# Welcome to CS 100!

CS 100: Introduction to the Profession

Matthew Bauer & Michael Lee

# Agenda

- Syllabus & Administrivia
- What is CS? (What is it not?)
- Teaching computers

# § Syllabus & Administrivia

ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

# CS 100: Introduction to the Profession

## Announcements

- Welcome to the Fall 2023 edition of CS 100!

## Lectures

**Aug 25**

- Welcome; What is CS?
  - Lecture 01 Survey deadline midnight

## Assignments

Week 2 starting **Aug 28** (due midnight, the day before your week 3 lab)

- Lab 01 - Picobot
  - Simulator page
  - Submission form

Week 1 starting **Aug 21** (due by the end of week 1 lab)

- Lab 00 - Panopticon Escape Room
  - Game Codes
  - Leaderboard

## Administrivia

### Syllabus

- Course syllabus

### Instructors

- Matthew Bauer
  - Lectures Friday 1:50-3:05pm in Stuart Building 104 (auditorium)
  - Office: online Discord cs100
- Michael Lee
  - Lectures Friday 3:15-4:30pm in Stuart Building 104 (auditorium)
  - Office: SB 226C
  - Hours by appointment

Website: moss.cs.iit.edu/cs100

ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

# Instructors

**Matthew Bauer**
Questions? Please use cs100 channel on cs@iit on Discord

**Michael Lee**
Office: SB 226C
Hours by appointment
https://calendly.com/michaelee/officehours

ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

# Teaching Assistants/Mentors

L01 Jamal Jowdeh jjowdeh@hawk.iit.edu

L02 Christopher Mocha cmocha@hawk.iit.edu

L03 Ali Guzelyel aguzelyel@hawk.iit.edu

L04 Allysa Cao acao1@hawk.iit.edu

L05 Sovannratana Khek skhek@hawk.iit.edu

L13 Shambhawi Sharma ssharma29@hawk.iit.edu

L14 Tyler Keating tkeating1@hawk.iit.edu

L15 Amena Tajammul atajammul@hawk.iit.edu

L16 Zoe Guidroz zguidroz@hawk.iit.edu

L17 Van Anderson vanderson1@hawk.iit.edu

L19 Varvara Bondarenko vbondarenko@hawk.iit.edu

L20 Joshua Godwin jgodwin3@hawk.iit.edu

# Course Overview and Outcomes

An introduction to computer science as an academic pursuit and profession. Presents a broad survey of CS related topics and research areas, emphasizing problem-solving processes and their interdisciplinary nature.

Students will be able to:

- Analyze a complex computing problem and apply principles of computing and other relevant disciplines to identify solutions.
- Communicate effectively in a variety of professional contexts.
- Recognize professional responsibilities and make informed judgments in computing practice based on legal and ethical principles.
- Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline.

# Course Components/Grading

10%: Attendance
05%: Lecture Quizzes
15%: Labs
20%: P33 Weekly Deliverables (varies per company)
25%: P33 Final Deliverable (varies per company)
25%: P33 Final Presentation

Assignment grades will be updated in Blackboard periodically.
A>=90 B>=80 C>=70 D>=60 E<60

ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

# Attendance

- Lecture attendance is mandatory! Attendance taken by lecture surveys in weeks 1-4. Attendance taken by your TA in P33 lectures (weeks 5-13).
- Attendance for labs is mandatory! Attendance taken by your TA.
- Two absences are automatically excused. Each following absence reduces attendance score by 10%
- In case of illness or emergency, you must contact bauerm@iit.edu <u>before</u> the lecture or lab for an excused absence.

# Lecture Quizzes

- Online quizzes administered during lecture (must be present to take them, due at the end of lecture)
- First 4 weeks on the course website (login to IIT gmail before accessing)
- Today's Password: FSM

# Lab Assignments

- Small team activity/coding problem/etc. based on lecture topic assigned in lab weeks 1-5 and submitted online. Deadline for each lab is the midnight your lab day the following week
- Graded on 4-point scale
  - 0 (did not attempt) - 4 (well executed & meets all reqs.)
- Unexcused absence for lab = 0 for lab!

