SYLLABUS
MPATE-GE 2618: C Programming for Music Technology

Steinhardt School of Culture, Education, and Human Development
Music and Performing Arts
Department of Music Technology

Instructor:	Dr. Schuyler Quackenbush Audio Research Labs www.audioresearchlabs.com
Email:	schuyler.quackenbush@nyu.edu
Office Hours:	Tuesdays and Thursdays immediately after class, or as indicated in Office Hours link on NYU Classes page
Lab TA:	Dirk Vander Wilt
Email:	dirk.vanderwilt@nyu.edu

Course Description

MPATE-GE 2618: C Programming for Music Technology is an intensive, graduate-level introductory course in programming concepts and computer science with a focus on software design, algorithms, and data representation for digital signal processing and audio applications. Assignments consist of extensive programming in C.

Topics include:

- C programming: syntax, primitive types, iteration, conditional expressions, functions, arrays, pointers, dynamic memory allocation, standard libraries.
- Software development: problem decomposition, abstraction, data structures, implementation, debugging, testing.
- Algorithms: design, specification, and analysis.
- Data representations for signal processing and audio applications.
- Introduction to audio software libraries and their APIs

Prerequisites

No prior programming experience is required.

Class Times

Lectures (MPATE-GE 2618) meets on Tuesday and Thursday 2:00- 3:15 at the Zoom URL indicated on NYU Classes.

Lab (MPATE-GE 2617) meets Thursday 3:30-4:45 at the Zoom URL indicated on NYU Classes.

NOTE: All students enrolled in MPATE-GE 2618 must also take MPATE-GE 2617.

Office Hours

Weekly office hours will be immediately after the Tuesday and Thursday lectures at a location at a public space near classroom or lab. Alternatively, as indicated in Office Hours link on NYU Classes page.

Class Materials

All class materials are available at the NYU Classes course website:

- Resources
  - Lecture slides as PDF
  - Any other course material
- Assignments
  - Problem description and any associated source code and data
  - Final project information

Problem Sets

Problem sets will be distributed via NYU Classes Assignments. Each will have a posted due data. Students should post completed assignments via the same NYU Classes Assignment mechanism.

There will be 7 "slip days" that students can use to accommodate handing in problem sets after the assigned due date. They can be used on several problem sets, or used on a single problem set. Without using slip days, assignments handed in after the due date are subject to a reduction in points awarded, as follows:

- First 6 days, reduction of 5% per day
- Next 7 days, reduction of 10% per day

Final Project

The final project will be your opportunity to put your programming skills to use and implement your own software application. As long as your project is written in C, the nature of your project is entirely up to you, albeit subject to the instructor's approval. You are welcome to utilize external libraries and hardware provided that the instructor has access to all of these things and you are able to present it in class at the end of the semester.

Final projects include a written report, the actual program code, and a presentation to the class.

Students in the music technology program are encouraged to choose a project topics that involve processing of audio signals, including real-time audio I/O.

Final Exam

The course does not have a final exam.

Grades

Final grades for the lecture class and lab will be determined using the following weights (these are approximate, actual weights determined by points assigned to each problem set and final project):

Problem Sets:	80%
Final Project:	20%

The instructor will post grades for each assignment and final project on the NYU Classes Gradebook.

Books

Required text:
Programming in C, Fourth Edition by Stephen Kochan

Optional text for students less comfortable with programming:
Absolute Beginner's Guide to C, Second Edition by Greg Perry

Optional recommended audio programming text:
DAFX, Digital Audio Effects, 2011 or 2002 editions, by Udo Zölzer

Platforms

Assignments are designed to be implemented on a UNIX-based system such as Mac OS or Cygwin on Windows.

Course Outline
The instructor may modify the course outline or course problem sets as he deems appropriate to enhance the course. Students are welcome to submit suggests on changes to the course outline or course problem sets.

