COS 583: Great Moments in Computing (Spring 2025)
Last updated: January 18, 2025

Professor:
Professor Margaret Martonosi
martonosi@princeton.edu
Office hours: T/Th 4:30-5:30 (after class) in CS 208
Or use this signup link: https://calendar.app.google/9D9fpEMdkZCzzVDf8

TAs:
Kart Kandula
kkandula@princeton.edu
Office hours: Th 2-3pm (before class) in Computer Science Building 2nd floor atrium (just outside 208)
Or use this signup link: https://calendar.app.google/ULZJKtkU37VpScAD6

Tim Kosfeld
tk9066@princeton.edu
Office hours: Fridays 11am-noon in Friend 010A
Or use this signup link: https://calendar.app.google/q3t1yzRqxcX8YjTu9

Where to find stuff:
Course materials available on Canvas.
Q&A about materials and about course logistics on Ed (linked to from Canvas)
Course assignments via Gradescope (linked to from Canvas)

Since we will sometimes need to send time-critical information via Ed, *please set your Ed notification settings appropriately.* Namely, you probably want to have email notifications enabled. All questions (that are not of a personal nature) should be posted to Ed. If you email an Ed-appropriate question to the instructor or TAs, it will receive the response "Please repost to Ed, where both the question and the answer will reach its full audience. It is in everyone's interest that we maintain this policy; this is absolutely the most effective way to communicate."

When:
Tuesdays and Thursdays: 3-4:20. CS Small Auditorium Room 105

Course Description:
This course will cover pivotal developments in computing, including hardware, software, and theory. Material will be covered by reading seminal papers, patents and descriptions of highly-influential architectures. Emphasis will be on developing deep understandings of the discoveries and inventions that brought computer systems to where they are today. Discussion-oriented class will focus on in-depth analysis of readings. Final project or paper required.

Goals:

1. **Understanding Great Computing Ideas:** The central goal of this course is to give all participants an appreciation for the roots of particular seminal ideas and technologies, as well as an overall sense for how key ideas mature and evolve over time.

2. **Reading technical papers across a range of topic areas:** The class also gives participants the opportunity to hone the skill of reading and understanding papers deeply and efficiently.
3. **Effective interactive discussions:** Through the lecture discussions, students will also get practice in engaging in effective interactive technical conversations.
4. **Projects:** Independently develop an idea of your own in depth, through either a research or coding project of your own choosing.

A few notes:
The papers you will read are seminal but are not always simple or approachable, and they span many different areas of CS. You should expect to have difficulty understanding some, or even many, of them. You therefore will not be expected to master these readings, but rather to make your best effort. The in-class discussion is intended to help you to better understand the parts that may be difficult.

There have been more than great moments in computing than I can fit in one semester, so nobody is claiming completeness. I've left some classes open, to accommodate some amount of class choice/suggestions, so please let me know your preferences.

Course Grading Overview:
Participation in class discussions: 35%
Written responses to per-paper questions: 30%
Paper/project: 35%

Class participation and response papers
This course uses a discussion, not lecture, format. Each class will cover particular subjects from the assigned reading; particular issues for discussion will be posed in a handout available at least a week in advance (via Canvas/Gradescope).

Students will be expected to have carefully read the relevant assigned readings and to have prepared responses to, and analyses of, any assigned questions or topics. Some of these will require a brief written response; your written responses (1 or 2 pages) are due no later than the *beginning* of the class to which they pertain. 30% of the course grade will come from response papers, but I will discard the worst two. (Late written responses won't be accepted.)

The quality and quantity of student participation in class discussions is worth 35% of the course grade. Participation grades will reflect the *quality* of the student's preparation and analysis as well as the student's contribution to the process of discussion: making connections with other students' remarks, raising overlooked issues, asking good questions, making good summaries. Be aware: effective participation requires a great deal more *listening* than speaking, and in particular requires careful listening *to other students*, and not just to the instructors. The goal is to have a truly dynamic discussion, not a student-instructor ping-pong match.

Course Project Description:

Project Phases:

- Topic proposal, due March 18. Please submit via Gradescope one paragraph with your topic or plan
- You are required to schedule one checkpoint meeting or demo session with MRM between April 7 and April 25. Please use Calendly to sign up for a 15-minute slot:

https://calendly.com/martonosi/cos-ece-583-project-1-1-s?month=2025-04

- April 17: 10-minute Lightning talk in class.
- Final paper due (pdf) May 6 (Dean's Date). Programming projects may do a demo instead of a final writeup. To schedule your final demo, sign up for a 15-minute slot on Calendly (above).

