

Today:
What's this course about???

Welcome to CS450!

High Level Languages

UMass Boston Computer Science
Instructor: Stephen Chang
Spring 2026

AN x64 PROCESSOR IS SCREAMING ALONG AT BILLIONS OF CYCLES PER SECOND TO RUN THE XNU KERNEL, WHICH IS FRANTICALLY WORKING THROUGH ALL THE POSIX-SPECIFIED ABSTRACTION TO CREATE THE DARWIN SYSTEM UNDERLYING OS X, WHICH IN TURN IS STRAINING ITSELF TO RUN FIREFOX AND ITS GECKO RENDERER, WHICH CREATES A FLASH OBJECT WHICH RENDERS DOZENS OF VIDEO FRAMES EVERY SECOND

BECAUSE I WANTED TO SEE A CAT JUMP INTO A BOX AND FALL OVER.



I AM A GOD.

This course is about learning to ...

- ... use high level languages effectively!
- ... implement your own high level language!

But ...
what's a **high level language??**

Welcome to CS450!

High Level Languages

UMass Boston Computer Science

Instructor: Stephen Chang

Spring 2026

What's this?

What's a Language?

- A language is for communication
 - With whom?



- A language is used to communicate to:

- Other people (in a conversation)
- To yourself (notes)
- Across time!

Dear Future Me,
By the time
you read this,
four years will
have passed...



To: The once and
future me
From: The me of
the past

From Wikipedia, the free encyclopedia

Language is a structured system of communication that consists of grammar and vocabulary. It is the primary means by which humans convey meaning, both in spoken and written forms,

Human language is characterized by its cultural and historical diversity, with significant variations observed between cultures and across time.



what is a language



A language is a structured system of communication that enables humans to convey information, thoughts, ideas, and emotions to one another. It is a complex and versatile tool that encompasses various components, such as words, grammar, syntax, semantics, and phonetics, which together allow for the creation and interpretation of meaningful messages.

This is a class about language

We will learn to use language to communicate (**read, write, and speak**) effectively

Welcome to CS 101

Programming High Level Languages

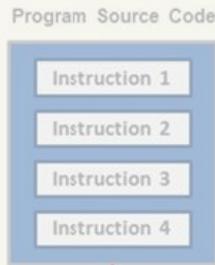
UMass Boston Computer Science
Instructor: Stephen Chang
Spring 2026

What's this?

What's a Programming Language?

- A way for **programmers** to **communicate** ...

- ... machine instructions (to a computer)
 - i.e., “programs”
- ... ideas (to another programmer)
 - e.g., code review,
 - pull requests
- ... ideas (to themselves)
 - **You** are the most frequent reader of your code!
- ... across time!



When you trying to understand
your 3 years old code



S what is a programming language

A programming language is a formalized system of communication that allows humans to instruct computers and perform various tasks. It serves as

Programs must be understandable by both computers and humans!

“Code is **read much more often than it is written**, so plan accordingly”
--- Raymond Chen

“The ratio of time spent **reading versus writing** is over **10 to 1**. We are constantly reading old code as part of the effort to write new code. ... [Therefore,] **making it easy to read makes it easier to write.**”

--- Robert C. Martin
Clean Code: Handbook of Agile Software Craftsmanship

Today:
What's this class about???

Welcome to CS450! **High Level Languages** Programming

UMass Boston Computer Science

Instructor: Stephen Chang

Spring 2026

This class is about learning to ...

- Use programming languages ... to **communicate effectively!**
 - To computers: via machine instructions
 - To humans (incl yourself): via reading, writing, speaking!

Today:
This class is about learning to ...

- Use programming languages ... to **communicate effectively!**
 - To computers: via machine instructions
 - To humans (incl yourself): via reading, writing, speaking!
- Write programs!

Today:
This class is about learning to ...

- Use programming languages ... to **communicate effectively!**
 - To computers: via machine instructions
 - To humans (incl yourself): via reading, writing, speaking!
- Write realistic programs!
 - ... that are clear and readable by humans!

What's this?

Welcome to CS450!

