McGill

Introduction

Instructor: Jonathan Campbell
Winter 2025

Plan for today

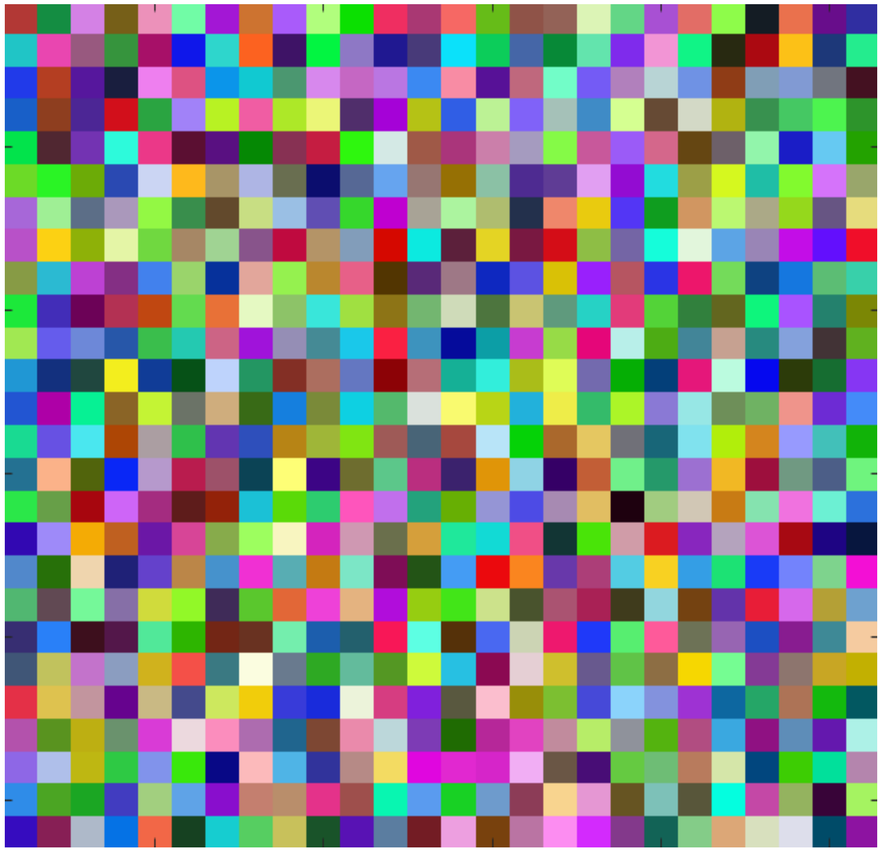
1. Introductions
2. Main course concepts
3. Course info + grading scheme

Who am I?

Jonathan Campbell

- Course lecturer / PhD student
- Research into AI/games
- Email: jonathan.campbell@mcgill.ca

Who are you??



Who are you??

B. Sci.....**93**

B. Arts.....**52**

Other.....**5**

150

Who are you??

U1.....**12**

U2.....**84**

U3+.....**54**

150

Who are you??