- Introduction
  - Book
  - Problem sets, Final project and Lab
  - Course grading
  - About instructor

- Programming problem statement
  - Algorithms
- Terminal
  - Bash commands
  - History of Unix
  - Directories
  - Unix $PATH
  - Redirection and pipes
- Compiler, Assembler, Machine code
- Basic C program
  - Hello world
  - Comments in code
  - Code structure

- Binary numbers

    - 2s complement signed integer
    - ASCII for characters
    - float, double
- Versions of C
    - ANSI
    - C99
- C is Strongly typed language
- Statement syntax
- Variables
    - char, int, short, long, long long, float, double
    - variable
        - declaration
        - assignment
    - Expressions
        - Integer division
        - Promotion
        - cast
    - Unsigned
    - Bool
    - #define for defining constants
    - typedef
        - typedef unsigned int counter
- Operators
    - Arithmetic
        - Precedence
        - Cast
        - Promotion in expression evaluation
    - ++, += and other variants
    - Relational
    - Boolean
    - Logical
    - Bit operations

- Flow of execution
    - Conditional
        - If else
        - ? :
        - switch
    - Looping
        - For
        - Break
        - While
        - Do while
    - Style

        - Brace position (K&R or Microsoft)
    - Conditional
        - switch
        - ? :
        -
- Logical operators

- Arrays
    - Integer
        - Declaration
        - Initialization
    - Array access and assignment
        - A[i]
        - Indexed from zero
        - “off-end” access or assignment
    - Character
        - Null terminated
        - String functions (“n” variants)

- Functions
    - Args
        - Call by value (is the default)
        - Call by reference
    - Return values
        - int, double
        - Char * (example of “status strings”);
    - Function prototypes
        - When do you need them within a single file
        - *.h in multiple files
    - Variable scope
        - Local automatic (on stack)
        - Within {} scope e.g. for (int i=0; …) loop
        - Global (in heap)
        - static

- Command line arguments and main()
    - Parsing command line
        - Pointer to string
    - Useful functions
        - atoi, atof,
    - #define and enum {}
    - Array of strings
        - Const modifier, e.g. for “message” strings
- Command line processing

  - Designating options
  - Processing of optional arguments
  - Qualifying arguments

- File I/O
  - Fopen
  - Fclose
  - Character I/O – ASCII
    - fscanf, fprintf
    - getc, putc
  - File I/O – Binary
    - fread/fwrite
  - Function to read a line of text

- Enum, const, typedef
- Pre-processor
  - #if () #else #endif

- Structures
  - Example: Student
  - Accessing structures
    - Direct
    - Pointer to struct
  - Typedef and structures
    - Example
  - Arrays of structures
- Pointers
  - Arguments to functions
  - Arrays
    - Index
    - Pointer
    - Pointer offset and pointer arithmetic
  - Pointers to structures
  - Pointers to functions

- Bit operators
  - Bit operators
    - and, or
    - shift
  - Bit test

- Characters
  - Characters

  - Strings

- Using random numbers
  - Numerical integration
- Recursion
  - Base case
  - Recursive call
  - Example programs

- Memory allocation
  - Why: when array size is unknown at compile time
    - Read in a wav file
  - Malloc, calloc and Free
  - Multi-dimensonal arrays
  - Arrays of structures
  - Memory leaks – malloc() without free()
- Memory usage
  - Virtual memory
  - Pages
  - Protection
  - Caching
  - Calling and Stack frame
  - Variables
    - Global
    - Automatic local
    - Static local

- Computational efficiency
  - Big O
  - Linear Log, Polynomial, Exponential
- DFT, FFT
- Searching algorithms
  - Linear
  - Binary
  - Hash
- Hash tables
  - For searching
- Sorting: quicksort overview

- Multi-file programs

- Linked lists
  - Alternative to arrays

    - Grow
    - Re-order
    - Singlely and doubly linked lists
- Binary Trees
    - Data representation
    - Huffman codes

- Search
    - Exhaustive
    - Greedy
- Example problems
    - Traveling Salesman
    - Shortest Path
    - 0-1 Backpack

- Convolution
    - Time Domain
    - Frequency Domain
    - Reverberation
- Filtering
    - FIR and IIR filtering
    - Latency
    - Arithmetic precision

- Using Libraries
- Real-Time operation
    - latency
- Multi-threaded code
    - Callbacks

- PortAudio
- LibSndFile
    - Open, Read, Write, Seek, Close
- NCurses
    - Non-blocking keyboard input
    - Non-tty output

- C++ Classes
    - scoped data and member functions

- Create VST plugin using JUCE

Statement on Academic Integrity

Students are expected-often required-to build their work on that of other people, just as professional researchers and writers do. Giving credit to someone whose work has helped you is expected; in fact, not to give such credit is a crime. Plagiarism is the severest form of academic fraud. Plagiarism is theft. More specifically, plagiarism is presenting as your own:

- a phrase, sentence, or passage from another writer's work without using quotation marks;
- a paraphrased passage from another writer's work;
- facts, ideas, or written text gathered or downloaded from the Internet;
- another student's work with your name on it;
- a purchased paper or "research" from a term paper mill.

