



CS 417 (Fall 2026 - Second 8 Weeks)

Thomas J. Kennedy

Last modified: Aug 10, 2026

Contents:

[1 Basic Course Information](#)

[1.1 Catalog Description](#)

[1.2 Overall Description](#)

[1.3 Prerequisites](#)

[1.4 When and Where](#)

[1.5 Instructors](#)

[2 Required Materials](#)

[2.1 Textbooks](#)

[2.2 Supplemental Materials](#)

[2.3 Technology Requirements](#)

[3 Course Objectives](#)

[4 \(Tentative\) Course Topics](#)

[5 Course Schedule](#)

[6 Grading](#)

[6.1 Canvas - Total is Wrong!?](#)

[6.2 Canvas - Extra Credit Value is Decreasing!?](#)

[6.3 Exams](#)

[6.4 Incomplete Grades](#)

[7 Expectations](#)

[8 Course Policies](#)

[8.1 Due Dates & Late Submissions](#)

[8.2 Scroll Down](#)

[8.3 Submission \(Written Exercises\)](#)

[8.4 Submission \(Knowledge Checks\)](#)

[8.5 Submission \(Programming Exercises\)](#)

[8.6 Academic Honesty](#)

[9 University Policies](#)

[9.1 Code of Student Conduct and Academic Integrity](#)

[9.2 Honor Pledge](#)

[9.3 University Email Policy](#)

[9.4 Withdrawal](#)

[9.5 Educational Accessibility](#)

1 Basic Course Information

The course schedule and website are located at <https://www.cs.odu.edu/~tkennedy/cs417/latest>.

1.1 Catalog Description

Algorithms and software for fundamental problems in scientific computing. Topics: properties of floating point arithmetic, linear systems of equations, matrix factorizations, stability of algorithms, conditioning of problems, least-squares problems, eigenvalue computations, numerical integration and differentiation, nonlinear equations, iterative solution of linear systems.

1.2 Overall Description

Computer Science involves a fair bit of math and logic. However, most students have a misplaced fear of or frustration with mathematics...

1. Many students look at a math problem, see unfamiliar notation, and *psyche themselves out*.
2. Many students hear the word “*proof*” and have no idea where to start.

My overarching goals are...

1. to help you build the intuition that you need to make use of your existing math knowledge.
2. to help you develop an intuition for constructing and analyzing proofs.
3. to develop a natural curiosity along with a strong foundation for further mathematical endeavors.

1.3 Prerequisites

Students are expected to have a strong foundation in mathematics and programming:

1. **Math 316** - Good understanding of calculus and linear algebra.
2. **CS 250 or CS 251 or CS 253** - Programming skills in a language such as: C/C++, Python, Rust, or Java.

1.3.1 General Programming Knowledge

Students should be familiar with certain basic programming techniques that are largely independent of any specific programming language:

- using editors, compilers and other basic software development tools.
- basic software design (i.e., stepwise refinement and top-down design)
- software testing, including the use of scaffolding code (stubs and drivers), selection of test cases for black-box testing, and head to head testing.
- debugging, including the use of debugging output, the use of automatic debuggers to set breakpoints and trace program execution, and the general process of reasoning backwards from failure locations to the faulty code responsible for the failure.

1.3.2 C, C++, Java, Python, or Rust Knowledge

I will assume that you are familiar with the basics of C, C++, Java, Python, or Rust. This includes:

- the various statements and control-flow constructs,
- the built-in data types,
- the use of arrays, pointers, pointers to arrays, and linked lists,
- the use and writing of functions, and
- the basic use of structs and/or classes for implementing abstract data types.

1.3.3 General Computer Literacy

You will be studying techniques in this course for preparing professional-quality software documentation. The key embedded word in “software documentation” is “document”. Students taking this course should be able to use word processors and other common tools to produce good quality documents, including mixing text and graphics in a natural and professional manner.

1.4 When and Where

This course is asynchronous. There are no required nor scheduled lecture meetings.

1.5 Instructors

Mr. Thomas Kennedy & Dr. Nikos Chrisochoides

General Information	
Instructor	Thomas J. Kennedy
Office	Monarch Hall 2123G
Phone #	757.683.7725
Email	tkennedy@cs.odu.edu
Home Page & Office Hours	https://www.cs.odu.edu/~tkennedy/

Important: All email related to this course should have the phrase **CS 417** somewhere in the subject line. This flags your message in my mailbox for faster attention.