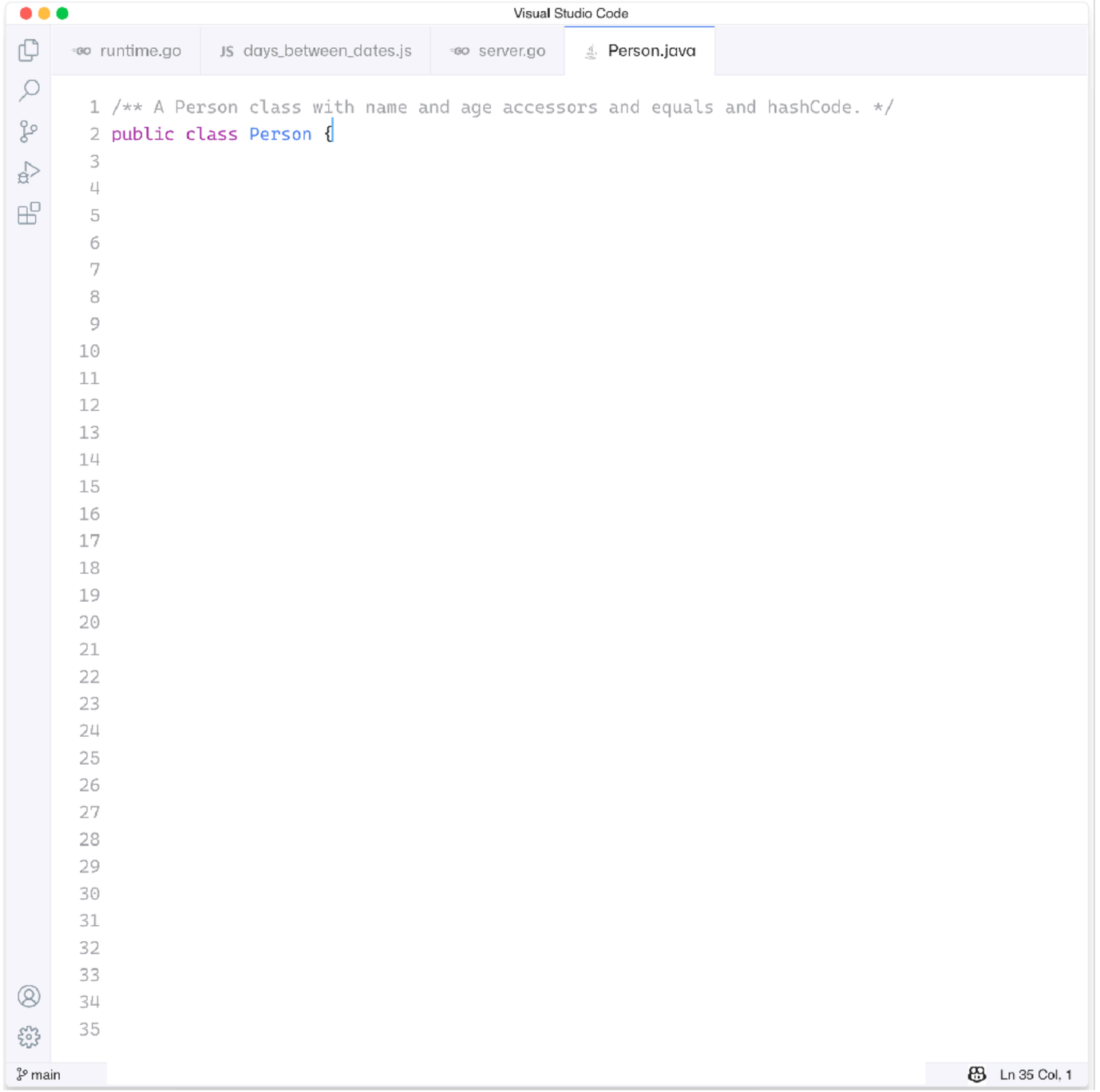
Past students...**43**

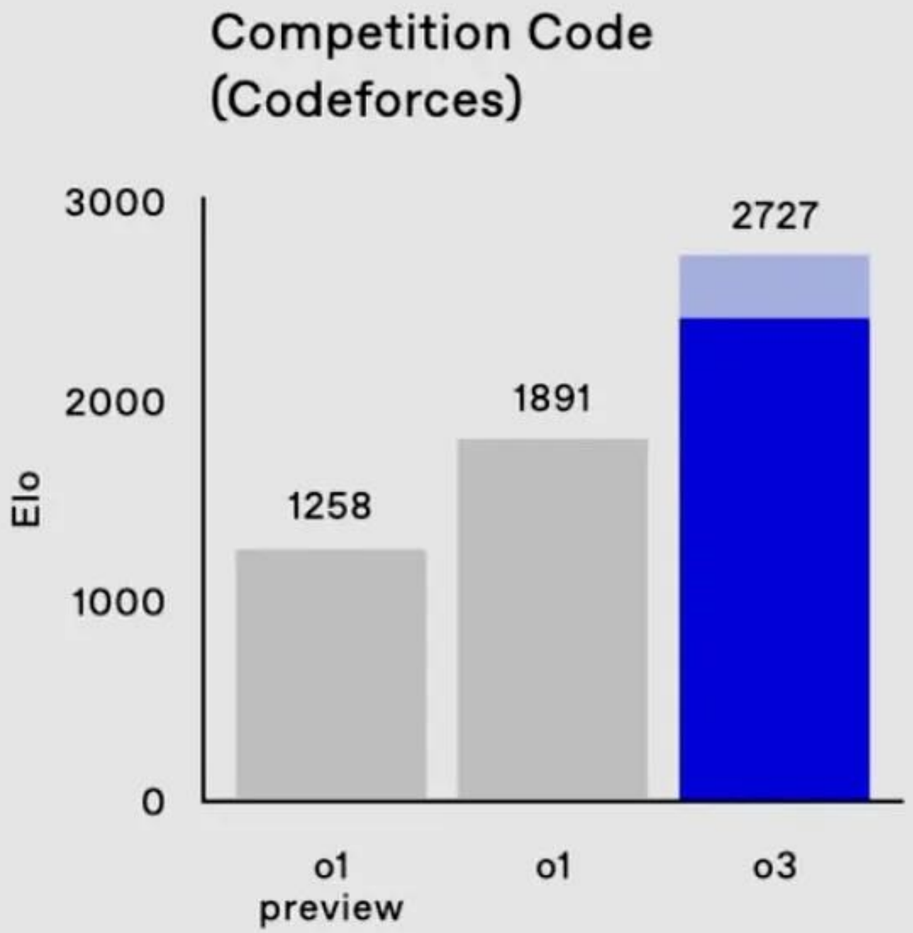
Summer '20.....1
Winter '21.....5
Fall '21.....5
Winter '22.....3
Fall '22.....27
Winter '24.....2

What is this course about?

What is this course about?

- How to make good software!


Visual Studio Code
runtime.go
days_between_dates.js
server.go
Person.java
