

Quantum Computing Algorithms and Applications

*Georgia Institute of Technology | College of Computing
CS 4803 | Spring 2027 | 3 credit hours*

Course information

Instructor: Jaideep Mulherkar
Email: jmulherkar3@gatech.edu
Meetings: Tuesday and Thursday, 11:00 a.m.–12:15 p.m.
Room: CoC 101
Office hours: TBD
Teaching assistants and contact information: TBD

The course meets twice each week for 75 minutes per session. Canvas is the primary location for course materials, assignments, announcements, and feedback.

Course description

Quantum computers process information using quantum states and operations that differ fundamentally from classical bits and logic gates. This course introduces the basic principles of quantum computation—including qubits, superposition, entanglement, measurement, and quantum circuits—and develops them from a computer science and algorithmic perspective.

Students will learn how these quantum primitives can be used to design algorithms that solve certain computational problems differently, and sometimes more efficiently, than classical algorithms. Topics include Grover search, amplitude amplification and estimation, quantum walks, Hamiltonian simulation, QAOA, and selected algorithms demonstrating exponential quantum speedups. Applications are drawn from search, cryptography, financial modeling, graph problems, optimization, and molecular simulation.

The course emphasizes hands-on programming: students implement quantum algorithms, compare them with classical approaches, and investigate when theoretical quantum speedups translate into meaningful computational advantages. No prior knowledge of quantum mechanics or quantum computing is required.

Why take this course

Quantum computing raises practical questions for computer scientists: Which problems can quantum computers solve differently? Where can they provide a computational advantage? What assumptions underlie that advantage, and when is a classical algorithm still the better choice?

Applications drive the course: searching for feasible solutions, estimating risk with Monte Carlo methods, exploring graphs, simulating molecules, and solving optimization problems. You will implement representative methods and compare them with classical alternatives, examining when a quantum approach could help and what limits its usefulness. These experiences will prepare you to assess applications in scientific computing, finance, optimization, and cryptography.

Prerequisites

Students must have completed:

- MATH 1554 – Linear Algebra
- CS 3510 – Design and Analysis of Algorithms

Students should be comfortable with Python programming. No prior quantum mechanics is required.

Learning outcomes

- Apply linear algebra and algorithmic reasoning to construct and analyze quantum circuits and algorithms.
- Implement quantum algorithms and simulations using Python-based computational tools.
- Benchmark quantum methods against appropriate classical algorithms and analyze scaling behavior.
- Apply amplitude amplification, amplitude estimation, quantum walks, Hamiltonian simulation, and variational methods to representative computational problems.
- Evaluate the role of problem structure, input models, precision, and resource requirements in claims of quantum advantage.
- Communicate the capabilities and limitations of quantum algorithms in search, optimization, sampling, cryptography, graph analytics, and molecular simulation.

Course materials and references

No textbook purchase is required. Course lecture notes and assignment materials are the primary instructional resources. Required readings will be identified in Canvas; other references provide optional background and enrichment.

- Noson S. Yanofsky and Mirco A. Mannucci, Quantum Computing for Computer Scientists. Cambridge University Press. Supplementary explanations and examples for students with a computer science background.
- Michael A. Nielsen and Isaac L. Chuang, Quantum Computation and Quantum Information. Cambridge University Press. A reference for deeper study of quantum computation and quantum information.
- Eleanor G. Rieffel and Wolfgang H. Polak, Quantum Computing: A Gentle Introduction. MIT Press. A step-by-step reference for quantum states, measurement, circuits, and foundational algorithms.
- Jack D. Hidary, Quantum Computing: An Applied Approach. Springer. Supplementary readings on applications, programming, and hybrid quantum–classical methods.

IBM Quantum Learning — courses and tutorials

Selected material from Basics of Quantum Information and Fundamentals of Quantum Algorithms will supplement the lectures. Relevant documentation and additional readings will be linked with each assignment.

Programming environment

Laboratories use Python, NumPy, and Qiskit. NVIDIA CUDA-Q may also be used in selected laboratories for quantum programming and simulation. Setup instructions, access guidance, and supported versions will accompany each lab. Starter code provides function interfaces and implementation guidance; prior expertise in Qiskit or CUDA-Q is not required.

Keep code, notebooks, and results backed up. Use the specified environment and check that submissions run reproducibly. Report installation or access difficulties early. NVIDIA CUDA-Q documentation is available at <https://nvidia.github.io/cuda-quantum/latest/>.

Course components and assessment

Lectures combine explanations, worked examples, classical–quantum comparisons, and short Think It questions. These questions are normally ungraded and are intended to help you reason through new ideas and discuss alternative approaches.