Project Topics:
There are a few choices of course projects.
Choice 1: Citation Timeline Survey (Written): As you will see when we start reading and discussing papers, there are often some significant steps required between the original "Great Moment" paper and the subsequent uses of the idea that make its greatness clear. For example, Boole's book on logic (covered in lecture 2) needed considerable further steps before it came to be the Boolean logic we manipulate today.

So the purpose of this project is to pick one great moment paper that we cover in the class, and write up a paper (roughly 10-20 pages) covering some aspect of the "citation chain" either following from this paper to the present, or leading up to this paper beforehand. For example, you might show the citation chain from Boole's logic book towards Shannon's logic minimization work. Or you might show what has happened to any of the other class ideas from their original publication to the present. Or perhaps you could choose to follow time *backwards* from a great moment, in order to see what were the seminal building blocks that helped lead to it. In general, the idea is to show and discuss a direct citation-by-citation set of at least five steps. You can use scholar.google.com or other citation indices to track citations forwards and backwards. If you have ideas that are related to this, but don't precisely fit the 5-citation-link requirement, please do talk with me about them.

Choice 2: Programming: Some folks like to build real systems more than they like to write essays. So, here's another option. Select a programming project of your choosing, related to one of the papers we discuss. This could be: An emulator of an early computer architecture, a GUI representation of the Turing test (e.g. to use as teaching tool in an undergraduate course), etc etc. The Internet Archive includes a bunch of old but playable computer games in its collection. (https://archive.org) Also, Paul Allen's Living Computer Museum allows one to write and execute programs on very old hardware. (http://www.livingcomputermuseum.org) Please check with me about what you're planning.

Choice 3: Your Idea Here: If you have your own unique project ideas that are similar in effort and depth to the types described above, let me know and we can discuss the possibility of using them for the final project.

Written projects must be done individually. Programming projects can be done individually or in teams of 2-3. For team projects, your project proposal should describe the type and amount of work done per team member.

Policies

Collaboration Policy for Assignments and Projects:

In preparing pre-class assignments or other course work, students must reach their own understanding of the problem and discover a path to its solution. During this time, discussions with friends are encouraged. However, when the time comes to write the specific solution to a written assignment, such discussions are no longer appropriate -- the work you hand in must be your own work (although you may ask teaching assistants for help in debugging). The course's Ed page is also intended for sharing insights and clarifications on the meaning of questions, terminology, etc.

Do not, under any circumstances, copy another person's solution for an assignment. Likewise, do not share your solution with someone else in the class. Do not use solutions found on the web, whether at Princeton or elsewhere. Writing code or text for use by another person or using another person's code or text violates the University's academic regulations.

Examples of unacceptable behavior for assignments include:

- Copying any part of another person's program files or text into your own. The only exceptions are code or files explicitly approved in the assignment.
- Consulting or using assignment solutions from any previous offering of this course.
- Helping another student to do any of the above.

Ask the instructor if you have questions regarding this policy.

ChatGPT and Generative AI Tools Policy:
Unless otherwise specified, the use of generative AI is allowed or encouraged in the specific context of the course *coding*/projects, and with attribution: You can choose to use AI tools to help generate code or designs, or to revise existing *programming* work you have written. Since GAI tools are imperfect and can produce inaccuracies or hallucinations, all assignment submissions in the course – whether AI-assisted or not – are expected to include thorough descriptions of how you have tested the submitted design, and your rationale for its completeness.

For non-programming assignments, ie the pre-class written assignments, GAI tools are not allowed. The work should be your own.

If you have questions, please contact Prof. Martonosi.

Great Moments: Syllabus and Reading Assignments

Week	Dates	Tuesday	Thursday
1	Jan 28, 30	Class Overview	Foundations of Digital Logic [Boole, 1854] [Shannon, 1938]
2	Feb 4, 6	Artificial Intelligence [Turing, 1950] [Searle, 1980]	Roots of Machine Learning and Neural Networks [Rumelhart, 1986] [Valiant, 1984] [Hopfield, 1982]
3	Feb 11, 13	Ethernet [Metcalfe and Boggs, 1976]	Early Architectures [Burks et al. 1946] [Wilkes, 1965]
4	Feb 18, 20	Computer Vision [Lowe, 1999]	Virtual Memory [Kilburn, 1962] [Daley & Dennis 1968] [Anderson, 2014]
5	Feb 25, 27	Human-Computer Interaction [Sutherland, 1963]	Operating Systems [Ritchie & Thompson, 1974] [Engler et al. 1995]
6	Mar 4, 6	Data Abstraction [Liskov et al. 1977] [Liskov, 1987]	Invention of the Mouse [Engelbart, 1970]
Mar 11, 13		No Class: Spring break	