I try to respond to all (properly marked) messages

- before I leave campus each day (Monday through Friday)
- within 48 on weekends & holidays.

This is my general email policy. However, this is usually $O(n)$ (i.e., an upper bound).

Any delays in replying to email will be noted in a Canvas Announcement.

1.5.1 Office Hours

My general office hours are available at <https://www.cs.odu.edu/~tkennedy/>. Instructions for scheduling a formal appointment are listed on the same page and [here](#). My office hours consist of web-conference (Zoom) appointments.

General questions about course content and reports of website problems should normally be asked in the Forums on Canvas or via email.

2 Required Materials

2.1 Textbooks

Class notes are available on Canvas and the course site. The required **Chapter Texts** are provided through Canvas.

2.2 Supplemental Materials

Supplemental texts are optional:

1. (Optional) Elementary Numerical Analysis: An Algorithmic Approach, by Samuel Daniel Conte, Carl De Boor
2. (Optional) Numerical Analysis: An Introduction, By Walter Gautschi.
3. (Optional) Data Structures and Algorithms in Python Book by Michael H. Goldwasser, Michael T. Goodrich, and Roberto Tamassia
4. (Optional) Advanced Calculus: Second Edition by David V. Widder

2.3 Technology Requirements

2.3.1 Computer Accounts

Students will need an account on the CS Dept. Linux network to participate in this class. This account is unrelated to any University-wide account you may have from ODU's Information Technology Services (ITS).

If you have had a CS Unix account in the recent past, you should find it still active with your login name, password, and files unchanged. If you have had an account and it has not been restored, contact the CS Dept systems staff at root@cs.odu.edu requesting that it be restored.

If you do not yet have such an account, go to the [CS Dept. home page](#) and look for "Account Creation" under "Online Services". All students in this course are responsible for making sure they have a working CS Unix account **prior to** the second week of class.

2.3.2 Compilers and Interpreters

The "official" environment in which students' programming assignments will be evaluated is defined by our Dept. Linux servers. It is the student's responsibility to be sure that their code compiles and executes using the compilers and run-time environment provided there. As of this writing, the compiler and run-time environment versions used are...

- C++: g++ 11.4.0
- Java: openjdk version "21.0.8"

- Python: 3.11 & 3.13
- Rust: 1.96

You may want to install [uv](#). It will handle the creation of Python virtual environments and the installation of any required libraries. Most Python examples can be run with...

```
uv run [program].py
```

...in place of...

```
python3.11 [program].py
```

3 Course Objectives

This course is designed as an introduction to the basics of numerical analysis. At the end of this course students will be able to:

1. Examine the impact of finite arithmetic on computation.
2. Formulate numerical approximation problems and investigate their solutions via programming.
3. Investigate the relationship between mathematical models and their matrix representations.
4. Synthesize solutions from simpler parts.
5. Design algorithms and prove their correctness using theory/analysis as building blocks.
6. Examine the spatial complexity, temporal complexity, and computational complexity of numerical solvers.
7. Implement numerical software using existing libraries.
8. Test and verify the correctness of their own codes using model problems and principles from theory.
9. Utilize vocabulary and terminology used by engineers and applied mathematicians.
10. Summarize the literature and utilize basic software packages on numerical approximation.
11. Discuss research issues in numerical computing.

4 (Tentative) Course Topics

Part I*

1. MACHINE ARITHMETIC AND RELATED MATTERS
 - Machine Arithmetic and Rounding
 - The Condition of a Problem and Algorithm
2. APPROXIMATION
 - Least Squares Approximation
3. INTERPOLATION
 - Polynomial Interpolation
 - Approximation and Interpolation by Spline Functions
4. NONLINEAR EQUATIONS
 - Examples, Iteration, Convergence, and Efficiency
 - Method of False Position
 - Secant Method
 - Newton's Method

Part II

1. NUMERICAL DIFFERENTIATION AND INTEGRATION

- Numerical Differentiation
- Numerical Integration

Topics of Interest (Time Permitting)

1. MATRIX COMPUTATIONS – SYSTEMS OF LINEAR EQUATIONS

- Gauss Elimination and LU Factorization: Basic Algorithms
- Iterative Methods: Basic Algorithms

The topics covered and time spent on each topic may change based on class performance and class pacing.

5 Course Schedule

The course schedule is available in Canvas under *Modules* and *Calendar*. An alternate schedule/outline can be accessed at <https://www.cs.odu.edu/~tkennedy/cs417/latest/Directory/outline/index.html>.

