COMP 1020 - Computing II
3 credits for Lecture & 1 for the Lab
Summer 2019 Syllabus

General Information

Instructor	Dr. Jonathan Mwaura
Office	Dandeneau Hall 214
Email	Jonathan_mwaura@uml.edu
Class Time	M/W/Th 10:30 – 12:50 / 13:30 -15.50 (lab)
Class Location	Dandeneau 321/416 (lab)

Required Textbook

- Data Structures – Zybooks
  - Sign in or create an account at learn.zybooks.com
  - Enter zyBook code: **UMLCOMP1020MwauraSummer2019**
  - Subscribe

Supplementary Textbook / Materials

- Algorithms in C, Third Edition by Robert Sedgewick. (Parts 1-5)
- Data Structures An Advanced Approach Using C, Esakov and Weiss
- Data Structures and Program Design in C, Second Edition by Kruse, Tondo and Leung.

Course Description

Pointers. Lists, stacks and queues. Binary trees, AVL trees, n-ary trees. Advanced sorting via quicksort, heapsort, etc. Characters and strings. Graphs. Advanced file techniques. Recursion. Programming style, documentation, and testing. This course includes extensive laboratory work. Effective Fall 2013, Co-req: Computing 2 lab.

Course Prerequisites

- COMP1010 Computing I and COMP1030L Computing I Lab.

Course Category

- Required

Additional Course Information

- **Purpose** – This course is intended to provide the student with an understanding of hardware and software concepts, program design with data structures, and programming in C.
- **Web Page** – on Blackboard
- **Software** – Visual Studio 2015

Additional Faculty Details

- **Office Location** – Dandeneau Hall 214

Tentative Schedule

	Dates (2019)	**Topic/Activities**
WEEK 01	Jul 08th	Course Introduction Revision: Pointers & Dynamic memory allocation
	Jul 10th	Vectors
	July 11th	Generic (type independent vectors)
WEEK 02	Jul 15th	Linked Lists & Recursion
	Jul 17th	Stacks
	Jul 18th	Queues
WEEK 03	Jul 22nd	Searching & Sorting Algorithms
	Jul 24th	Searching & Sorting Algorithms
	Jul 25th	Revision + (Exam 1)
Week 04	Jul 29th	Trees - Binary Trees / BST
	Jul 31st	Trees – Balanced Trees (AVL)
	Aug 01st	Review
Week 05	Aug 5th	Heaps/ Priority Queues
	Aug 07th	Binomial Queues
	Aug 08th	Review
Week 06	Aug 12th	Hash Maps/ Hash Tables
	Aug 14th	Revision
	Aug 15th	Final Exam

ABET Student Outcomes:

By the conclusion of the course, students must demonstrate:

(a) An ability to apply knowledge of computing and mathematics appropriate to the discipline.

This course will extend the student's knowledge about programming. Data structures, complexity analysis, testing, and good software development practices will be practiced and evaluated.

(b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution.

The presentation of the material will be strongly based on providing a motivation for existing practices and language features. New material will be introduced as a solution to a problem based on the services provided by available computing hardware. Throughout the course, computing will be presented in a framework of:

1. Understanding known tools and their applicability to desired results;

2. Reformulating the problem specification to require a more sophisticated solution, beyond these tools' limitations; and

3. Introducing new tools or facilities that can meet the new requirements.

In this way, computational problems are presented as admitting a wide range of solutions of varying degrees of complexity.

(c) An ability to use current techniques, skills, and tools necessary for computing practice.

Students will cover vectors, linked lists, stacks, queues, trees, graphs, heaps and hash tables all using opaque objects in C.

(d) An ability to apply design and development principles in the construction of software systems of varying complexity.

The practical component of the course will entail implementing solutions to specific problems based on specification documents. As the student's repertoire of programming techniques expands, programming assignments will gradually allow more freedom in implementation details, requiring selection of the most appropriate tools and careful consideration of the tradeoffs involved in selecting one approach over another.

Course Outcomes:

At the completion of this course, students will have demonstrated:

1. An understanding of terminology associated with computing systems. [ABET a]
   Assessment: Exams and/or quizzes
2. The ability to create structured modular programs [ABET b]
   Assessment: Programming exercises and/or exams
3. An understanding of the basics of procedural programming and the concept of generic data structures in C. [ABET a and b]
   Assessment: Programming exercises and/or exams
4. Recognition of the importance and proficiency in the use of good programming style. [ABET c and d]
   Assessment: Programming exercises
5. The ability to use the basic features of a Unix/Linux environment to create, debug, execture and submit programs in C using makefiles. [ABET c]
   Assessment: Programming exercises
6. A demonstrated proficiency with lists, stacks, queues, vectors, trees, and basic graphs as well as an introductory understanding of asymptotic analysis of algorithms. [ABET c, d]
   Assessment: Programming exercises

Methodology

Teaching methods: This course will make use of lectures, examples, and question and answer sessions. Hands-on programming activities using computers during class, in addition to active reading, experimenting with sample programs, and problem solving, will be used during the semester. Independent programming assignments are used to evaluate language and coding skills.

Assessment: There will be in-class exercises, regular problem sets and quizzes, programming exercises, one in-class exam and a comprehensive final exam.

