San José State University
Department of Computer Engineering

CMPE 180-92 Data Structures and Algorithms in C++

Section 92
Fall 2017

Course and Contact Information

Instructor:	Ron Mak
Office Location:	ENG 250
Email:	ron.mak@sjsu.edu
Website:	http://www.cs.sjsu.edu/~mak/
Office Hours:	TuTh 3:00 - 4:00 PM
Class Days/Time:	Th 6:00 – 8:45 PM
Classroom:	ENG 189
Prerequisites:	Admission into the Computer Engineering or the Software Engineering master's degree program.

Course Format

This course will be taught primarily face-to-face instruction. Course materials, syllabus, assignments, grading criteria, exams, and other information will be posted on the SJSU Canvas course site at http://sjsu.instructure.com/ You are responsible to check Canvas regularly for class work and exams. You also can find Canvas video tutorials and documentations at http://ges.sjsu.edu/canvas-students

Faculty Web Page and MySJSU Messaging

Course materials such as syllabus, handouts, notes, assignment instructions, etc. can be found on my faculty web page at http://www.sjsu.edu/people/firstname.lastname and/or on Canvas Leaning Management System course login website at http://sjsu.instructure.com/. You are responsible for regularly checking with the messaging system through MySJSU at http://my.sjsu.edu to learn of any updates.

Piazza will be available for announcements and to serve as an online discussion forum for the class. You are responsible for responding to enrollment invitations.

Course Catalog Description

"Individual work in computer engineering. Prerequisite: Upper division standing and instructor consent. Not available to Open University Students."

"Object-oriented data organization and representation as strings, arrays, stacks, queues, dequeues, lists, sets, trees, tables, and graphs. Sorting and searching and algorithm design and performance analysis. Testing methods and data will be discussed."

Course Goals

CMPE 180-92 is an introduction to data structures and algorithm design with C++. The course emphasizes important data structures, such as linked lists, stacks, queues, hash tables, trees, and graphs. It also introduces recursive algorithm design and algorithm analysis techniques.

Course Learning Outcomes (CLO)

Upon successful completion of this course, you will be able to:

- Discuss and analyze a variety of fundamental concepts and practices in the areas of software engineering.
- Apply object-oriented software design methodologies in software programs.
- Use standard abstract data types and data structures, including stacks, queues, and linked lists, trees, in the design of software programs.
- Apply standard algorithmic techniques including recursions, hashing, searching, and sorting, in the design of software programs.
- Use high-level software development tools, including advanced text editors, compilers, linkers, source-level debuggers in implementation and debugging of software programs.

Required Texts/Readings

Textbook

Title: Author: Publisher: ISBN:	**Problem Solving with C++, 10th edition** Walter Savitch Pearson, 2017 978-0134710747
Title: Author: Publisher: ISBN:	**Data Structures Using C++, 2nd edition** D.S. Malik Cengage Learning, 2010 978-0324782011

References

Title: Author: Publisher: ISBN:	**Big C++, 3rd edition** Cay S. Horstmann, Timothy A. Budd Wiley, 2013 978-1118674291
Title: Author: Publisher: ISBN:	**C++ How to Program, 10th edition** Paul Deitel, Harvey Deitel Pearson, 2016 978-0134448237
Title: Author: Publisher: ISBN:	**The C++ Programming Language, 4th edition** Bjarne Stroustrup Addison-Wesley Professional, 2013 978-0321563842
Title: Author: Publisher: ISBN:	**Data Structures and Algorithms in C++, 4th edition** Adam Drozdek Cengage Learning, 2012 978-1133608424

Title:	**Data Structures and Algorithm Analysis in C++, 4th edition**
Author:	Mark A. Weiss
Publisher:	Pearson, 2013
ISBN:	978-0132847377

Software to install

This class will use the GNU C++ compiler. You can use your favorite editor to write your programs and then run the C++ compiler from the command line. However, you are strongly encouraged to use one of the following integrated development environments (IDE) for C++ development on the Mac and Linux platforms:

- Eclipse CDT (C/C++ Development Tooling): https://eclipse.org/cdt/
  Eclipse is the most popular IDE in industry.
- NetBeans C and C++ Development: https://netbeans.org/features/cpp/

GNU C++ is usually pre-installed on the Mac and Linux platforms. If you are on a Mac, you should not use Apple's Xcode development environment for this class, because it may cause you to write programs that will not port to other platforms. For the same reason, if you are on Windows, you should not use Microsoft Visual C++.

Windows users should install VirtualBox (https://www.virtualbox.org/wiki/VirtualBox) and run Linux as a virtual machine. Debian (https://www.debian.org/intro/about) is a good Linux distribution. Download a .iso installation image (https://www.debian.org/distrib/) and install it VirtualBox. Run Debian on the virtual machine and use its pre-installed GNU C++ compiler. Then you can install Eclipse or NetBeans for Linux.

The C++ 2011 standard

We will use the 2011 standard of C++. You must set this explicitly for your project in Eclipse and in NetBeans:

- **Eclipse:** Right-click on your project in the project list at the left side of the window. Select "Properties" from the drop-down context menu. In the left side of the properties window, select "C/C++ Build" ➔ "Settings". In the Settings dialog, select "GCC C++ Compiler" ➔ "Dialect". For "Language standard" select "ISO C++ 11". Click the "Apply" button, answer "Yes", and then click the "OK" button.
- **NetBeans:** Right-click on your project in the project list at the left side of the window. Select "Properties" from the drop-down context menu. In the left side of the properties window, select "Build" ➔ "C++ Compiler". In the table, for "C++ Standard" select "C++11". Click the "Apply" button and then click the "OK" button.