/** A Person class with name and age accessors and equals and hashCode. */
public class Person {
main
Ln 35 Col. 1



An ELO of 2727 places O3 in the **top 200** competitive programmers in the world on CodeForces.

Viewpoint

The End of Programming

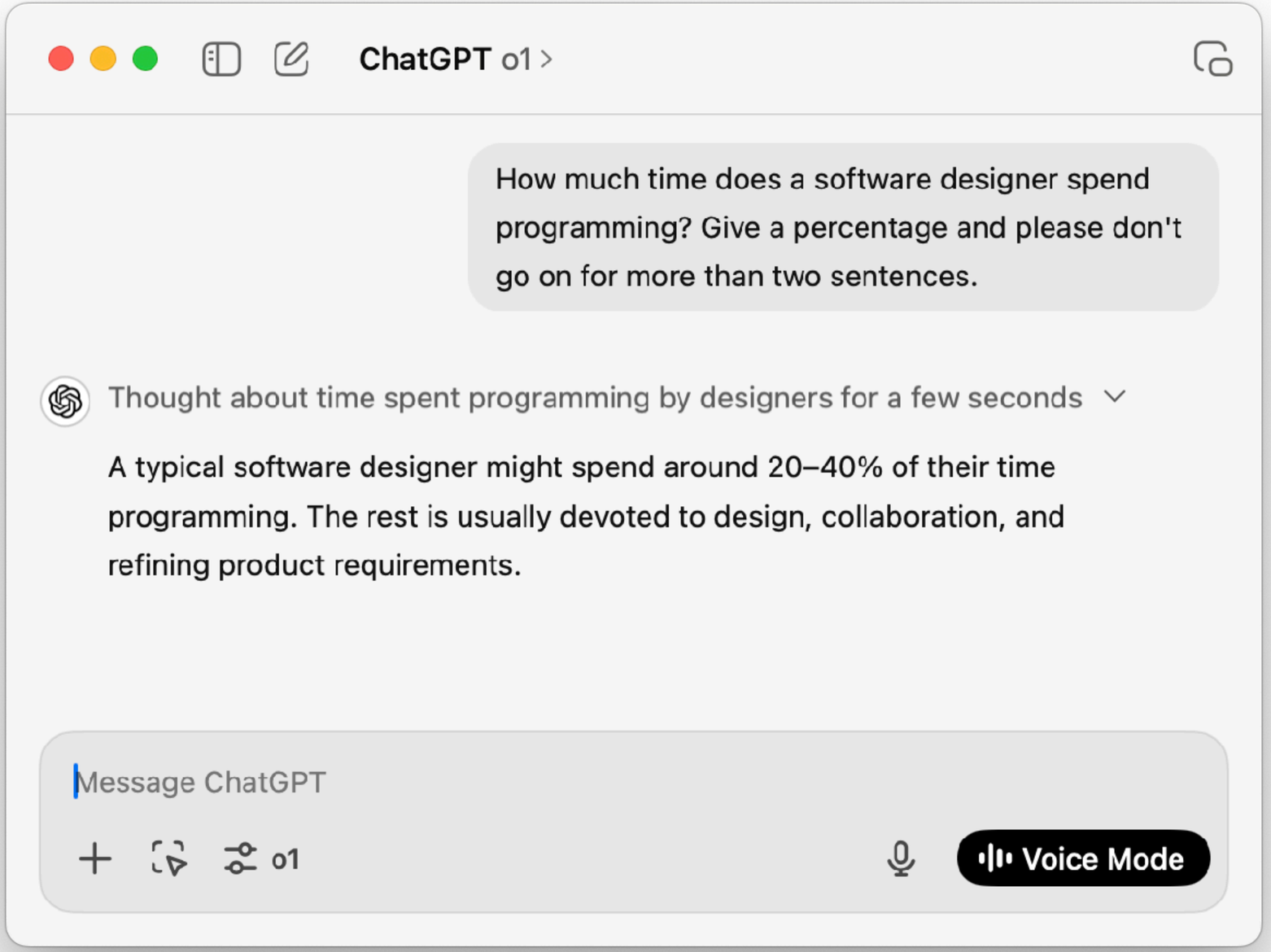
The end of classical computer science is coming, and most of us are dinosaurs waiting for the meteor to hit.