Assessment	Course weight
Coding laboratories	30%
Two midterm examinations	40%
Final project	30%

Coding laboratories

Five coding assignments develop implementation, experimentation, benchmarking, and interpretation skills. The fifth assignment combines Hamiltonian simulation and QAOA. Credit depends on the required reasoning and analysis as well as working code. Explain experimental results and use an appropriate classical baseline where requested.

The five coding laboratories have equal weight, contributing 6% each to the final course grade.

Midterm examinations

Two midterms assess conceptual understanding, circuit and algorithmic reasoning, and classical–quantum comparisons. Coverage, permitted resources, and exam arrangements will be announced in Canvas. AI tools are not permitted unless expressly authorized.

The two midterm examinations have equal weight, contributing 20% each to the final course grade.

Final project

The project investigates a quantum algorithm or application and compares it with an appropriate classical approach. The report and presentation should explain the problem, implementation or analysis, experimental evidence, resource requirements, assumptions, and limitations. Detailed milestones and grading criteria will be provided in the project specification.

The final project is completed in groups. Each student must contribute meaningfully, and the submission must identify individual contributions. There is no separate final examination; the project report and presentations serve as the culminating assessment.

Grading questions

Request a review within one week of a grade being posted. Contact the instructional team privately, identify the specific item, and explain the suspected grading error. Do not post your graded work or personal grade information publicly.

Academic integrity and collaboration

All work must comply with the Georgia Tech Honor Code. Plagiarism, unauthorized collaboration, use of prohibited resources, fabricated results, and misrepresentation of authorship are not permitted. Suspected violations will be handled through the applicable Institute process.

Unless an assignment explicitly permits team submissions, write and submit your own code, derivations, and explanations. You may discuss general concepts and approaches, but do not exchange completed solutions or copy another student's work. Acknowledge substantive assistance and cite external sources. Do not publish assignment solutions or restricted course materials.

Georgia Tech Office of Student Integrity

Artificial intelligence policy

AI tools may support learning, but they must not replace the reasoning and implementation that an assignment asks you to perform. Assignment-specific instructions may restrict AI use further or explicitly authorize additional uses. If a use is unclear, ask before proceeding.

- Permitted support includes conceptual explanations, additional practice examples, clarification of Python or Qiskit documentation, and help diagnosing errors in code you have attempted.
- Unless explicitly authorized, do not use AI to generate complete assignment solutions, required derivations, experimental analyses, or report sections for submission. Debugging assistance must not become generation of the required solution.
- AI may help improve spelling, grammar, and phrasing, but the submitted ideas, explanations, analysis, and conclusions must be your own.
- Exams and individual assessment quizzes prohibit AI unless the instructor explicitly permits it. Follow the stated instructions for each in-class activity.

Include a brief AI-use statement with each laboratory and project submission. Identify the tool, the purpose of its use, and the parts of the work it assisted. If you did not use generative AI, state "No generative AI used." Disclosure does not make an otherwise prohibited use acceptable.

You are responsible for the accuracy and integrity of all submitted work. Verify suggested code, mathematical claims, citations, and results. You may be asked to explain, test, debug, or modify your code and to justify your reasoning. Do not submit material you cannot explain.

Do not provide AI services with another student's work, personal or confidential information, exam questions, or restricted course materials. Use general examples or minimal code excerpts where appropriate, and respect any restrictions stated in the assignment.

Attendance and missed work

Regular attendance and participation are expected, but attendance is not a separate graded component. Think It questions are normally ungraded. Students are responsible for announcements and material covered during an absence. Attendance is required for examinations and scheduled presentations, subject to approved accommodations and applicable excused-absence provisions.

Notify the instructor as early as possible about an absence affecting an assessment. For illness or personal emergencies, contact Student Life when assistance or absence verification is needed; do not send sensitive medical records to the instructional team. Missed-work arrangements will follow Institute policy. For religious observances, notify the instructor in writing within the first two weeks of class, consistent with that policy.

[Georgia Tech attendance policy](#)

[Student Life assistance and absence support](#)

Deadlines and extensions

Submit work through the location specified in Canvas and verify that the correct files were received. Contact the instructional team promptly if a technical problem prevents submission. Request extensions as early as practicable; emergencies and approved accommodations will be considered under applicable Institute policies.

Late assignments incur an automatic deduction of 5% of the assignment's maximum points for each day late, for up to five days. Each started 24-hour period after the posted deadline counts as one day. The deductions are 5%, 10%, 15%, 20%, and 25% for days one through five. Submissions close 120 hours after the deadline and are not accepted thereafter, except for approved extensions or accommodations. For example, work earning 80/100 and submitted one day late receives 75/100. Scores cannot fall below zero. This policy does not reschedule exams or presentations.

Communication and professionalism

Check Canvas announcements regularly. Use Ed Discussion, accessible through Canvas, for course questions and discussion; do not post complete solutions publicly. Send personal concerns and grade questions privately to the instructor. Include the course number and section in email subject lines, and allow up to two business days for email responses.