High Level Languages

Programming

UMass Boston Computer Science

Instructor: Stephen Chang

Spring 2026

CPU Language

```
0000000 0000 0001 0001 1010 0010 0001 0004 0128  
00000010 0000 0016 0000 0028 0000 0010 0000 0020  
00000020 0000 0001 0004 0000 0000 0000 0000 0000  
00000030 0000 0000 0000 0010 0000 0000 0000 0204  
00000040 0004 8384 0084 c7c8 00c8 4748 0048 e8e9  
00000050 00e9 6a69 0069 a8a9 00a9 2828 0028 fdfe  
00000060 00fc 1819 0019 9898 0098 d9d8 00d8 5857  
00000070 0057 7b7a 007a bab9 00b9 3a3c 003c 8888  
00000080 8888 8888 8888 8888 288e be88 8888 8888  
00000090 3b83 5788 8888 8888 7667 778e 8828 8888  
000000a0 d61f 7abd 8818 8888 467c 585f 8814 8188  
000000b0 8b06 e8f7 88aa 8888 8b3b 88f3 88bd e988  
000000c0 8a18 880c e841 c988 b328 6871 688e 958b  
000000d0 a948 5862 5884 7e81 3788 lab4 5ab4 3eeec  
000000e0 3d86 dcba 5cbb 8888 8888 8888 8888 8888  
000000f0 8888 8888 8888 8888 8888 8888 8888 8000  
0000100 0000 0000 0000 0000 0000 0000 0000 0000  
*  
0000130 0000 0000 0000 0000 0000 0000 0000 0000  
000013e
```



cpu

Machine code

“low level”

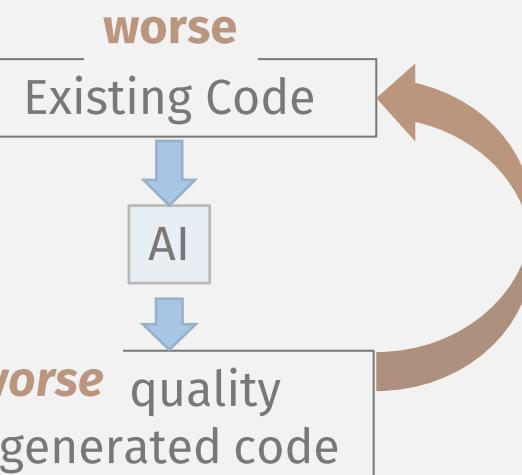
Programmers don't write machine code!

Because it's difficult for humans to understand

Humans need “higher level” languages!

Every programming language is created only to improve human communication

“high” level
(easier for humans
to understand)



“low” level
(runs on cpu)

English?

Q: Why don't we just
program in English?

A: It's too imprecise

In order to
use AI to program,
you first need to
know how to program!



What about AI???

S

write python to sort a list



Sure, you can sort a list in Python using the `sorted()` function or the `list.sort()` method. Here are examples of both methods:

1. Using the `sorted()` function:

```
python
original_list = [5, 2, 9, 1, 5, 6]
sorted_list = sorted(original_list)
print(sorted_list)
```

Copilot writes bad code 🔪

(impossible!
See cs 420!)

son is because of how language models work. They show how, on average, most people write. They don't have any sense of what's correct or good. Most code on GitHub is (by software standards) pretty old, and (on average) written by average programmers. Copilot spits out its best guess as to what those programmers might write if they were writing the

Universal Language Model Fine-tuning for Text Classification

Jeremy Howard*
fast.ai
University of San Francisco
j@fast.ai

Sebastian Ruder*
Insight Centre, NUI Galway
Aylion Ltd., Dublin
sebastian@ruder.io

“high” level
(easier for humans
to understand)



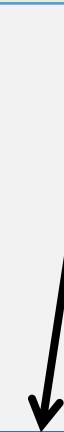
English

Q: Why don't we just
program in English?

A: It's too imprecise

Still needed in programs, for:
- Documentation
- Comments
- Specifications
(programs are more than code)

Code cannot be ambiguous



???

Machine code

“low” level
(runs on cpu)

“high” level
(easier for humans
to understand)

This is easier for humans
to understand, but what
about the computer?

“low” level
(runs on cpu)

// I = 15;
MOV R3, #15
STR R3, [R11, #-8]

// J = 25;
MOV R3, #25
STR R3, [R11, #-12]

// I = I + J;
LDR R2, [R11, #-8]
LDR R3, [R11, #-12]
ADD R3, R2, R3
STR R3, [R11, #-8]

ASSEMBLY LANGUAGE

Language Level:



Less performant “high” level
(easier for humans
to understand)

(usually)

Assembler

This is easier for humans
to understand, but what
about the computer?

It still runs machine code!

A higher-level language needs a
compiler (another program!) to
translate it to machine code

(Covered in another course!)