...I believe the computer scientists of the future will be so far removed from the classic definitions of “software” that they would be hard-pressed to reverse a linked list or implement Quicksort.

...Programming will be obsolete...

I am talking about *replacing the entire concept of writing programs with training models...*

https://dl.acm.org/doi/pdf/10.1145/3570220


ChatGPT o1 >
How much time does a software designer spend programming? Give a percentage and please don't go on for more than two sentences.
Thought about time spent programming by designers for a few seconds
A typical software designer might spend around 20–40% of their time programming. The rest is usually devoted to design, collaboration, and refining product requirements.
Message ChatGPT
o1
Voice Mode

Don’t quit your day job: Generative AI and the end of programming

Image Credit: VentureBeat made with Midjourney

...much of “**the rest of the job**” is understanding the user’s needs, designing, testing, debugging, reviewing code, finding out what the user really needs (that they didn’t tell you the first time), refining the design, building an effective user interface, auditing for security and so on.

https://venturebeat.com/ai/dont-quit-your-day-job-generative-ai-and-the-end-of-programming/

State of the Art of the Security of Code Generated by LLMs: A Systematic Literature Review

Leonardo Criollo Ramírez[1], Xavier Limón[2], Ángel J. Sánchez-García[3] and Juan Carlos Pérez-Arriaga[3].
Facultad de Estadística e Informática
Universidad Veracruzana
Xalapa, México

Our analysis revealed a range of vulnerabilities present in code generated by LLMs...

Notably, popular LLM-powered solutions like Github Copilot and ChatGPT exhibited a higher rate of vulnerabilities compared to code written by humans.

