# HARNOOR DHILLON

Berkeley, CA · dhillon.h@berkeley.edu · (209) 924-9234

## EDUCATION

**The University of California, Berkeley** **Expected Graduation: Spring 2025**
***B.A. in Computer Science, B.A. in Data Science***
Relevant Coursework:

- CS61A (Structure/Interpretation of Computer Systems)
- CS61B (Data Structures/Algorithms)
- CS70 (Discrete Math/Probability Theory)
- Math 54 (Linear Algebra/Differential Equations)

## SKILLS & OTHER

- **Technical Stack:** Python, Java, Javascript, HTML/CSS, SQL, Scheme, C
- **Languages:** English, Spanish, Punjabi, Japanese
- **Hobbies/Interests:** Travel, Football, Skiing, Card games, Video games

## WORK EXPERIENCE

**GamesCrafters**
**Berkeley, CA**
***Undergraduate Researcher*** ***August 2022 – Present***

- Worked under Prof. Dan Garcia.
- Applied methods of Computational Game Theory to strongly solve strategy games.
- Used Python and C to develop autonomous opponents in abstract strategy games.
- Utilized Visual Value History to keep track of game trees.
- Created/improved GUI's (Graphical user Interface) for games.

**UC Berkeley Electrical Engineering & Computer Sciences Department**
**Berkeley, CA**
***Academic Intern* CS61A** ***January 2023 – Present***

- Worked under Prof. John Denero.
- Worked alongside other interns and course staff to teach CS fundamentals to a class of ~2000 students.
- Content: Higher-Order Functions, Recursion, Data Abstraction, Trees, Linked Lists, Object-Oriented Programming, and Efficiency.
- Languages Taught: Python, SQL, and Scheme.

## PROJECTS

**Ants Vs Bees Game**

- Using Python, I created a tower defense game based on Plants Vs Zombies where various types of ants are placed in lanes to protect against incoming bees.

**Scheme Interpreter**

- Created an interpreter for a small subset of the Scheme language using Python.
- Consisted of 3 parts: the Evaluator, Procedures, Special Forms.
    - Evaluator: Developed symbol evaluation, calling built-in functions, definitions
    - Procedures: Created Lambda (lambda…), Named (define (...) …), and Mu (mu…) procedures.
    - Special Forms: Implemented and, or, let, cond for the Scheme language.