BOB JONES UNIVERSITY
Build Faith. Challenge Potential. Follow Christ.

College of Arts & Science

CPS 210

Data Structures

Spring/2020

Professor

Dr. Sarah Gothard

Office Hours:	By appointment only
Recommended appointments:	TTh 8:00 a.m. MWF 9:00 a.m. MWF 7:30 a.m.
Contact Information:	sgothard@bju.edu 937-321-5167 (texting acceptable)

Course Information

CpS 210-1	MWF 8:00-8:50 a.m. in ML 3
CpS 290	Tuesday 10:00-10:50 a.m. in ML 3
Credit/Load	3/3

Textbooks and Resources

Data Structures and Algorithm Analysis in C++ by Clifford Shaffer: Online Print
The C++ Programming Language by Bjarne Stroustrup. Fourth Edition; ISBN: 0-321-56384-0
Online: C++ 11 FAQ
Online: SGI's STL documentation

Catalog Description

Data structures and algorithm analysis. Includes an introduction to an alternate computing platform

Course Context:

Data Structures is a course primarily taken by computer science majors and minors. Hence, its main context is to fulfil the goals and objectives of the Computer Science Program.

Computer Science Program
It is our desire that all students in the Computer Science Major exhibit the ability to:
CS1. Design and implement solutions to practical problems
CS2. Use appropriate technology as a tool to solve problems in various domains
CS3. Create efficient solutions at the appropriate abstraction level

CS7. Demonstrate an understanding of social, professional and ethical considerations related to computing
CS8. Demonstrate understanding of fundamental concepts in the student's discipline

Course Goal:

The goals of this course are to

- improve your knowledge and experience with
    - the C/C++ programming languages,
    - the Standard Template Library, and
    - the Linux operating system
- increase your knowledge of computer science, specifically in these areas:
    - Data structures -- both fundamental and advanced;
    - Recursive algorithms, greedy algorithms, and dynamic programming;
    - Depth-First and Breadth-First Searching
- increase your awareness of social issues involving computers including privacy and civil liberties.

Course Objectives:

The student will be able to

1. Describe how the data structures in the topic list are allocated and used in memory. *Evaluated in test 1.*
2. Describe common applications for each data structure in the topic list. *Evaluated in multiple tests especially test 2.*
3. Implement the user-defined data structures in a high-level language. *Evaluated in labs 7, 10, & 11; program 4.*
4. Write programs that use each of the following data structures: arrays, records, strings, linked lists, stacks, queues, and hash tables. *Evaluated in programs 1, 2, 3 & 4.*
5. Compare and contrast the costs and benefits of dynamic and static data structure implementations. *Evaluated in test 2*
6. Choose the appropriate data structure for modeling a given problem. *Evaluated in test 3 and the final.*
7. Describe the concept of recursion and give examples of its use. *Evaluated in test 2.*
8. Identify the base case and the general case of a recursively defined problem. *Evaluated in test 2.*
9. Describe the divide-and-conquer approach. *Evaluated in test 3.*
10. Implement, test, and debug simple recursive functions and procedures. *Evaluated in labs 2, 3 & 4.*
11. Describe how recursion can be implemented using a stack. *Evaluated in lab 2.*
12. Determine when a recursive solution is appropriate for a problem. *Evaluated in test 2 and final.*
13. Explain the use of big O, omega, and theta notation to describe the amount of work done by an algorithm. *Evaluated in all chapter tests esp. test 1.*
14. Use big O, omega, and theta notation to give asymptotic upper, lower, and tight bounds on time and space complexity of algorithms.

15. Determine the time and space complexity of simple algorithms. *Evaluated in all chapter tests esp. test 1.*
16. Implement a greedy algorithm to solve an appropriate problem. *Evaluated in program 4 and lab 9.*
17. Implement a divide-and-conquer algorithm to solve an appropriate problem. *Evaluated in program 3.*
18. Solve problems using the fundamental graph algorithms, including depth-first and breadth-first search, single-source and all-pairs shortest paths, transitive closure, topological sort, and at least one minimum spanning tree algorithm. *Evaluated in programs 2, 3, & 4.*
19. Demonstrate the following capabilities: to evaluate algorithms, to select from a range of possible options, to provide justification for that selection, and to implement the algorithm in programming context. *Evaluated in the Object Selection Project.* **Note: this assessment is also used in the evaluation of the CpS program.**

Course Requirements:

A 10-point grading scale, with 1-digit rounding, will be used. The grade for this class will be based upon the following categories:

Category	Points	Description
Labs	180	Twelve labs worth 15 points each are scheduled.
Object Selection Project	120	Has three parts worth 20%, 40%, and 40%
Programs	200	Four programs worth 50 points each will be assigned
Tests	420	Four tests are scheduled. Each one will cover one or more chapters.
Final Exam	100	Emphasizes the application of the knowledge gained during the semester

General Policies:

Class Deportment

Compliance with student handbook policies is expected during class. All class deportment should reflect your intention to pay attention, to be polite, and to be professional. Laptops may be used to take notes and to perform calculations and constructions during class. Please do not use the laptop for other purposes during class since studies have demonstrated that one's student's misuse of a laptop during class tends to diminish the learning of the surrounding students.

Student Responsibilities

- Students are responsible for reading all posted material and acting accordingly.
- Students are responsible for tracking their grades and working early to improve bad averages.
- Students are responsible for notifying the instructor promptly about grade discrepancies.
- Students are responsible for finding a way to keep up with the notes in class. The notes of this class may not be disseminated beyond students in this class with this instructor this semester.

Accommodations for Students with Disabilities

Any student with disabilities or any additional needs is encouraged to contact the instructor within the first week of the course to discuss accommodations that may be necessary.

Attendance Policies and Academic Penalty for Absences

- Attendance will be tracked and reported according to the university attendance policy:
    - Students are expected to attend and arrive on time for all scheduled class sessions, including the final exam.
    - Students are to use effective time management in order to meet their class attendance responsibilities.
    - Up to three (3) personal Absences may be taken for funerals, for sickness, for doctor's or dentist's appointments, for visits and interviews at graduate schools or for interviews for future employment.
    - Up to four (4) Service Absences may be taken to attend approved academic functions or conferences, approved Christian service projects, required military duty or as part of an intercollegiate athletic team. However, students who exceed the Personal Absence limit due to a chronic illness are not eligible to participate in events that require Services Absences. Also, students who are on any type of academic restriction (including probation) or who have a current grade report with a cumulative GPA below 2.0 are not eligible to participate in events that require Service Absences.
    - Arriving late or leaving early is marked as a partial attendance. Three (3) partial attendance marks count as a personal absence.
    - Missing more than 15 minutes of class is marked as an absence.
    - For more details and information about chronic illness, please see the Class Attendance Policy on the BJU Policies page.
- Students are responsible for all material and announcements given in class.
- If a student is absent for an exam and has a good reason, the student is to notify the instructor before the exam is covered in the next class.

Late Work

Work is due at the specified deadline. Late work is accepted at the instructor's discretion. Notify the instructor immediately if a situation arises necessitating an extension. Early, impressive work is encouraged and may be rewarded.

Academic Honesty

You are expected to uphold the school standard of conduct relating to academic honesty:

- School-wide – The link can be found on the BJU Policies page.
- CpS Department clarifications – What is allowed/disallowed in code submissions.

You must assume full responsibility for the content and integrity of the academic work you submit. The guiding principle of academic integrity is that your submitted work; examinations, reports, and projects must be your own work. You are guilty of violating this policy if you

- Represent the work of others as your own.
- Use or obtain unauthorized assistance in any academic work.

- Give unauthorized assistance to other students.
- Modify, without instructor approval, an examination, paper, record, or report for the purpose of obtaining additional credit.
- Misrepresent the content of submitted work.

Misrepresenting your work is unethical in any setting. In an academic setting, it is a breach of the university policies. The penalty for cheating is severe. Any student cheating is subject to receive a failing grade for the assignment and will be reported to the Dean. If you are unclear about whether a particular situation may constitute cheating, consult with your instructor about the situation. For this class, it is permissible to assist classmates in general discussions of construction techniques. General advice and interaction are encouraged. Each of you must develop your own solutions to the assigned projects, assignments, and tasks. In other words, you may not "work together" on graded assignments with other students unless instructed to work as a group on a particular assignment. Such collaboration constitutes cheating. You may not use or copy (by any means) another's work (or portions of it) and represent it as your own.

Learning how to use sources appropriately is a vital part of your development as a student. To assist you in this endeavor, the university uses Turnitin, an academic plagiarism checker. Registration in this course constitutes permission for the teacher to submit any or all assignments to Turnitin.

Need Help?