https://ieeexplore.ieee.org/abstract/document/10795572

Software design

Software design

the construction of abstractions of data and computation and the organization of these abstractions into a working software application

```
#include/**/<time.h>
#include <ncurses.h>
# include <stdlib.h>
/**                 */#define O()for(y-=                !!\
y;y<H&&           /*...Semi-Automatic.*/y<              p/W+2;\
y++)for(x=p%   W,x-=!!/*..              ...*/x;x<W&&   x<p%W+2;x++)
#define _(x,y)COLOR_##x,COLOR_##y /*  click / (R)estart / (Q)uit  */
#define Y(n)attrset(COLOR_PAIR(n)),mvprintw(/* IOCCC2019 or IOCCC2020 */
typedef int I;I*M,W,H,S,C,E,X,T,c,p,q,i,j,k;char G[]=" x",U[256];I F(I p){ I
r=0,x,y=p/W,q;O()q=y*W+x,r+=M[q]^=p-q?(M[q]&16)<<8:0;return r;}I K(I p
,I f,I g){ I x=(g+     f/256)%16-(f+g/256)%16,y=p/W,c=0,n=g/4096
,m=x==n?0:x==g            /16%16-f/16%16-n?256:-1; if(m+1)O()if
((4368&M[n=y*W             +x])==4112){ M[c=1,n]=(M[n]&~16)|m; }
return c;}void             D(){I p,k,o=0,n=C,m=0,q=0;if(LINES-1<H
||COLS/2<W)clear           (),Y(4)LINES/2,COLS/2-16,"Make the ter\
minal bigger!");else{for   (p=0;p<S;o+=k==3,Y(k)p/W+1,p%W*2,G),p++)G[1]=""
"_*!..12345678"[k=E?256&M[p     ]?n--,2:E-2||M[p]%2<1?M[p]&16?q=p,m++,3:4+F(p)%16:
1:3];k=T+time(0);T=o||T>=0||E-1?T:k;k=T<0?k:T;Y(7)0,0,"%03d%*s%03d",n>999?999:n,W*
2-6,"",k>999?999:k);Y(9)0,W-1,E>1?"X-(":E-1||o?":-)":"8-)");M[q]|=256*(n==m&&n); }
refresh();}short B[]={_(RED,BLACK),_(WHITE,BLUE),_(GREEN,RED),_(MAGENTA,YELLOW),_(
CYAN,RED)};I main(I A,char**V){MEVENT e;FILE*f;srand(time(0));initscr();for(start\
_color();X<12;X++){init_pair(X+1,B[X&&X<10?X-1:2],B[X?X<3?2:1:0]);}noecho();cbreak
();timeout(9);curs_set(0);keypad(stdscr,TRUE);for(mousemask(BUTTON1_CLICKED|BUTTO\
N1_RELEASED,0);;){S=A<2?f=0,W=COLS/2,H=LINES-1,C=W*H/5,0:fscanf(f=fopen(V[A-1],"r"
),"%d %d %d",&W,&H,&C)>3; ;S+=W*H;M=realloc(M,S*sizeof(I)*2);for(i=0
;i<S;i++)!f?M[i]=i,i&&(k=M[j=rand()%i],M[j]=M[i],M[i]=k):fscanf(f,
"%d",M+i);if(f)fclose(f);T=E=X=0;for(clear();D(),c=getch(),c-'r'
&&(c-KEY_RESIZE||E);){ if(c=='q'){ return(endwin(),0); }if(c==
KEY_MOUSE&&getmouse(&e)==OK&&e.x/2<W&&e.y<=H){if(!e.y&&(W-2<e.x&&
e.x<W+2)){break;}p=e.x/2+e.y*W-W;if(p>=0){if(!E){for(i=0;i<S;i++)M[S+M
[i]]=i,M[i]=16+(M[i]<C);C-=M[p]&1;M[p]=16;E=1;T=-time(0);}if(E<2)M[p]&=(M[p]
&257)==1?T+=time(0),E=2,273:257;}}for(p=0;p<S&&E==1;M[p++]&=273){}for(i=
(X+S-1)%S;E==1&&i!=X;X=(X+1)%S){if(!(M[p=M[X+S]]&272)){if(K(p,c=F(p)
,0)){goto N;}   for(k=p/W-2,k=k<0?0:k;k<p/W+3&&k   <H;k++)for(j=
p%W-2,j          =j<0?0:j;j<W&&j<p%W+3;)if          (!(M[q=
k*W               +j++]&272)){ if(K(p,              c,F
(q))){ goto N; }F(q)
; }F(p); }}N:; } } }
/*(c)Yusukse Endoh*/
```