# P33 Project https://p33chicago.com/

- Work with local tech employers to solve real-life business problems.
- Small groups in lab with industry mentors to get a sense of "real" tech work.
- Goal: Acquire skills, experiences, and connections for future academic and practical work.
- From week 5 through 12 there will be weekly P33 team assignments and a final project deliverable and presentation due at the end of week 13.
- Teamwork is the key. You will get feedback on improving teamwork from your lab TA.
- Start Date: Friday, Sep 22 Presentation Date: Friday, Nov 17

ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

# CS100 - Introduction to the Profession (ITP)

- What does it mean to be a CS practitioner
  - Ethical and social concerns
  - Research / Industry Career Paths
  - Teamwork and Collaboration
- What is CS all about? What is it not?

ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

Is:

- software design
- algorithms
- theory of computing
- mathematical proofs

Isn't:

- building computers
- hardware focused
- a traditional "science"
- information technology

Computer science is no more about computers than astronomy is about telescopes.

*Anonymous*

# Not about computers?

- Sure: we use computers as *tools*
- But so do folks in nearly every other data/computation intensive fields!
  - Physics, Chemistry, Economics, Sociology, Music Production, etc.

# Science?

science |ˈsīəns|

noun

the intellectual and practical activity encompassing the **systematic study** of the structure and behavior **of the physical and natural world** through **observation and experiment**

*New Oxford American Dictionary*

ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

# Science?

- i.e., the scientific method
  - observe, hypothesize, experiment, analyze → refute/ validate hypothesis
- Yeah. We don't really do that.

Computer science is often defined as "the systematic study of algorithmic processes, their theory, design, analysis, implementation and application." An algorithm is a precise method usable by a computer for the solution of a problem.

*Encyclopedia.com*

# Ultimate Problem Solvers

- After a computer scientist comes up with the solution to a problem — *an algorithm* — a monkey can apply it!
  - A monkey with boundless patience, a perfect memory, and who can follow instructions to the letter
    - I.e., a computer

# Programs

- We codify solutions into *programs* which effectively teach computers how to solve our problems for us.
- And, ideally, reuse our code to build every grander programs!

ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

Programs have *billions* of moving pieces!

The Great Wall of China has nothing on an operating system kernel's codebase.

Nor does any ingenuous *mechanical device*.

Programming is certainly *not all we do*, but in order to efficiently carry out the solutions we invent, it's often a critical step!

§ Teaching computers

Question: what are some different ways in which we can program (teach) a computer to solve problems for us?

- Pre-existing software (typically application specific)
- Step-by-step instructions (*imperative* programming)
- Describing *what* we want done, but not *how* to do it (*declarative* programming)
- Building a system to *learn* how to solve the problem on its own (machine learning)

... and many more!

# Types of Programming Languages

- Imperative: *here's how to do it*
- Declarative: *here's what to do*
  - Logic: *deduce what I want*
  - Functional: *compute what I want*
- Domain-specific: tailored to the application

# Two Central Issues

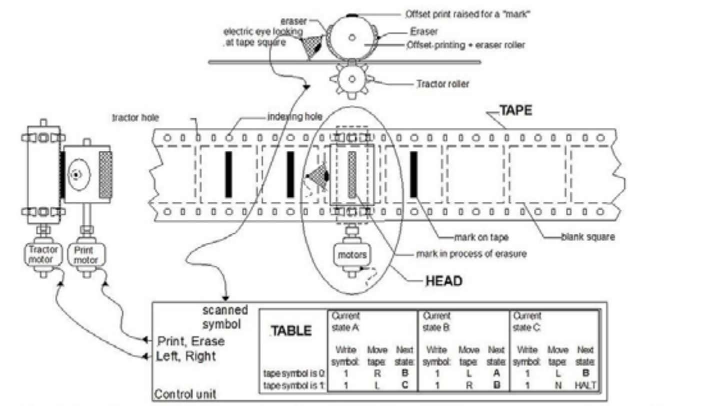
- Data representation: how do we describe the problem?
- Resource constraints: how much / what sort of computing power do we have available?

# E.g., Robotic Vacuum (Roomba)

- How to program a robot to vacuum a room thoroughly?
- Goal: maximize manufacturer profit (i.e., minimize cost of production), but still make a good robotic vacuum
- One solution: fast CPU, lots of memory, complex AI, full-room mapping — is this really necessary?
- What's the alternative?