Treat classmates and instructional staff respectfully in person and online. Ask questions, disagree constructively, and help maintain an environment in which students can work through unfamiliar ideas without embarrassment.

Accessibility and student support

Students seeking disability-related accommodations should work with the Office of Disability Services and share their approved accommodation letter with the instructor as early as possible so arrangements can be made. Contact the instructor if course materials or activities create an access barrier.

[Office of Disability Services](#)

If stress, personal circumstances, or mental health concerns interfere with learning, seek support early. Georgia Tech's Center for Mental Health Care and Resources provides information about available services and urgent support.

[Center for Mental Health Care and Resources](#)

Instructional plan

The following plan identifies the course topics and intended sequence. Canvas will provide session-level readings, assessment dates, and any announced adjustments. University holidays and the academic calendar determine the final meeting schedule.

Week 1 From Classical to Quantum Information

Double-slit experiments motivate interference and probability amplitudes. Classical deterministic and stochastic operations, reversible computation, qubit states, normalization, and the distinction between amplitudes and probabilities.

Week 2 Quantum Gates and Multiqubit Systems

Single-qubit gates and unitary evolution; measurement in different bases; tensor products, joint states, controlled gates, and entanglement. Compare classical joint probability distributions with quantum states and measurement outcomes.

Week 3 Measurement and Quantum Information Tasks

Observables, projective measurement, two-qubit observables and parity, expectation values, and uncertainty. Teleportation and the CHSH game illustrate communication protocols and correlations beyond classical models.

Week 4 No Cloning and First Quantum Algorithms

The no-cloning theorem and its implications; reversible oracles, phase kickback, and the Deutsch and Deutsch-Jozsa algorithms. Compare classical and quantum query requirements and distinguish exact from randomized classical methods.

Week 5 Grover Search

Grover's algorithm, oracle construction, diffusion operator, geometric interpretation, query complexity, and optimality. Applications: unstructured database search and constraint satisfaction/SAT search.

Week 6 Midterm Examination 1

Assessment of quantum primitives, multi-qubit reasoning, interference, oracle algorithms, and Grover search.

Week 7 Amplitude Amplification

Generalization of Grover search to probabilistic subroutines; success-probability amplification; unknown or variable success probabilities and algorithmic cost. Applications: accelerating randomized search, constraint satisfaction, and boosting low-success-probability quantum subroutines.

Week 8 Monte Carlo and Quantum Amplitude Estimation

Classical Monte Carlo estimation and sampling error; quantum amplitude estimation, emphasizing iterative and modern variants. Applications: financial risk metrics, option pricing, reliability estimation, and rare-event probability estimation.

Instructional plan continued

Week 9 Classical and Quantum Walks

Classical random walks and mixing intuition; discrete-time quantum walks; interference and hitting/search behavior; Grover search viewed through a walk perspective. Applications: spatial and graph search, search on hypercubes, network exploration, and graph-structured algorithm design.

Week 10 Midterm Examination 2

Assessment of amplitude amplification and estimation, Monte Carlo comparison, and quantum-walk algorithms.

Week 11 Hamiltonian Simulation

Hamiltonian evolution, product formulas/Trotterization, simulation error, and resource scaling. Case study: H₂ molecular Hamiltonian. Applications: quantum chemistry, molecular energy estimation, materials modeling, and simulation of quantum dynamical systems.

Week 12 QAOA and Hybrid Quantum-Classical Optimization

QAOA construction, cost and mixer Hamiltonian, benchmarking against classical heuristics. Applications: MaxCut, graph partitioning, scheduling, resource allocation, and other combinatorial optimization problems.

Week 13 Exponential Speedups

Period Finding and Shor's Algorithm: Quantum Fourier transform and phase/period-finding ideas at a conceptual algorithmic level; Shor's factoring algorithm and complexity implications. Applications: integer factoring, public-key cryptography, and post-quantum security motivation.

Week 14 Quantum Advantage Dequantization and Limits

Meaningful quantum advantage, resource costs, and input and data-access assumptions; a high-level introduction to quantum error correction and why reliable quantum computation needs it. Recommendation systems and Ewin Tang's dequantization result illustrate how stronger classical algorithms can change quantum-speedup claims.

Week 15 Final Project Workshop and Presentations

Student project discussions and presentations. Projects implement or analyze a quantum algorithm in an application domain, compare it with an appropriate classical baseline, and discuss scaling, resources, assumptions, and limitations.

Final examination week: Final project reports and/or presentations, as scheduled in Canvas. No separate final examination.

How to succeed in this course

Work through small examples by hand before coding. Start laboratories early, test intermediate results, and ask questions when a step is unclear. In each algorithm, identify what the classical approach does, what changes in the quantum approach, and which assumptions make the comparison meaningful.