http://www.ioccc.org/2020/endoh1/index.html

Goal of software design

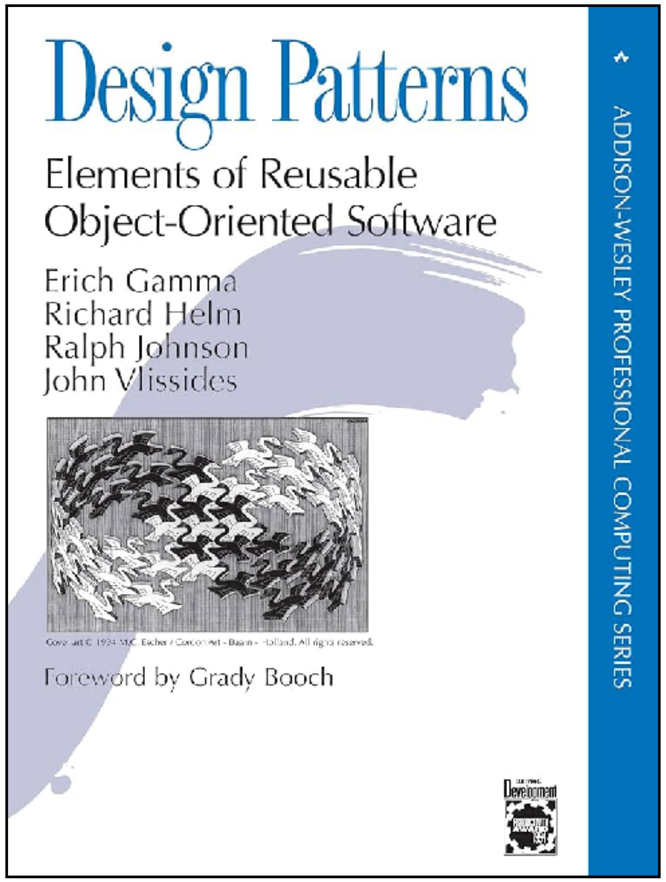
Writing code that is simple, easy to understand and modify.

What is software design?

- Deciding how to structure your code:
  - What data structure to use?
  - What methods should be defined in an interface?
  - Where should an error be handled?
- A heuristic process.

How to approach software design?

Design patterns



Design patterns

- A reusable solution to common, recurring problems.
- Makes a design more flexible.
- We will go over a subset of them.

Design anti-patterns

- Designs which lead to problems, e.g.:
  - Duplicated code
  - Long methods
  - "God" classes

Course info

Prerequisites

- You must have already taken COMP 250 and 206.

Prerequisites

- You should be able to:
  - understand and use basic data structures;
  - understand basic OOP concepts (objects, references, interfaces, inheritance, etc.);
  - write Java and Python programs to solve small problems;
  - use a version control system (e.g., Git); and
  - use the debugger!!!

Prerequisites

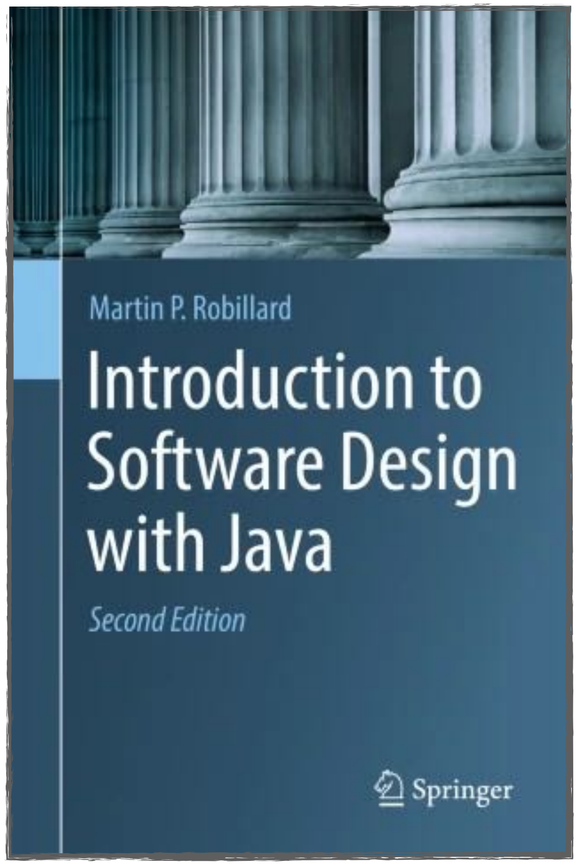
- A brief self-assessment, to check if you need to review any concepts, can be found here:
  - https://github.com/prmr/COMP303/blob/master/Assessment.md

Lectures

TR 2:35 - 3:55 p.m.