Tradeoff: This can
introduce inefficiencies

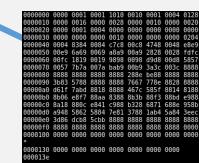
// I¹⁵;
MOV R3, #15
STR R3, [R11, #-8]

// J²⁵;
MOV R3, #25
STR R3, [R11, #-12]

// I¹ J¹;
LDR R2, [R11, #-8]
LDR R3, [R11, #-12]
ADD R3, R2, R3
STR R3, [R11, #-8]

ASSEMBLY LANGUAGE

More performant “low” level
(runs on cpu)



Assembly Language
Machine code

Named instructions

Less performant “high” level
(easier for humans
to understand)

(Covered in
other courses!)

Programs are
sequences of
statements or
“commands”

More performant “low” level
(runs on cpu)

“imperative”

JavaScript, Python	“eval”
C# / Java	GC (no alloc, ptrs)
C++	Classes, objects
C	Scoped vars, fns
Assembly Language	Named instructions
Machine code	

“dynamic” programs
(no pre-compiling)
Enables “interactive”
web apps, e.g., IDEs!

HUGE security
improvements
- No more “buffer
overflow” or “use
after free”

Less performant “high” level
(easier for humans
to understand)

“not imperative?”

Programs are
sequences of
statements or
“commands”

More performant “low” level
(runs on cpu)

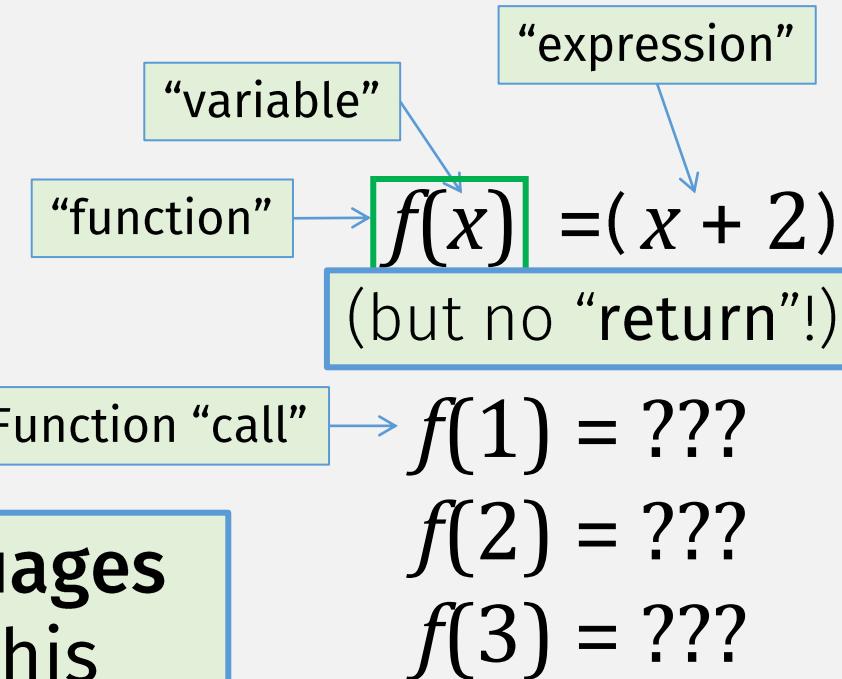
“imperative”

???	???
JavaScript, Python	“eval”
C# / Java	GC (no alloc, ptrs)
C++	Classes, objects
C	Scoped vars, fns
Assembly Language	Named instructions
Machine code	

Arithmetic

Functional languages
compute like this
(combining arithmetic
expressions)
(instead of sequences of statements)

(main topic in this course)



Is this programming?

A screenshot of Microsoft Excel showing a spreadsheet titled "Result Table". The table has columns A and B. Column A lists product names, and column B lists sales values.

A	B
Product Name	Sales
Alice Mutton	\$266,760
Boston Crab Meat	\$176,841
Camembert Pierrot	\$318,240
Ipoh Coffee	\$139,840
Hot Pepper Sauce	\$134,736
Spiced Okra	\$150,960
Giovanni	\$139,000

Cell B2 contains the formula `=SUM(B2:B11)`. The formula bar also shows `=SUM(B2:B11)`. To the right of the table, there is a "Result Table" section with three rows:

	Result Table
Grand Total	=SUM(B2:B11)
No Of Product	10
Average Sale	\$ 192,390.4

Below the table, the formula `-4 =SQRT(-4)` is shown with the error message `#NUM!`.