To compile on the command line, include the **`-std=c++11`** option:

```
g++ foo.cpp -std=c++11 -o foo
```

Course Requirements and Assignments

This class will progress rapidly, and you must work hard to keep up. Do not fall behind in the reading. There can be in-class quizzes during to check your understanding, and multiple programming assignments each week.

Each assignment will be worth a specified maximum number of points, depending on difficulty, and it will be due before the start of the next class. No assignment will be accepted after its solution is presented in class (it will get a 0 score).

This is a challenging course that will demand much of your time and effort throughout the semester.

Learning to program in a new language requires much practice. Each week, there will be several short practice programs that emphasize specific language features that you will need to master to complete that week's main programming assignments. All the practice programs and many of the main assignments will be graded automatically online.

The university's syllabus policies:

- University Syllabus Policy S16-9 at http://www.sjsu.edu/senate/docs/S16-9.pdf.
- Office of Graduate and Undergraduate Programs' Syllabus Information web page at http://www.sjsu.edu/gup/syllabusinfo/

The University's Credit Hour Requirement:

"Success in this course is based on the expectation that students will spend, for each unit of credit, a minimum of 45 hours over the length of the course (normally 3 hours per unit per week with 1 of the hours used for lecture) for instruction or preparation/studying or course related activities including but not limited to internships, labs, clinical practica. Other course structures will have equivalent workload expectations as described in the syllabus."

Academic Integrity

You may study together and discuss the assignments, but what you turn in must be your individual work. Assignment submissions will be checked for plagiarism using Moss from the Computer Science Department at Stanford University (http://theory.stanford.edu/~aiken/moss/). See http://www.cs.sjsu.edu/~mak/Moss/ for a report from a Moss run.

Copying code from another student's program or sharing your program code are equal violations of academic integrity. Moss is not fooled by renaming variables, reformatting code, or re-ordering functions.

Violators of academic integrity will suffer severe sanctions, including academic probation. Students who are on academic probation are not eligible for work as instructional assistants in the university or for internships at local companies.

Final Examination

The quizzes, midterm, and final examinations will be closed book. Instant messaging, e-mails, texting, tweeting, file sharing, or any other forms of communication with anyone else during the exams will be strictly forbidden.

There can be no make-up midterm examination unless there is a documented medical emergency. There will be no make-up quizzes. The quizzes are online, and you are responsible for bringing a laptop or mobile device that can connect to the wireless network in the classroom. Make-up final examinations are available only under conditions dictated by University regulations.

Grading Information

This class is graded credit/no credit (CR/NC). Your individual final class grade will be weighted as follows:

50%	Assignments
15%	In-class quizzes
15%	Midterm exam
20%	Final exam

Each assignment and exam will be scored (given points) but not assigned a letter grade. The average score will be posted after each assignment and exam.

During the semester, you can keep track of your progress in Canvas. At the end of the semester, all the students will be ranked in order of their weighted class scores. Students who score above a threshold will receive CR, and the rest will receive NC. We expect at least 75% of students to receive CR.

"All students have the right, within a reasonable time, to know their academic scores, to review their grade-dependent work, and to be provided with explanations for the determination of their course grades." See University Policy F13-1 at http://www.sjsu.edu/senate/docs/F13-1.pdf for more details.

Classroom Protocol

It is very important for each student to attend classes and to participate. Cell phones in silent mode, please.

University Policies

Per University Policy S16-9, university-wide policy information relevant to all courses, such as academic integrity, accommodations, etc. will be available on Office of Graduate and Undergraduate Programs' Syllabus Information web page at http://www.sjsu.edu/gup/syllabusinfo/.

CMPE 180-92 Data Structures and Algorithms in C++

Section 92
Fall 2017

This schedule is subject to change with fair notice which will be communicated through emails and announcements via Canvas and Piazza. Chapter readings are from the required texts.

Course Schedule

Week	Dates	Topics and activities	Savitch	Malik
1	Aug 24	Introduction C++ basics Flow of control Simple input and output (I/O)	1, 2, 3	
2	Aug 31	Procedural abstraction Functions	4, 5	
3	Sep 7	I/O streams Introduction to classes and objects Arrays Strings Vectors	6, 7, 8	
4	Sep 14	Pointers Dynamic arrays	9	
5	Sep 21	Structures Classes Public and private members Constructors Abstract data types (ADT)	10	1, 2, 3
6	Sep 28	Friend functions Operator overloading Classes and dynamic arrays Destructors Separate compilation Namespaces	11, 12	
7	Oct 5	Copy constructors The assignment operator The “Big Three” A “safe” array type Linked lists Stacks Queues	13	
8	Oct 12	***Midterm exam Thursday, March 16*** Sorted linked list Class hierarchies Inheritance Polymorphism Virtual destructors	15	

Week	Dates	Topics and activities	Savitch	Malik
9	Oct 19	Exception handling Templates Standard Template Library (STL) Containers Iterators Linked lists	16, 17, 18	4
10	Oct 26	Recursion Binary search Backtracking	14	6
11	Nov 2	Introduction to algorithm analysis Recurrence relations Proof by induction Big-O notation Rates of growth and scalability Maps Hashing Sets	18.2, 18.3	9
12	Nov 9	Selection sort Insertion sort Shellsort Mergesort Quicksort		10
13	Nov 16	Trees Tree traversals Binary search tree AVL tree		11
14	Nov 30	Graphs Topological sort Unweighted shortest path Weighted least cost path and Dijkstra’s algorithm Minimum spanning tree and Prim’s algorithm Depth-first and breadth-first searches		12
15	Dec 7	Type auto Object-oriented design The unified modeling language (UML) Design patterns		
Final	Thursday Dec 14	Time: 5:15 – 7:30 PM Room: ENG 189		