Slides and (best effort) recordings posted by end of week.

Textbook


Martin P. Robillard
Introduction to Software Design with Java
Second Edition
Springer

Textbook

- Available for free for McGill students at:
  - https://link.springer.com/book/10.1007/978-3-030-97899-0
- Companion website with exercises:
  - https://github.com/prmr/DesignBook
  - https://codesample.info/

Textbook

- Other textbooks:
  - The Pragmatic Programmer by Andrew Hunt and David Thomas
    - https://mcgill.on.worldcat.org/search/detail/1112609085?queryString=The Pragmatic Programmer by Andrew Hunt and David Thomas
  - Effective Java by Joshua Bloch
    - https://learning.oreilly.com/library/view/effective-java-3rd/9780134686097/

Syllabus

McGill | School of Computer Science

COMP 303

Software Design

Course Outline
Winter 2025

Instructor:
Jonathan Campbell
jonathan.campbell@mcgill.ca

Contents

Office hours

- Office hours will take place in-person and through Zoom and will start next week.
  - First come first serve, one-on-one.
- More information will be shared with you next week (schedule, rooms, Zoom links, etc.).
  - Times will be variable during the term (more during weeks where work is due).

TAs

Avinash

Issa

Divya

Keyu

Mentors

Angela

Laura

Daniel

Lauren

Class discussion: Ed

- We will be using Ed for class discussion.
  - You can access by clicking on "Discussions" on myCourses nav bar.
  - Make sure to set up your email/notification settings so that you are up to date with what is posted on Ed.
  - Here is a quick start guide to understand the features of Ed: https://edstem.org/quickstart/ed-discussion.pdf

Class discussion: Ed

- Please post **all questions about the course** on Ed.
  - Please check here for policy/etiquette:
    - https://www2.uwstout.edu/content/profdev/rubrics/discuss_etiquette.html
    - http://blogs.onlineeducation.touro.edu/15-rules-netiquette-online-discussion-boards/
    - Course syllabus
  - Do not post duplicate questions!
- You are encouraged to answer other students' questions.
  - OK to provide one or two lines of code to illustrate a point.
  - NOT ok to provide solution code.

Communication policy

- For personal or medical issues, you can email me at jonathan.campbell@mcgill.ca.
  - Make sure to put 'COMP 303' in subject and use McGill email.
  - If urgent, please write Urgent in the subject.
- For questions regarding a grade on an assessment, please see our policy in the course syllabus (section 4.5).
- For all other questions, please post on Ed, or see me or a TA in office hours and we will be happy to help you.

Assessment scheme

Old scheme from past terms

4 lab tests	25%
Midterm exam	25%
Final exam	50%

Old scheme from past terms

~~4 lab tests~~	~~25%~~
Midterm exam	25%
Final exam	50%

Old scheme from past terms

Midterm exam	25%
Final exam	**45%**

Our grading scheme

Midterm exam	25%
Final exam	45%
Project	**30%**

Our grading scheme

Midterm exam	25% or 0%
Final exam	45% or 70%
Project	**30%**

Project (30%)

- Two phases
  - Planning (10%) (January to mid February)
  - Implementing (20%) (mid February to April)

Project

- Planning
  - Forming teams
  - Brainstorming ideas: 1%
  - Project proposal: 5%
  - Reviewing other proposals: 4%

Project

- Implementing
  - Check-in meeting: 4%
  - Final demo + code submission: 16%

Project

- If working in team of 3:
  - Weekly reflection survey: 0.25% for each, 8 total (2%)
  - The 2% is added on top of your final project mark (capped at 30%).

Your to-do list

- Read the course syllabus (on myCourses).
- Read Chapter 1 of Intro to Software Design by Robillard.
- Try the self-assessment.
- Do exercise M-0.