Grading:

The available points for the various assignments in this class are:

Program Average (Daily Programming Exercises.)	35%
Quiz Average (at least 1 quiz each week) + Take home quizzes	15%
Homework Average (All drawn from Zybooks includes participation)	10%
Mid-term Exam	20%
Final Exam	20%

The above comprises your lecture grade. Your lab grade will account for up to 25% of your total score where the remaining score is accounted for using your lecture grade and you will receive the same grade in the lecture as you receive in the lab; the courses are linked.

Final grades will be assigned using the following scale:

Distinction		
A	90.00	100.0
Merit		
B+	87.00	89.00
B	83.00	86.00
B-	80.00	82.00
Credit Pass		
C+	77.00	79.00
C	73.00	76.00
C-	70.00	72.00
Pass		
D+	67.00	69.00
D	60.00	66.00
Fail		
F	59	Less

At the end of the semester, a course average is computed for each student using the percentage weights given above. The scale for assigning letter grades based on the course average is formed by averaging the grading scales of the programs, problem sets, quizzes,

and exams using the weights given above. Factors such as attendance, improvement (especially as demonstrated on the final exam), and class participation may be considered when making the final decision in borderline cases.

Requests for re-grading assignments may be made up to one week after the assignment is returned. The request must be submitted in writing and include a short paragraph outlining the rationale for the re-grade. Acceptable requests include correcting errors in calculating a score, marking a correct answer incorrect, etc.

Course Policies

Attendance: Formal attendance is **NOT** mandatory. It is the student's prerogative to look out for what has been taught in class and regularly check contents on Blackboard. Feedback in terms of whether a student is failing the course shall be given by the middle of the term.

Academic integrity: The practice of good ethical behavior is essential for maintaining good order in the classroom, providing an enriching learning experience for students, and as training as a practicing computing professional upon graduation. This practice is manifested in the University's Academic Integrity policy. Students are expected to strictly avoid academic dishonesty and adhere to the Academic Integrity policy as outlined in the course catalog. Violations will be dealt with as outlined therein.

As a general rule, all work submitted for grading must be the student's own work. Students are allowed to help each other solve compiling and linking problems, and may generally discuss issues related to a student's particular program, but students may not share code, write code, or examine another student's code.

In regard to homework and projects, students may discuss the problems (what is being asked for), appropriate material from class lectures or the textbook or acceptable other sources. Students, however, may not share answers or the specifics of how to answer the question.

Use of material from previous classes, solution manuals, material from the Internet or other sources (e.g., parents, siblings, friends, etc.) that directly bears on the answer is strictly prohibited.

At the discretion of the instructor, students may be asked to sign a statement that they have abided by the University's Academic Integrity policy and its application to this class. This statement may appear on homework, tests, or projects.

When in doubt, consult the course professor before doing something that may result in violation of the University's Academic Integrity policy.

Application to this course:
Programming assignments are to be done by the student alone. No outside help is permitted. If you need help on a programming assignment you can only receive aid from the instructor of the course, lab TAs or approved tutors.

The sanction for the first violation of the Academic Integrity policy or plagiarism policy will result in a minimum failing grade on the relevant assignment and the violation will be reported to the student's department chair. Once the final decision has been rendered and any or all appeals exhausted by any parties involved, the instructor or appropriate parties will carry out the recommended sanction.

Personal conduct: In order to minimize distractions and interruptions, students will be expected to:

1. Arrive to class on-time and fully prepared.

2. Give the instructor full and undivided attention once the lecture has begun.

3. Turn off and stow all cell phones, pagers, and any other personal electronic devices once the lecture has begun.

Failure to adhere to these policies may result in immediate dismissal from class and loss of any in-class credit for relevant assignments or activities.

Computer use: Students are encouraged to use their laptop computer for taking notes and other activities directly related to the course. The manner in which students use the computer in class is considered a matter of honor and professionalism. Students will adhere to the following guidelines:

1. Computer use must be for taking notes or other activities related directly to the course.

2. Computer use must be subtle and must not distract fellow classmates or the instructor.

Inappropriate use of a computer in the classroom will be viewed as disrespectful to the instructor and classmates and will be considered unprofessional. Examples of inappropriate use include, but are not limited to:

- Sending, receiving, or reading e-mail
- Instant messaging
- Web browsing
- Working on assignments
- Playing games
- Listening to music

- Watching movies

Judgments regarding the appropriateness of computer use are at the discretion of the instructor. The consequences for violating this policy are also at the discretion of the instructor, and may include loss of in-class computer privileges, grade reduction, and so forth.

Course readings: Readings are to be completed before the class session for which they have been assigned; material covered in each reading is fair game for class discussions and unannounced quizzes.

Assignments: Assignments will be distributed via a mixture of paper and electronic means. Students are responsible for managing due dates and understanding submission procedures to turn in programming assignments.

Exams: A discussion of the topics covered by each exam will occur during class at most one week from the scheduled exam date.

Final exam: The comprehensive final exam for this class will be on 15th August 2019.

Late work: Late assignments or projects will not be accepted without prior approval. Students must consult the instructor at least three days prior to the scheduled due date of any assignment to make alternative arrangements; however, the instructor is under no obligation to grant any such request. Penalties such as a reduced score may be applied at the instructor's discretion.

In-class exams will not be rescheduled except in the event of an excused absence. The instructor will work with the student to schedule such a makeup exam; however, it is in the student's best interest not to miss the regularly scheduled in-class exam.