Is this a programming language?

YES!

This kind of programming is sometimes called "**declarative**"

"Declare" the computation you want.
It's "**high level**" because low-level details are not specified

“high” level
(easier for humans
to understand)

“declarative”

Declare computation
with **expressions**
(compiler decides low
level instructions)

Describe
computation with
exact sequence of
statements

“low” level
(runs on cpu)

“imperative”

Functional lang (Racket)	Expressions (no stmts)
JavaScript, Python	“eval”
C# / Java	GC (no alloc, ptrs)
C++	Classes, objects
C	Scoped vars, fns
Assembly Language	Named instructions
Machine code	

Lazy Arithmetic

Lazy (functional) **languages**
(also mathematical
languages like R) delay
computation until it's
needed

(may cover in this course)

$$f(x, y) = x + 2$$

$$f(1, 2 + 3) = ???$$

Result of this expression
is not needed,
so no need to compute it

“high” level
(easier for humans
to understand)

“declarative”

“imperative”

“low” level
(runs on cpu)

Lazy lang (Haskell, R)	Delayed computation
Functional lang (Racket)	Expressions (no stmts)
JavaScript, Python	“eval”
C# / Java	GC (no alloc, ptrs)
C++	Classes, objects
C	Scoped vars, fns
Assembly Language	Named instructions
Machine code	

Enables new kinds of
programs,
e.g., “tying the knot”

Logic Programming – Even Higher Level

Why does this have to
be the “input”?

$$f(x) = x + 2$$

$$\begin{aligned} f(??) &= 3 \\ f(??) &= 4 \end{aligned}$$

“relational”
programming

```
1 child_fact(oscar,karen,franz).
2 child_fact(mary,karen,franz).
3 child_fact(eva,anne,oscar).
4 child_fact(henry,anne,oscar).
5 child_fact(isolde,anne,oscar).
6 child_fact(clyde,mary,oscarb).
7
8 child(X,Z,Y) :- child_fact(X,Y,Z).
9 child(X,Z,Y) :- child_fact(X,Z,Y).
10
11 descendant(X,Y) :- child(X,Y,Z).
12 descendant(X,Y) :- child(X,U,V), descendant(U,Y).
```

(may cover in this
course)

Not code, but
programs need it for:
- Documentation
- Comments
- Specifications

Potential Problem:
not checked against code,
not guaranteed to match up

“native”

“imperative”

“low” level
(runs on cpu)

English	Types? pre/post cond? asserts
Specification langs	
Markup (html, markdown)	tags
Database (SQL)	queries
Logic Program (Prolog)	relations
Lazy lang (Haskell, R)	Delayed computation
Functional lang (Racket)	Expressions (no stmts)
JavaScript, Python	“eval”
C# / Java	GC (no alloc, ptrs)
C++	Classes, objects
C	Scoped vars, fns
Assembly Language	Named instructions
Machine code	

More “domain specific”

“high” level
(easier for humans
to understand)

“declarative”

Declarative languages
can have **imperative**
features, and vice versa

Can program
with expressions

Java Lambda Syntax	
Concise	<code>n -> System.out.print(n)</code>
Expanded	<code>(String n) -> System.out.print(n)</code>
Verbose	<code>(String n) -> { System.out.print(n); }</code>

“low” level
(runs on cpu)

English

Specification langs

Markup (html, markdown)

Database (SQL)

Logic Program (Prolog)

Lazy lang (Haskell, R)

Functional lang (Racket)

JavaScript, Python

C# / Java

C++

C

Assembly Language

Machine code

NOTE: This hierarchy is approximate

Types? pre/post cond? asserts

tags

queries

relations

Delayed computation

Expressions (no stmts)

“eval”

GC (no alloc, ptrs)

Classes, objects

Scoped vars, fns

Can program
with statements

```
> (define x 12)
> (set! x (+ 1 x))
> x
13
```

Goal: learn to use “high-level”
programming **concepts**, not a
specific programming language

This class is about learning to ...

- Use programming languages to ...
communicate effectively!
 - To computers: via machine instructions
 - To humans (incl yourself): via reading, writing, speaking!
i.e., write programs! (that are clear and readable by humans!)

This class is about learning to ...

- Use **high-level** programming language features to ... **communicate effectively!**
 - To computers: via machine instructions
 - To humans (incl yourself): via reading, writing, speaking!
i.e., write programs! (that are clear and readable by humans!)

This class is about learning to ...