# Computational Models

- We tend to reach for the most familiar — at this point, probably a general purpose CPU that can execute a "regular" computer program
  - A "Turing Machine"
- But other, possibly more efficient computing models exist

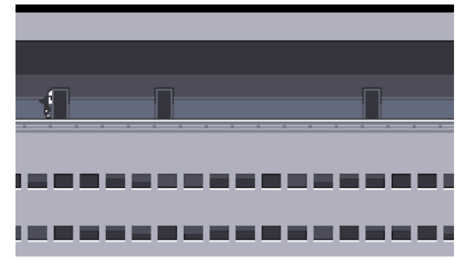


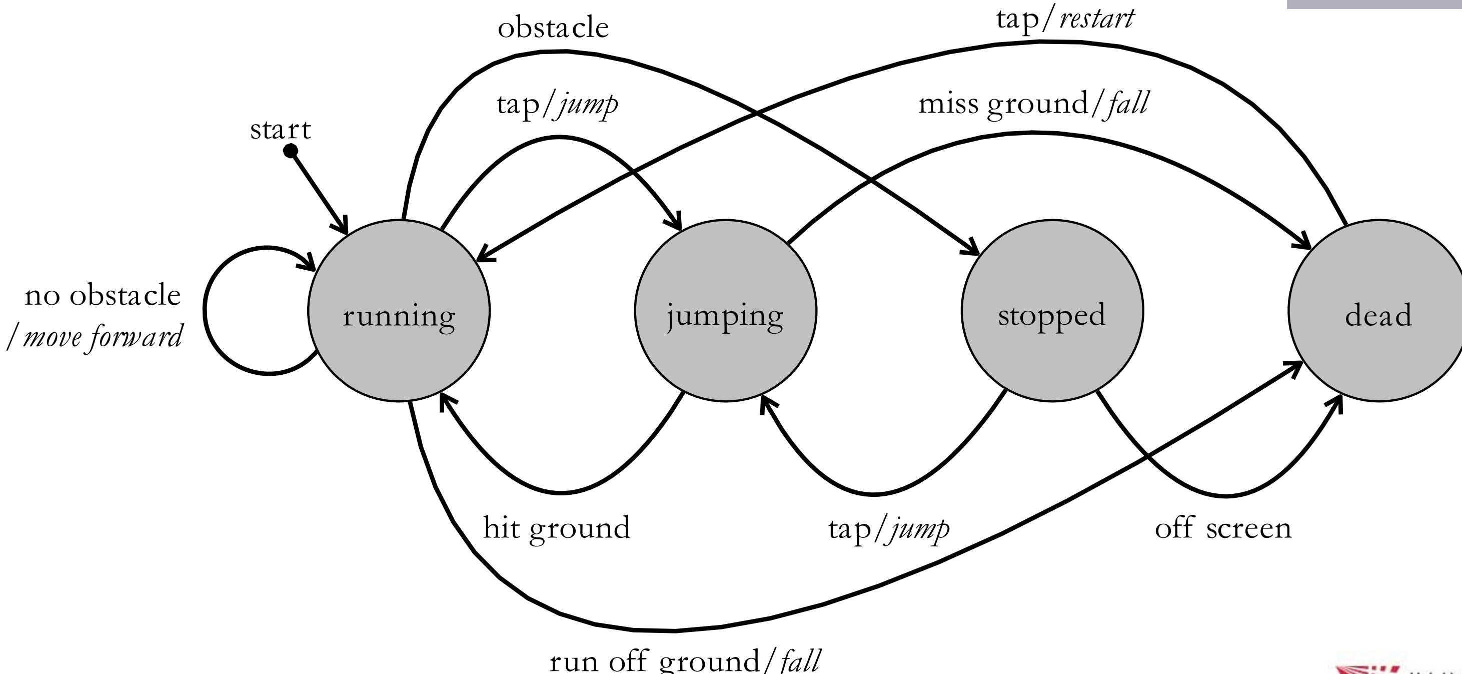

A fanciful mechanical Turing machine's TAPE and HEAD. The TABLE instructions might be on another "read only" tape, or perhaps on punch-cards. Usually a "finite state machine" is the model for the TABLE.

