration):

- you may discuss assignment *strategies* openly
- you may search the Internet for strategies or code examples
- you may discuss or show your code with an *individual*, but **not** with a larger group (piazza, facebook, *etc.*)
- You may show your code to other individuals to give or receive help, but **copying is never allowed** (electronic transfer, copying code from the screen, printouts, *etc.*)

Gold Assignment Questions

For Gold questions, **no collaboration** is permitted:

- **never share or discuss your code** with other students
- do not discuss assignment *strategies* with fellow students
- do not search the Internet for strategies or code examples

You may always discuss your code **with course staff**.

Piazza posts regarding Gold questions must be private (*post to: Instructors*).

Integrity

If you submit any work that is not your completely your own (*e.g.,* you receive help) you must cite (identify) the source of the assistance in an integrity statement. Even though moderate collaboration is allowed on **Black** questions, you must still cite any assistance or collaboration.

You do *not* have to cite any assistance you receive from course staff (including office hours, piazza posts, tutorials, lecture notes, *etc.*).

If you do not cite a source of assistance, you are presenting work of others as your own; this is called *plagiarism* and constitutes a violation of academic integrity.

Hand-marking

Assignment questions might be labeled as *hand-marked*. They may be evaluated for *style*:

- documentation and comments
- code readability
- white space and indentation
- identifiers (variable & function names)
- appropriate use of helper functions
- testing methodology

Hand-marking

The purpose of hand-marking is not to "punish" or "torture" you. It is formative feedback to improve your learning.

Unfortunately, we do not have the resources to hand-mark all assignment questions.

Well formatted and documented code is still expected, even if it is not hand-marked.

We will not provide assistance (office hours or piazza) if your code is poorly formatted or undocumented.

View your formative feedback on `MarkUs`.

Second Chances for Assignments

Assignment deadlines are strict, but some assignment questions may be granted a “second chance”.

- Second chances might be granted automatically, depending the quantity and quality of the submissions.
- Don’t ask in advance if a question will be granted a second chance; we won’t know.
- Second chances are (typically) due 48 hours after the original.
- Your grade is: $\max(\text{original}, \frac{\text{original}+\text{second}}{2})$
  (*i.e.,* there is no risk in submitting a second chance).

Assignment Implementation via `Seashell`

We use our own development environment called `Seashell`:

- browser-based for platform independence
- works with both `C` and `Racket`
- integrates with Marmoset, our submission & testing environment
- helps to facilitate your own testing

See the website and view Tutorial 01 for how to use `Seashell`.

Design recipe

In CS 135 you were encouraged to use the ***design recipe***, which included: contracts, purpose statements, examples, tests, templates, and data definitions.

The design recipe has two main goals:

- to help you **design** new functions from scratch, and
- to aid **communication** by providing **documentation**.

In this course, you should already be comfortable designing functions, so we focus on **communication** (through documentation).

Documentation

In this course, every function you write must have:

- a **purpose** statement, and
- a **contract** (including a **requires** section if necessary)

Unless otherwise stated, you are **not** required to provide templates, data definitions, or examples.

Later, we extend contracts to include *effects* and *time* (speed / efficiency).

Assignment Submission via Marmoset

Assignments are submitted to the Marmoset submission system:

`http://marmoset.student.cs.uwaterloo.ca/`

There are two types of Marmoset *tests*:

- **Public** (*basic / simple*) test results are available immediately and ensure your program is "runnable"
- **Private** (*comprehensive / correctness*) test results are available after the deadline and fully assess your code

Public tests do not thoroughly test your code.

Assignment Submission via Marmoset

- Marmoset uses the best result from all your submissions, and we encourage frequent submission and re-submission
- for questions that are *hand-marked*, we mark the submission with the highest score; if two submission have the same score, we mark the one that was submitted closest to the deadline
- when you submit your assignments, you can view public test results immediately in `Seashell`

To view your private test results, you must log into Marmoset after the deadline.

Quizzes

There will be weekly quizzes to encourage active learning and provide timely feedback that allows you to assess your understanding of the course content.

Quizzes are posted on LEARN and cover the weekly content from the lecture notes.

While all Quizzes are weighted equally, the format and grading scheme can vary from week to week. Make sure to read the instructions carefully.

Support – Course Content

If you are struggling with the course content (concepts and material from the lecture notes) you may:

1. participate in a Q&A
2. post a public question on piazza
3. ask a question during individual office hours
4. reference the textbook

Support – Assignments

If you are struggling with an assignment you should:

1. carefully re-read the assignment
2. read the official assignment post (FAQ)
3. **search** piazza to see if your question has already been asked
4. review the corresponding tutorial
5. post privately on piazza
6. get assistance during individual office hours

Support – Assignments (Black)

For **Black** questions, you may additionally:

- Get help from a fellow student
- Search the Internet
- post a public question on piazza (but do not post your code)

Support – Other

If you need help with something that is not related to the course content or an assignment (*e.g.,* an administrative issue):

- file a request (via the website)
- post on piazza
- discuss the issue during office hours
- write an email to an instructor and/or the course ISC

Piazza etiquette

- **read** the *official assignment post* before asking a question
- **search** to see if your question has already been asked
- **use** meaningful titles
- **ask** *clarification questions* for assignments (do not ask *leading questions* for **Gold** questions)
- **do not** discuss strategies for **Gold** questions
- **do not** post any of your assignment code *publicly*
- you can post your **commented** code *privately*, and an ISA or Instructor *may* provide some assistance

Course Learning Goals

At the end of each Section there are ***learning goals*** for the Section (in this Section, we present the learning goals for the entire course).

These learning goals clearly state what our expectations are.

Not all learning goals can be achieved just by listening to the lecture. Some goals require reading the text or using `Seashell` to complete the assignments.

Course Learning Goals

At the end of this course, you should be able to:

- produce well-designed, properly formatted, documented and tested programs of a moderate size (200 lines) that can use basic I/O
- use imperative paradigms (e.g., mutation, iteration) effectively
- explain and demonstrate the use of the C memory model, including the explicit allocation and deallocation of memory
- explain and demonstrate the principles of modularization and abstraction

- implement, use and compare elementary data structures (structures, arrays, lists and trees) and abstract data type collections (stacks, queues, sequences, sets, dictionaries)
- analyze the efficiency of an algorithm implementation