Part 1

- Use high-level programming language features to ... communicate effectively!

• To computers: via machine instructions

• To humans (incl yourself): via reading, writing, speaking!

i.e., write programs! (that are clear and readable by humans!)

Redundant!

(Remember: high-level languages invented for human communication)

This class is about learning to ...

Part 1

- Use high-level programming language features to ... communicate effectively!

• To computers: via machine instructions

• To humans (incl yourself): via reading, writing, speaking!

i.e., write programs! (that are clear and readable by humans!)

Part 2

- Implement^a high-level programming language features

This class is about learning to ...

Part 1

• **Use** high-level programming language features to ...

~~communicate effectively!~~

~~• To computers: via machine instructions~~

~~• To humans (incl yourself): via reading, writing, speaking!~~

helps

use

i.e., **write** programs! (that are **clear** and **readable** by humans!)

Part 2

• **Implement** a high-level programming language

Course Logistics

All course info available on web site:
<https://www.cs.umb.edu/~stchang/cs450/s26>

Racket (main programming language for this course)

450 edition



- Primarily “Functional”

And Practice / Improve Your
Most Valuable Skill:

- Easy (syntax) to learn
 - (But different than you might be used to!)

Learning New Concepts!

1. Download at racket-lang.org/download
2. Install 450 Edition (**racket450**)
 - See hw0
 - Install and be ready to write code in next (Tuesday) lecture

My goals for the course

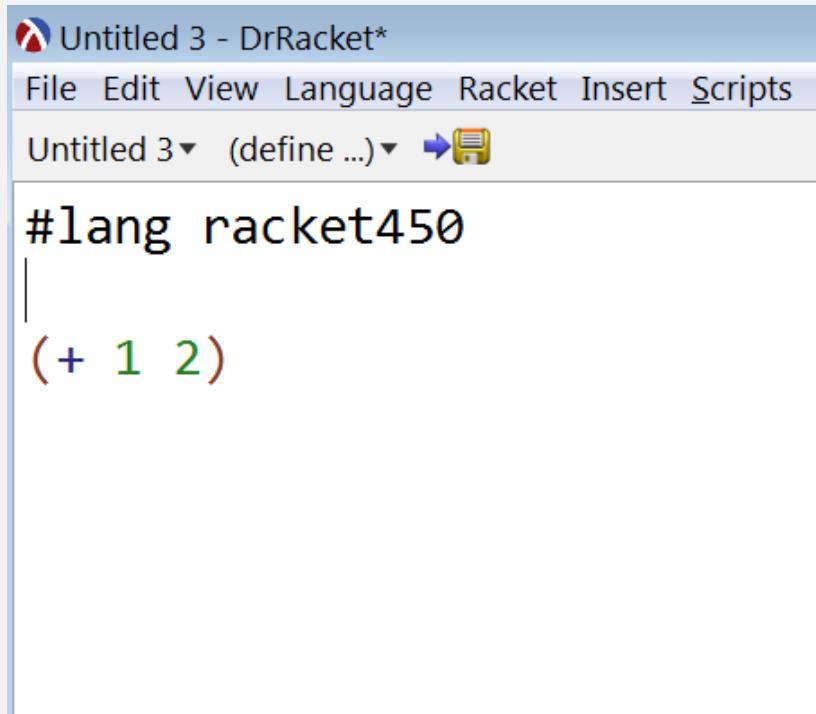
1. Teach students about high-level languages
2. Prepare students for post-UMB CS career

Installing “racket450”

The screenshot shows the DrRacket interface. The title bar says "hw13.rkt - DrRacket". A blue arrow points from the "File" menu to the "Package Manager..." option, which is highlighted. Below the menu, the "File" tab of the "Package Manager" window is active. The window has tabs for "Do What I Mean", "Currently Installed", "Available from Catalog", "Copy from Version", and "Settings". A search bar at the top says "Filter: racket450". The results table shows one entry: "racket450" by stchang@racket-lang.org, described as a programming language used in CS 450 at UMass Boston. The "Install" button in the bottom right corner of the table row is highlighted with a yellow box.

✓	Package	Author	Description	Ta...	Checksum	Source
✓	racket450	stchang@racket-lang.org	programming language used in CS 450 at UMass Boston		c5346437e0c4ee7eaa3b8581ae2fd79bcc6a8135	https://

Using “racket450”



The screenshot shows the DrRacket interface with the title bar "Untitled 3 - DrRacket*". The menu bar includes File, Edit, View, Language, Racket, Insert, and Scripts. The "Language" menu is currently selected. The main window displays the Racket code "#lang racket450" followed by "(+ 1 2)". The "define ..." dropdown menu is open, and the "racket450" option is highlighted.