6 Grading

Final grades will be computed using the following weights:

	CS 417	CS 517
Homework	40%	35%
Knowledge Check Quizzes	40%	40%
Machine Assignments	--	5%
Semester Project (Final Exam)	20%	20%
Total	100%	100%

Final course grades will be assigned based on the standard 10-point scale.

Example 1: Grading Schema

Percent Letter Grade 4pt Value

≥ 94	A	4.0
≥ 90	A-	3.7
≥ 87	B+	3.3
≥ 84	B	3.0
≥ 80	B-	2.7
≥ 77	C+	2.3
≥ 74	C	2.0
≥ 70	C-	1.7
≥ 67	D+	1.3
≥ 64	D	1.0

Percent	Letter Grade	4pt Value
≥ 60	D-	0.7
< 60	F	0.0

1. CS417 students are encouraged to complete Machine Assignments. 5% of your Machine Assignment average will be added to your Final Grade.

6.1 Canvas - Total is Wrong!?

In Canvas... your Interim Total does not always match these weights. **At the end of the semester... your Total Grade (i.e., Final Grade) will be correct.**

There are a few obvious discrepancies if you are a CS 417 student...

1. *Homework* is only 35% instead of 40%
2. *Machine Assignments* are 5% instead of showing up as Extra Credit
3. There is a *417/517* column
4. There is a *417 - Homework Adjustment* column
5. There is a *Grade Computation Category* worth 5%

and... **Weights add up to 105%.** Due to

1. how Canvas grading works
2. how Extra Credit is applied
3. the Homework and Machine Assignment differences for CS 417/517

we have a bit of a [Rube Goldberg machine](#).

If you are a CS 517 student... your *Total* is always correct.

6.1.1 Canvas - Homework

If you are a CS 417 student your Homework Total is shown with a weight of 35%. You will see the same total listed under *417 - Homework Adjustment* with a weight of 5%. And...

$$35 + 5 = 40$$

...which is the correct *Homework* weight.

If you are a CS 517 student... the *417 - Homework Adjustment* will be blank.

6.1.2 Canvas - Machine Assignments

If you are a CS 417 student... *Machine Assignments* are optional. They can add up to 5 points to your final course grade using...

$$0.05 * (\text{Machine Assignment Average})$$

However, Canvas requires that all students have the same Final Grade Computation. The *Machine Assignment* average does impact your Total Grade **until** the end of the semester when all grade are entered.

At the end of the semester... Machine Assignments will be correctly included in your grade (i.e., as points).

If you are a CS 517 student... the 5% weight is correct.

6.2 Canvas - Extra Credit Value is Decreasing!?

This is can be explained with some quick math (well... arithmetic). Suppose that you have

- earned 10 points of *Homework* extra credit
- earned 10 out of 10 points on Homework 0

The Homework portion of your grade uses the standard earned-divided-by-possible computation...

$$(\text{total earned points} + \text{extra credit points}) / (\text{total possible points})$$

In our example this would become...

$$10 / 10 = 1.0 = 100\%$$

...before Extra Credit... and...

$$(10 + 10) / 10 = 2.0 = 200\%$$

...after incorporating Extra Credit.

As you complete more *Homework* the Extra Credit will still be worth the same number of points. But the *Homework* possible point-total will increase.

Consider the following table. For this example... assume a perfect score on every exercise.

Most Recent Assignment	Total Earned	+15 Points	Possible Points	Avg.	Avg. with Extra Credit
Homework 1 - Personal Introduction	10	25	10	100	250.0
Homework 2	20	35	20	100	175.0
Homework 3	30	45	30	100	150.0
Homework 4 - Proof 1	40	55	40	100	137.5
Homework 5	50	65	50	100	130.0
Homework 6	60	75	60	100	125.0
Homework 7 - Proof 2	70	85	70	100	121.4
Homework 8	80	95	80	100	118.7

6.3 Exams

The Semester Project serves as the Final Exam.

6.4 Incomplete Grades

A grade of “I” indicates assigned work yet to be completed in a given course, or absence from the final examination, and is assigned only upon instructor approval of a student request. The “I” grade may be awarded only in exceptional circumstances beyond the student’s control. The “I” grade becomes an “F” if not removed by the day grades are due for following term based on specific criteria: [Incomplete, Withdraws and Z grades.](#)

7 Expectations

This course covers theory oriented and math oriented topics and problems. You are expected to review required mathematics, programming, and theory as directed. If whilst reviewing you would like direction or clarification contact the instructor (via email or in the discussion board).