Week	Dates	Tuesday	Thursday
7	Mar 18, 20	Network Protocols [Cerf & Kahn, 1974]	Compilers [Hopper, 1952] [Backus et al. 1957]
8	Mar 25, 27	Moore's Law and its Future [Moore, 1965] [Moore, 2003]	Crypto and Encryption [Diffie & Hellman, 1976] [Rivest et al., 1978]
9	Apr 1, 3	The Rise of GPUs [Buck, 2004]	Quantum Computing [Feynman, 1982] [Shor, 1997]
10	Apr 8, 10	Privacy [Sweeney, 1997] [Dwork, 2006] [Narayanan and Shmatikov, 2008]	Distributed Systems [Lamport, 1978] [Lamport, 1998] [Castro & Liskov, 1999]
11	Apr 15, 17	Computer Music [Mathews, 1963] [Hiller & Isaacson, 1958]	Projects Lightning Round (10 minutes per person)
12	Apr 22, 24	Remaining Lightning Talks, (10 minutes per person) + Computer Games [Brand, 1972]	Back to the Future [Bush, 1945] [Weiser, 1999]

Reading List

[Amdahl, et al. 1964] G. M. Amdahl, G. A. Blaauw, and F. P. Brooks, Jr. Architecture of the IBM System/360. *IBM Journal of R & D*, vol. 8, no. 2 (April 1964), pp. 87-101.

[Anderson 2014] David Anderson. Tom Kilburn: a tale of five computers. Commun. ACM 57, 5 (May 2014), 35-38. DOI=10.1145/2594290 http://doi.acm.org/10.1145/2594290 .

[Backus, 1957] J. W. Backus, R. J. Beeber, S. Best, R. Goldberg, L. M. Haibt, H. L. Herrick, R. A. Nelson, D. Sayre, P. B. Sheridan, H. Stern, I. Ziller, R. A. Hughes, and R. Nutt. 1957. The FORTRAN automatic coding system. In Papers presented at the February 26-28, 1957, western joint computer conference: Techniques for reliability (IRE-AIEE-ACM '57 (Western)). 188-198.

[Boole, 1854] George Boole, *An investigation into the Laws of Thought, on Which are founded the Mathematical Theories of Logic and Probabilities*. 1854. Chapters 2 and 3.

[Buck, 2004] Ian Buck, Tim Foley, Daniel Horn, Jeremy Sugerman, Kayvon Fatahalian, Mike Houston, Pat Hanrahan. Brook for GPUs: stream computing on graphics hardware. ACM Transactions on Graphics (TOG), Volume 23, Issue 3. Pages 777 – 786 https://doi.org/10.1145/1015706.1015800

[Burks et al. 1946] Arthur W. Burks, Herman H. Goldstine, and John von Neumann, "Preliminary discussion of the Logical Design of an Electronic Computing Instrument," report to U.S. Army Ordnance Dept, 1946.

[Bush, 1945] Vannevar Bush, "As We May Think," *Atlantic Monthly*, July 1945. http://www.theatlantic.com/unbound/flashbks/computer/bushf.htm

[Brand, 1972] Stewart Brand. SPACEWAR: Fanatic Life and Symbolic Death Among the Computer Bums. Rolling Stone magazine. December 7, 1972.

[Castro & Liskov 1999] Miguel Castro and Barbara Liskov. 1999. Practical Byzantine fault tolerance. In Proceedings of the third symposium on Operating systems design and implementation (OSDI '99). USENIX Association, Berkeley, CA, USA, 173-186.

[Cerf & Kahn 1974] V.G. Cerf and R. E. Kahn, A Protocol for Packet Network Intercommunication. *IEEE Trans. Comms*. Vol. COM-22, No. 5, May 1974, pp. 637-648.

[Cook, 1971] S. Cook, The complexity of theorem-proving procedures. *Proc. Third Annual ACM Symposium on Theory of Computing*. 1971. pp. 151 - 158.

[Daley & Dennis 1968] Robert C. Daley and Jack B. Dennis. 1968. Virtual memory, processes, and sharing in MULTICS. Commun. ACM 11, 5 (May 1968), 306-312. DOI=10.1145/363095.363139 http://doi.acm.org/10.1145/363095.363139

[Dean & Ghemawat, 2004] Dean, J. and Ghemawat, S. 2004. MapReduce: Simplified data processing on large clusters. In Proceedings of Operating Systems Design and Implementation.

[Diffie & Hellman 1976] W. Diffie and M.E. Hellman, New Directions in Cryptography, *IEEE Trans. on Information Theory*. Vol. IT-22, No. 6. Nov. 1976, pp. 644-654.