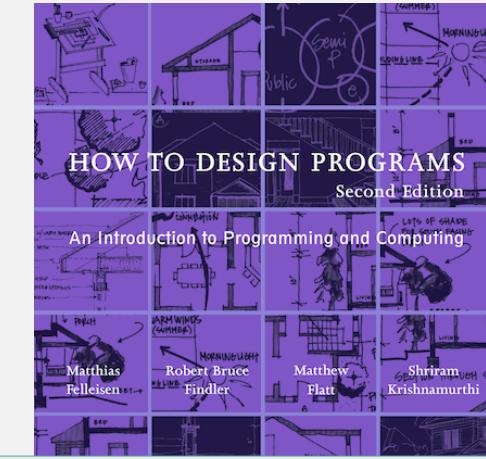
```
#lang racket450
(+ 1 2)
```

(textbook for this course)

How to Design Programs, 2nd ed.

Lessons:

- Programs are also for high-level communication
- This means that programs are more than what the code does
- Must be readable and explainable by others



Every org / company has rules for how to write clean, readable program

This is our rulebook!

Available free at: **htdp.org**

- Can buy paper copy (make sure it's 2nd ed) if you wish

All course info available on web site:

<https://www.cs.umb.edu/~stchang/cs450/s26>

GitHub

We will use GitHub for code management

1. Create an account (free) if you don't have one
2. Install a GitHub client and learn basic commands
3. Tell course staff your account name
 - (fill out pre-class survey if you have not done so!)

All course info available on web site:
<https://www.cs.umb.edu/~stchang/cs450/s26>

HW 0

- due: (next) Tuesday 2/3 11am
 - Create github account and learn basics
 - Tell course staff github account name (see hw0 details)
 - Install Racket and racket450
 - “Hello World”ish Racket450 programs
 - Be ready to program in class

Other Infrastructure

- Gradescope
 - Submitting HW and grading
- Piazza
 - Non-lecture communication

Grading

- **HW:** 80%
 - Weekly: in/out Tuesday (usually)
 - Approx. 12 assignments
 - Lowest grade dropped
 - **Participation:** 20%
 - In-class work, lecture, office hours, Piazza
 - No exams
- **A range:** 90-100
 - **B range:** 80-90
 - **C range:** 70-80
 - **D range:** 60-70
 - **F:** < 60

All course info available on web site:
<https://www.cs.umb.edu/~stchang/cs450/s26>

Grading

- **HW:** 80%
 - Weekly: in/out Tuesday (usually)
 - Approx. 12 assignments
 - Lowest grade dropped
- Evaluated on a program's:
 - **correctness**
 - i.e., test suites
 - **readability**
 - Can someone read and explain what it does?
 - **understanding**
 - Can you read and/or explain what it does?

All course info available on web site:
<https://www.cs.umb.edu/~stchang/cs450/s26>

Late HW

- Is bad ...
 - Grades get delayed
 - Can't discuss solutions
 - You fall behind!
- Late Policy: 3 late days to use during the semester

HW Collaboration Policy

Allowed

- Discussing HW with classmates (but must cite)
- Using other resources, e.g., youtube, other books, etc.
- Writing up answers on your own, from scratch, in your own words / code

Not Allowed

- Submitting someone else's answer
- It's still someone else's answer if:
 - variables are changed,
 - words are omitted,
 - or sentences rearranged ...
- Using sites like Chegg, CourseHero, Bartleby, Study, etc.
- Using AI bots like ChatGPT, Copilot, Claude, DeepSeek, etc.

This is an AI-free university course

Message from the provost:

"In this class, all submitted work must be original and created by the student(s) alone or in groups. Students should not have another person or entity write *any* portion of an assignment, including hiring others or using AI tools like ChatGPT. Always cite sources for quoted or referenced material. If unsure about a source's appropriateness, consult the instructor."

Honesty Policy

- 1st offense: zero on problem
- 2nd offense: zero on hw, reported to school
- 3rd offense+: F for course

Regret policy

- If you self-report an honesty violation, you'll only receive a zero on the problem and we move on.

All Up to Date Course Info

Survey, Schedule, Office Hours, HWs, ...

See course website:

<https://www.cs.umb.edu/~stchang/cs450/s26>