We discuss various notations (for math and pseudocode) in this course. Keep in mind *“That it is just notation.”* Familiarizing oneself with unfamiliar notations is a skill we will endeavor to develop this semester.

If you have questions... Ask the instructor The cliché *“If you have a question... at least one of your classmates has the same question.”* is true (especially this semester).

8 Course Policies

8.1 Due Dates & Late Submissions

I do not accept late work. Exceptions to this and other grading policies will be made only in situations of unusual and unforeseeable circumstances beyond your control, and such arrangements must be made prior to the due date in any situations where the conflict is foreseeable.

8.2 Scroll Down

Make sure to scroll down. Most exercises (Homework Assignments, Machine Assignments, and Exams) consist of multiple pages.

8.3 Submission (Written Exercises)

All written exercises must be submitted through Canvas in PDF format. Each such submission must take the form of a single multi-page PDF document. **Failure to follow these requirements will result in a zero.**

All work must be readable. Make sure that all submitted PDFs are readable, e.g.,

- content is not cut off
- writing (or text) is not faded (i.e., too light to read)

Problems must be submitted in order. Any scans with the problems listed out of order will result in a zero.

Show your work. Partial credit will not be awarded if:

- intermediary steps are not shown
- intermediary steps are not legible

If there is insufficient work to show how you arrived at an answer, you will not receive full credit.

8.4 Submission (Knowledge Checks)

With a few exceptions... every Module will have a *Knowledge Check*. These will take the form of Canvas Quizzes. Knowledge Checks will be open notes, open book, and open lecture recordings.

Quizzes are **not** proctored.

8.5 Submission (Programming Exercises)

All programming exercises (e.g., Machine Assignments) must be submitted as a single zip file (zip) or tarball (tar.gz, tar, or tgz). Unless explicitly stated in the prompt, other formats will not be accepted. Each submission must include:

1. A brief [ReadMe file](#) that specifies how to compile and run your code.
2. Makefiles (or equivalent build files) for any compilable code.
 - For C/C++, I recommend make or Cmake
 - For Java, I recommend Gradle
 - For Rust, stick with Cargo
 - For Python, no build file is required. Python is an interpreted language

IDE (e.g., Code::Blocks, Eclipse, or VSCode) projects are **not** build files. No credit (i.e., zero points) will be awarded for any compilable code submitted without an appropriate build file.

All code must follow best practices, including:

- Documentation
- Indentation and spacing
- Naming conventions
- Top-down design (i.e., no monolithic functions)
- D.R.Y
- Language specific best practices

8.5.1 Programming - Permitted Languages

You may use any combination of C, C++, Java, Python (3.11+), and Rust. All other languages must be discussed with the instructor before assignment/project submission.

Extra consideration will be given to **novel solutions** (e.g., those written in a functional language).

8.5.2 Programming - Extra Credit

Any code submissions with a formal test suite may receive extra credit. The amount of extra credit will be determined by...

- the specific exercise
- completeness of the test suite
- quality of the test suite

8.6 Academic Honesty

Everything turned in for grading in this course must be your own work.

The instructor reserves the right to question a student orally or in writing and to use his evaluation of the student's understanding of the assignment and of the submitted solution as evidence of cheating. Violations will be reported to the Office of Student Conduct & Academic Integrity for consideration for possible punitive action.

Students who contribute to violations by sharing their code/designs with others may be subject to the same penalties.

- Students are expected to use standard Unix protection mechanisms (chmod) to keep their assignments from being read by their classmates. Failure to do so will result in grade penalties, at the very least.

This policy is *not* intended to prevent students from providing legitimate assistance to one another. Students are encouraged to seek/provide one another aid in learning to use the operating system, in issues pertaining to the programming language, or to general issues relating to the course subject matter.

Students should avoid, however, explicit discussion of approaches to solving a particular programming assignment, and under no circumstances should students show one another their code for an ongoing assignment, nor discuss such code in detail.

Use of Online Resources

You may **not** post details of course assignments, projects, or tests at online Forums, Bulletin Boards, Homework sites, etc., soliciting help.

You may use information that you have not solicited but have located, subject to the following restrictions:

- Just as when writing a paper, if you use someone else's ideas, you must cite your sources appropriately. Within code, such citations appear in comments.

Example:

```

:
double x = 23.0;
double xsqrt = sqrt(x);
// Search algorithm based upon code by S Zeil at
// https://www.cs.odu.edu/~zeil/cs361/latest/Public/functionAnalysis/index.html#orderedsequ
int loc = 0;
while (loc < arraySize && numbers[loc] < xsqrt)
:

```

- Just as when writing a paper, if you use someone else's words (code), you must cite your sources appropriately **and** mark the quoted text. Within code, such citations appear in comments.