[Dwork, 2006] Dwork, C. (2006). Differential Privacy. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds) Automata, Languages and Programming. ICALP 2006. Lecture Notes in Computer Science, vol 4052. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11787006_1

[Engelbart, 1970] Engelbart, D. X-Y Position Indicator for a Display System. US Patent #3,541,541, filed June 1967, issued Nov., 1970.

[Engler et al. 1995] DR Engler, MF Kaashoek, J O'Toole Jr Exokernel: An operating system architecture for application-level resource management. ACM SIGOPS Operating Systems Review, 1995 https://pdos.csail.mit.edu/6.828/2016/readings/engler95exokernel.pdf

[Feynman, 1982] Feynman, R.P. Simulating physics with computers. Int J Theor Phys 21, 467–488 (1982). https://doi.org/10.1007/BF02650179

[Hiller & Isaacson, 1958] L. A. Hiller, L. M. Isaacson. Musical Composition with a High-Speed Digital Computer. J. Audio Eng. Soc, Vol. 6, No. 3. (1958), pp. 154-160.

[Hopfield, 1982] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. PNAS, 1982. https://www.pnas.org/doi/10.1073/pnas.79.8.2554

[Hopper, 1952] Grace Murray Hopper. 1952. The education of a computer. In Proceedings of the 1952 ACM national meeting (Pittsburgh) (ACM '52). ACM, New York, NY, USA, 243-249.

[Karp, 1972] Karp, R.M. Reducibility among combinatorial problems. R.E Miller and J.W. Thatcher, editors, *Complexity of Computer Computations*, 85-103, Plenum Press, NY, 1972.

[Kay, 1972] Alan Kay. A Personal Computer for Children of All Ages. www.mprove.de/diplom/gui/kay72.html

[Kilburn 1962] T. Kilburn, D. B. G. Edwards, M. J. Lanigan, F. H. Sumner. 1962. One-Level Storage System. IRE Transactions on Electronic Computers. April, 1962.

[Kilburn 1961] T. Kilburn, R. B. Payne, and D. J. Howarth. 1961. The Atlas supervisor. In Proceedings of the December 12-14, 1961, eastern joint computer conference: computers - key to total systems control (AFIPS '61 (Eastern)). ACM, New York, NY, USA, 279-294. DOI=10.1145/1460764.1460786 http://doi.acm.org/10.1145/1460764.1460786

[Lamport 1978] Leslie Lamport. 1978. Time, clocks, and the ordering of events in a distributed system. Commun. ACM 21, 7 (July 1978), 558-565.

[Lamport 1979] Lamport, L., "How to Make a Multiprocessor Computer That Correctly Executes Multiprocess Programs," *Computers, IEEE Transactions on* , vol.C-28, no.9, pp.690-691, Sept. 1979

[Lamport 1998] Leslie Lamport. 1998. The part-time parliament. ACM Trans. Comput. Syst. 16, 2 (May 1998), 133-169.

[Liskov et al. 1977] B. Liskov, A. Snyder, R. Atkinson, C. Schaffert. Abstraction Mechanisms in CLU. Communications of the ACM. August, 1977. Volume 20, Number 8.

[Liskov, 1987] B. Liskov. Data Abstraction and Hierarchy. OOPSLA 1987 keynote.

[Lowe, 1999] D. G. Lowe, "Object recognition from local scale-invariant features," Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, 1999, pp. 1150-1157 vol.2, doi: 10.1109/ICCV.1999.790410.

[Mathews 1963] M. V. Mathews. The Digital Computer as a Musical Instrument. Science, New Series, Vol. 142, No. 3592 (Nov. 1, 1963), pp. 553-557

[Metcalfe and Boggs, 1976] R.M. Metcalfe and D. R. Boggs, Ethernet: Distributed packet switching for local computer networks. *Comm. ACM*. Vol. 19, Issue 7 (July 1976). pp: 395 – 404.

[Moore, 1965] Gordon E. Moore. Cramming more components onto integrated circuits. Electronics, 38(8) pp. 114 - 117, April 1965.

[Moore, 2003] Gordon E. Moore. No exponential is forever: but "Forever" can be delayed! ISSCC Keynote Address. 2003.

[Narayanan and Shmatikov, 2008] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets (how to break anonymity of the Netflix prize dataset). In Proceedings of IEEE Symposium on Security and Privacy. 2008.

[Page et al., 1998] Page, L., Brin, S, Motwani, R., and Winograd, T. The PageRank Citation Ranking: Bringing Order to the Web, Unpublished draft (!), 1998.