# Finite-State Machine

- Computational model for describing programmable logic
- Consists of *states*, *transitions* between states based on *inputs*, and possible *actions* (aka *outputs*) that occur on transitions
- We can use a *state-transition diagram* to describe a FSM

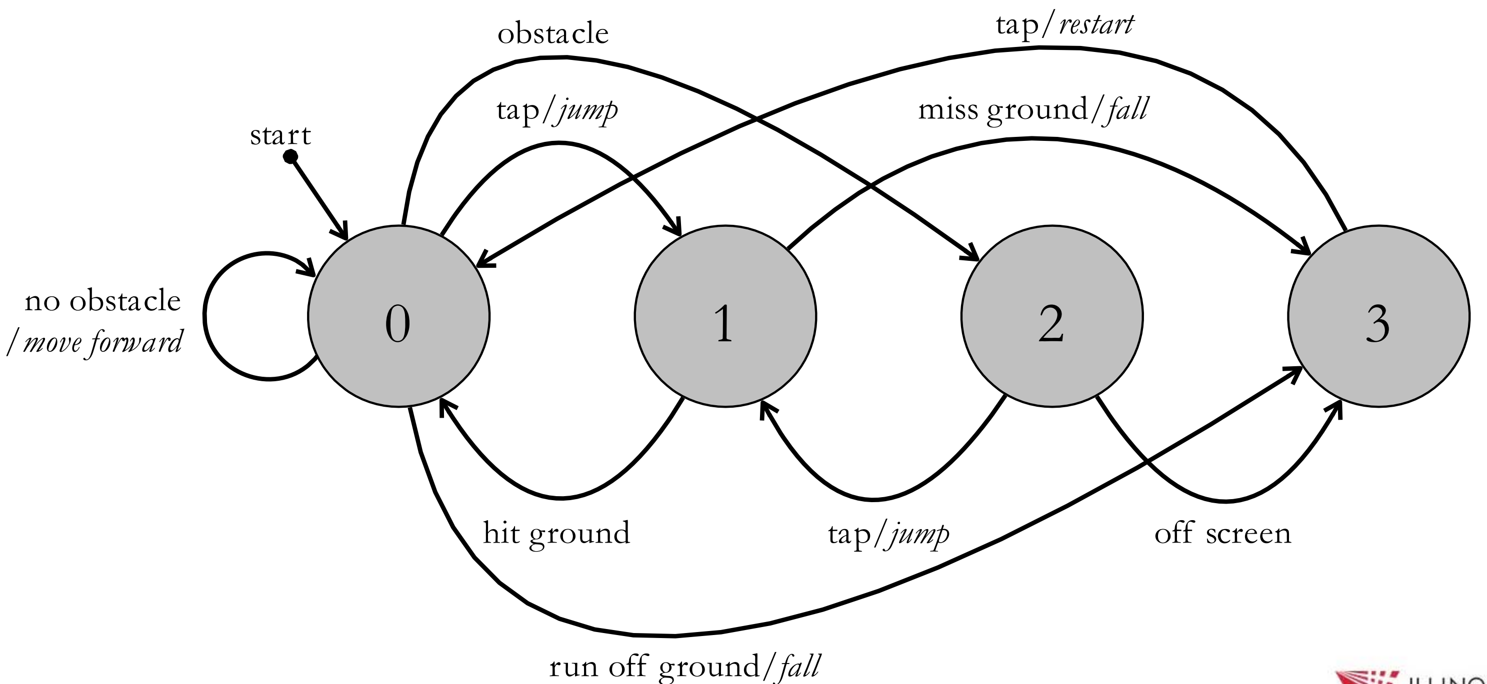
# Infinite Runner FSM






ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

# Infinite Runner FSM

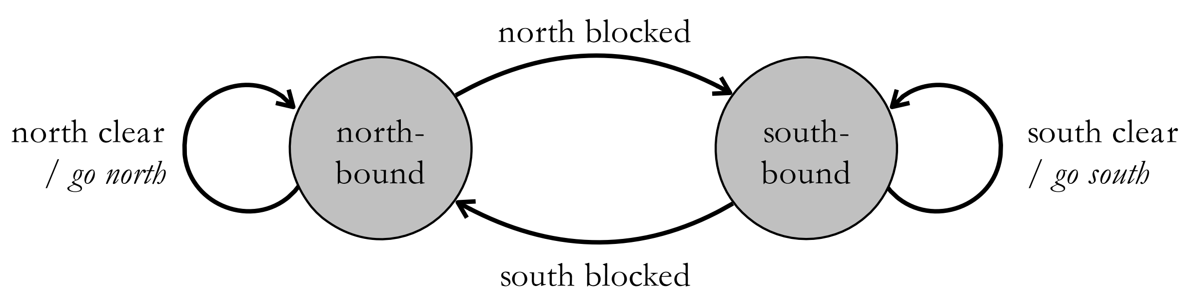


What inputs/actions might be needed for a robotic vacuum?