You must seek help when needed because you are the only one who knows when you need it. If you need help, reach out to one of the following ways:

- Teacher – It is always best to seek help in person, either around class time or by appointment. You may also text me or email me for help.
- Classmates – Studying for tests with other students is helpful. You may work together on the labs. The four programming assignments are to be completed individually. The Academic Honesty CS-specific link above describes the help allowed for programs.

Copyright Policy

TENTATIVE SCHEDULE				
DSA: *Data Structures and Algorithm Analysis* text | C++: *The C++ Programming Language* text				
Date	**Day**	**Topic**	**Reading**	**Assignment Due**
15-Jan	W	Abstraction Mechanisms	C++: Chapter 3	
17-Jan	F	Containers and Algorithms	C++: Chapter 4	
20-Jan	M	Martin Luther King Jr. Day – No Class		
21-Jan	T	Linux basics and Radix Sort	Lab 1	
22-Jan	W	Pointers in C++; Iterators	C++: Chapters 7, 33	
24-Jan	F	Memory and Resources	C++: Chapter 34	
27-Jan	M	STL Containers, esp. vector	C++: Chapter 31	Lab 1

28-Jan	T	Mergesort	Lab 2	
29-Jan	W	Complexity Analysis	DSA: Chapter 3	
31-Jan	F	STL unordered_map and algorithms	C++: Chapters 31, 32	
3-Feb	M	BJU Online test		Lab 2
4-Feb	T	Magic Squares	Lab 3	
5-Feb	W	Lists using linking	DSA: Chapter 4	
7-Feb	F	Deques, array-based lists, etc.	WP: Doubled-ended queue	Prog 1
10-Feb	M	Container adapters	C++: Section 31.5	Lab 3
11-Feb	T	Fast exp. And Fibonacci numbers	Lab 4	
12-Feb	W	Recursion – replacement		
14-Feb	F	Usages of stacks and queues; P2		Prog 2
17-Feb	M	Programming thoughts	WP: Master theorem	Lab 4
18-Feb	T	Infix to postfix	Lab 5	
19-Feb – 21-Feb	Bible Conference – No Classes			
24-Feb	M	Recursion – backtracking		
25-Feb	T	Debugging Lab		
26-Feb	W	Trees	DSA: Sections 6.1-6.4	Lab 5
28-Feb	F	Test on lists, stacks, queues, and deques		
2-Mar	M	Heaps	DSA: Section 6.17	Debugging Lab
3-Mar	T	Expression trees	Lab 6	
4-Mar	W	Heap implementations		
6-Mar	F	Heap alternatives		
9-Mar	M	Binary search trees	DSA: Section 6.8, 6.11	Lab 6
10-Mar	T	Heap applications	DSA: Section 6.18, Lab 7	
11-Mar	W	Insertion and deletion		
13-Mar	F	Balancing	WP: Tree rotation	
16-Mar	M	Red-black trees; B+ trees	WP: Red-black trees	Lab 7
17-Mar	T	Optimal Binary Search Trees	Lab 8, Prog 3 intro	
18-Mar	W	Test		
20-Mar	F	Huffman compression	DSA: Sections 13.1-13.2	
23-Mar – 27-Mar	Spring Break – No classes			
30-Mar	M	Graphs and their representations	DSA: Section 13.3	Lab 8
31-Mar	T	Partitions and disjoint sets	DSA: Section 12.2, Lab 9	
1-Apr	W	Graph traversals	DSA: Section 13.4	
3-Apr	F	Topological Sort	DSA: Section 13.5	
6-Apr	M	Minimum Weight Spanning Trees	DSA: Section 13.6	Lab 9
7-Apr	T	Shortest paths	Lab 10	
8-Apr	W	Multilists		OS1
10-Apr	F	Matching		Prog 3
13-Apr	M	Network flows		Lab 10
14-Apr	T	Prims	Lab 11; Prog 4 intro	
15-Apr	W	Test		
17-Apr	F	Shell sort, quicksort, heap sort	DSA: Secs 7.8, 7.11, 7.12	

20-Apr	M	Sorting Bounds	DSA: Section 7.15, Lab 12	Lab 11
21-Apr	T	Empirical comparison of sorting algorithms	DSA: Section 7.16	
22-Apr	W	Hash Functions	DSA: Section 9.1-9.3	Prog 4
24-Apr	F	Open hashing	DSA: Section 9.4	OS 2
27-Apr	M	Closed hashing		
28-Apr	T	Object selection test	DSA: Section 9.5-9.7	OS 3
29-Apr	W	Extra class		Lab 12
1-May	F	Review		
4-May – 7 May	Exams			