Other forms of academic fraud include:

- "collaborating" between two or more students who then submit the same paper under their individual names.
- submitting the same paper for two or more courses without the knowledge and the expressed permission of all teachers involved.
- giving permission to another student to use your work for a class.

Term paper mills (web sites and businesses set up to sell papers to students) often claim they are merely offering "information" or "research" to students and that this service is acceptable and allowed throughout the university. THIS IS ABSOLUTELY UNTRUE. If you buy and submit "research," drafts, summaries, abstracts, or final versions of a paper, you are committing plagiarism and are subject to stringent disciplinary action. Since plagiarism is a matter of fact and not intention, it is crucial that you acknowledge every source accurately and completely. If you quote anything from a source, use quotation marks and take down the page number of the quotation to use in your footnote.

Consult The Modern Language Association (MLA) Style Guide for accepted forms of documentation, and the course handbook for information on using electronic sources. When in doubt about whether your acknowledgment is proper and adequate, consult your teacher. Show the teacher your sources and a draft of the paper in which you are using them. The obligation to demonstrate that work is your own rests with you, the student. You are responsible for providing sources, copies of your work, or verification of the date work was completed.

Students are responsible for understanding the concept of plagiarism, and knowing and understanding the contents of the University "Statement of Academic Integrity" http://steinhardt.nyu.edu/policies/academic_integrity

Plagiarism will immediately result in a failing grade in the course and the student will be reported to their school's academic Dean.

Students with Disabilities

Academic accommodations are available for students with documented disabilities. Please contact the Moses Center for Students with Disabilities at 212-998-4980 for further information.

Appendix A - Graduate Scale and Rubric

Steinhardt School of Education Grading Scale

There is no A+	
A	93-100
A-	90-92
B+	87-89
B	83-86
B-	80-82
C+	77-79
C	73-76
C-	70-72
D+	65-69
D	60-64
There is no D-	
F	Below 60
IP	Incomplete/Passing
IF	Incomplete/Failing
N	No Grade

Letter Grade Rubic

A—Outstanding Work
An "A" applies to outstanding student work. A grade of "A" features not simply a command of material and excellent presentation (organization, coding, asset management etc...), but importantly, sustained intellectual engagement with the material. This engagement takes such forms as shedding original light on the material, investigating patterns and connections, posing questions, and raising issues.
An "A" assignment is excellent in nearly all respects:

- It is well organized, with a clear focus.
- It is well developed with content that is relevant and interesting.
- It fulfills all the technical and creative requirements of the assignment.
- It demonstrate a clear understanding of the material discussed in class.
- It is engaging

B—Good Work
A "B" is given to work of high quality that reflects a command of the material and a strong presentation but lacks sustained intellectual engagement with the material.
A "B" project shares most characteristics of an "A" project, but

- It may have some minor weaknesses in its implementation, either technical or creative.
- It may have some minor lapses in implementing the one or two required elements.

C—Adequate Work

Work receiving a "C" is of good overall quality but exhibits deficiencies in the student's command of the material or problems with presentation or implementation.

A "C" project is generally competent; it is the average performance. Compared to a "B" paper:

- It may have serious shortcomings in its implementation or organization.
- It fails to meet two to three requirements outlined in the assignment.
- The functionality of one or more elements has been compromised.

D or F—Unsuccessful Work

The grade of "D" indicates significant problems with the student's work, such as a shallow understanding of the material.

- It is messy in its implementation
- It displays major organizational problems
- It fails to fulfill three of more of the requirements outlined in the assignment
- It is unrelevant to the assignment
- It includes confusing transitions or lacks transitions altogether

An "F" is given when a student fails to demonstrate an adequate understanding of the material, fails to address the exact topic of a question or assignment, or fails to follow the directions in an assignment, or fails to hand in an assignment.

Pluses (e.g., B+) indicate that the assignment is especially strong on some, but not all, of the criteria for that letter grade. Minuses (e.g., C-) indicate that the paper is missing some, but not all, of the criteria for that letter grade.