- inputs: collision sensors
- actions: *move* in direction; *suck* (perpetually — won't specify)

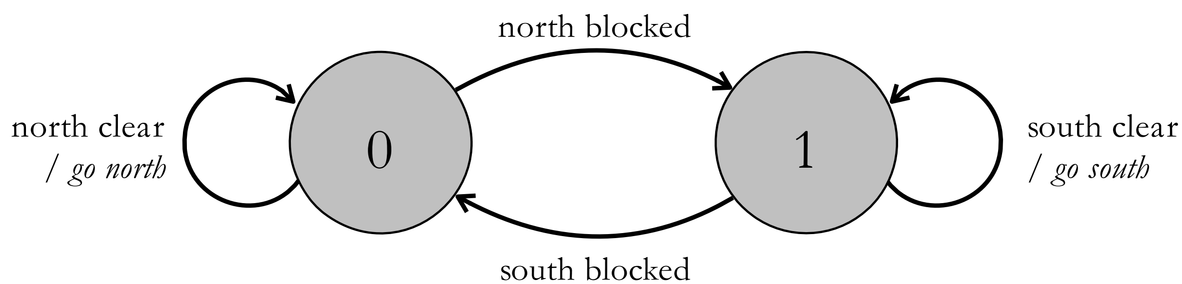

North
West
East
South

# Straight-line Robovac




ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

# Straight-line Robovac



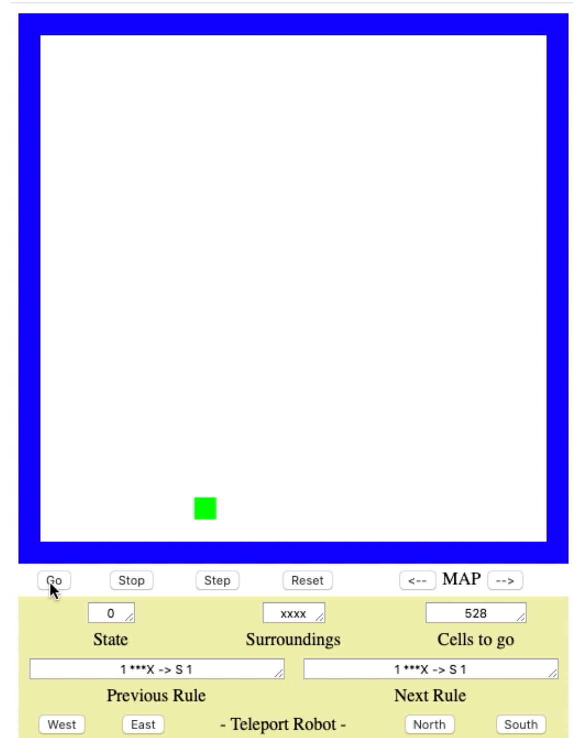
# Domain Specific Language

- Syntax: `STATE SURROUNDINGS -> ACTION NEXT_STATE`
- `STATE` / `NEXT_STATE` = `0`, `1`, `2`, …
- `SURROUNDINGS` = 4 letters for matching `N`, `E`, `W`, `S` sensor inputs — '`X`' for clear, `*` to ignore, direction letter for blocked
- `ACTION` = `N`, `E`, `W`, `S` for movement in direction, `X` for no move

ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

# Straight-line Robovac

```
0 x*** -> N 0  # head N if N is clear
0 N*** -> X 1  # N is blocked, switch state
1 ***x -> S 1  # head S if S is clear
1 ***S -> X 0  # S is blocked, switch state
```




ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

# Next Week's Lab: Picobot

- Write program(s) to make a simulated robovac navigate rooms with different kinds of obstacles
- Interesting question: is an FSM-based bot capable of fully covering any kind of room? (Arbitrary layout/obstacles)
  - CS meta-problem: *computability*

ILLINOIS INSTITUTE OF TECHNOLOGY
College of Science

# Lecture Quiz

- on the course website (login to IIT gmail before accessing)
- Today's Password: FSM