[Patterson et al., 1988] David A. Patterson, Garth Gibson, and Randy H. Katz. 1988. A case for redundant arrays of inexpensive disks (RAID). In Proceedings of the 1988 ACM SIGMOD international conference on Management of data (SIGMOD '88), Haran Boral and Per-Ake Larson (Eds.).

[Ritchie and Thompson, 1974], B. Ritchie and K. Thompson. The UNIX Time-Sharing System. *Communications of the ACM*. Vol. 17, Issue 7 (July 1974). Pp. 365 – 375.

[Rivest et al., 1978] R. L. Rivest, A. Shamir and L. Adleman, A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*. Vol. 21 , Issue 2 (February 1978) pp. 120 – 126.

[Rumelhart, 1986] Rumelhart, Hinton, Williams. Learning representations by back-propagating errors. Nature. Vol. 323 9 October, 1986.

[Searle, 1980] John Searle. Minds, Brains, and Programs. *Behavioral and Brain Sciences*, 1980

[Shannon, 1938] Claude Shannon. ``A Symbolic Analysis of Relay and Switching Circuits,'' *Transactions American Institute of Electrical Engineers*, Vol. 57 (1938), pp. 713-723. Part B.

[Shor 1997] Peter W. Shor. 1997. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM J. Comput. 26, 5 (October 1997), 1484-1509.

[Sutherland, 1963] I. E. Sutherland, "Sketchpad---A man-machine graphical communication system," *Proc. Spring Joint Computer Conference (AFIPS)*, 1963, pp. 328-346.

[Sweeney, 1997] L. Sweeney. Weaving technology and policy together to maintain confidentiality. Journal of Law, Medicines Ethics, 25:98–110, 1997.

[Tomasulo, 1967] R. Tomasulo. An Efficient Algorithm for Exploiting Multiple Arithmetic Units. *IBM Journal of R&D*, 1967

[Turing, 1936] A.M. Turing, On Computable Numbers, with an application to the Entscheidungsproblem, *Proc. Lond. Math. Soc*. (2) 42 pp 230-265 (1936-7); correction ibid. 43, pp 544-546 (1937).

[Turing, 1950] A.M. Turing, Computing Machinery and Intelligence, *Mind 49*, pp. 433-460 (1950)

[Valiant, 1984] L. G. Valiant. 1984. A theory of the learnable. Commun. ACM 27, 11 (November 1984), 1134-1142. DOI=10.1145/1968.1972 http://doi.acm.org/10.1145/1968.1972

[Wilkes, 1965] M. V. Wilkes, Slave Memories and Dynamic Storage Allocation, *IEEE Trans.*, vol. EC-14, no. 2, pp. 270-271, 1965.

[Weiser 1999] Mark Weiser. 1999. The computer for the 21st century. ACM SIGMOBILE Mobile Computing and Communications Review, Volume 3, Issue 3. https://doi.org/10.1145/329124.329126

Other Papers (Additional interesting papers, backup references, ones we have covered in past semesters)

[Appel, 2012] The Birth of Computer Science at Princeton in the 1930s, by Andrew W. Appel. Introduction to Alan Turing's Systems of Logic: The Princeton Thesis, Princeton University Press, 2012. https://assets.press.princeton.edu/chapters/s9780.pdf

[Brand, 1972] Stewart Brand. SPACEWAR: Fanatic Life and Symbolic Death Among the Computer Bums. Rolling Stone magazine. December 7, 1972.

[Faggin et al., 1996] Faggin, F., Hoff, M.E., Mazor, S., and Shima, M. The history of the 4004. *IEEE Micro*, Vol 16, Issue 6, Dec. 1996, pp. 10-20.

[Graetz, 1981] J. Martin Graetz. The Origin of Spacewar. Creative Computing. 1981. [Karp, 1972] Karp, R.M. Reducibility among combinatorial problems. R.E Miller and J.W. Thatcher, editors, *Complexity of Computer Computations*, 85-103, Plenum Press, NY, 1972.

[Kilby, 1964] Kilby, J. Miniaturize Electronic Circuits. U.S. Patent #3,138,743, issued 1964.

[Kilby, 2000] Kilby, J. Turing potential into realities: the invention of the integrated circuit. Nobel Lecture, Dec. 2000.

[Russell, 1978] Russell, The Cray-I Computer System. *Communications of the ACM*. Vol. 21, Issue 1 (Jan. 1978) Special issue on computer architecture pp. 63-72.

[Weiser 1993] Mark Weiser. 1993. Some computer science issues in ubiquitous computing. Commun. ACM 36, 7 (July 1993), 75-84.