Example:

```

:
double x = 23.0;
double xsqrt = sqrt(x);

// Begin quoted code from S Zeil at
// https://www.cs.odu.edu/~zeil/cs361/latest/Public/functionAnalysis/index.html#orderedsequ
int loc = 0;
while (loc < listLength && list[loc] < searchItem)
{
    ++loc;
}
// End quoted code

:

```

- Failure to appropriately cite any such “found code” will be taken as evidence of plagiarism.
- The overall principle stated in the first sentence of this section remains in effect. “Everything turned in for grading in this course must be your own work.” If the bulk of your assignment, code, paper, etc., are copied, even with appropriate citation, to the degree that, in the judgment of the instructor, you have not demonstrated your own knowledge of the course material, you will receive a zero for that exercise.

9 University Policies

9.1 Code of Student Conduct and Academic Integrity

The Office of Student Conduct & Academic Integrity (OSCAI) oversees the administration of the student conduct system, as outlined in the Code of Student Conduct. Old Dominion University is committed to fostering an environment that is: safe and secure, inclusive, and conducive to academic integrity, student engagement, and student success. The University expects students and student organizations/groups to uphold and abide by standards included in the [Code of Student Conduct](#). These standards are embodied within a set of core values that include personal and academic integrity, fairness, respect, community, and responsibility.

9.2 Honor Pledge

By attending Old Dominion University, you have accepted the responsibility to abide by the Honor Pledge:

I pledge to support the Honor System of Old Dominion University. I will refrain from any form of academic dishonesty or deception, such as cheating or plagiarism. I am aware that as a member of the academic community it is my responsibility to turn in all suspected violations of the Honor Code. I will report to a hearing if summoned.

9.3 University Email Policy

Reformatted to follow <https://ww1.odu.edu/about/policiesandprocedures/computing/standards/11/02>.

9.3.1 Student Email

With the increasing reliance and acceptance of electronic communication, email is considered an official means for University communication. Old Dominion University provides each student an email account for the purposes of teaching and learning, research, administration, and service. It is important that all students are aware of the expectations associated with email use as outlined in the [Student Email Standard](#).

9.3.2 Email Account Activation

It is the responsibility of every eligible student to activate MIDAS, the Monarch Identification and Authorization System, in order to obtain email access.

9.3.3 Expectations Regarding Use of Email

The email account provided by the University is considered to be an official point of contact for correspondence. Students are expected to check their official e-mail account on a frequent and consistent basis in order to stay current with University communications. Mail sent to the ODU email address may include notification of University-related actions, including academic, financial, and disciplinary actions. For more information about student email, please visit [Student Computing](#).

9.3.4 Educational Uses of Email

University offices and instructors cannot validate that a communication coming by email is from an ODU student unless it comes from a valid ODU email address. If students send mail from non-ODU email accounts (e.g., Hotmail or Yahoo), faculty and staff are not obligated to respond and may request that official e-mail accounts be used.

9.4 Withdrawal

Enrollment in this course indicates your acceptance of its teaching focus, requirements, and policies. Please review the syllabus and the course requirements as soon as possible. If you believe that the nature of this course does not meet your interests, needs or expectations, if you are not prepared for the amount of work involved – or if you anticipate assignment deadlines or adherence to course policies will constitute an unacceptable hardship for you – you should drop the course by the drop/add deadline, which is listed in the [ODU Schedule of Classes](#). Visit the [Office of the University Registrar](#) for more information.

9.5 Educational Accessibility

Old Dominion University is committed to ensuring equal access to all qualified students with disabilities in accordance with the Americans with Disabilities Act. The Office of Educational Accessibility (OEA) is the campus office that works with students who have disabilities to provide and/or arrange reasonable accommodations.

- If you experience a disability which will impact your ability to access any aspect of my class, please present me with an accommodation letter from OEA so that we can work together to ensure that appropriate accommodations are available to you.
- If you feel that you will experience barriers to your ability to learn and/or testing in my class but do not have an accommodation letter, please consider scheduling an appointment with OEA to determine if academic accommodations are necessary.

The Office of Educational Accessibility is located at 1021 Student Success Center and their phone number is (757) 683-4655. Additional information is available at the [OEA website](#)



© 2018-2026, Old Dominion Univ.