Columbia College
Vancouver, Canada

Course Outline			
Term: Summer 2025	Course No: CSCI 125	Course Credits: 3	
Instructor: Jetic Gū	Course Section No: 17	Total Hours: 5	Total Weeks: 13
Instructor Office: Room No. 544 Main Campus	Course Title: Introduction to Digital and Computer Systems Design	Class Room No. 340	
Instructor Email: jgu@columbiacollege.ca			
Class Meeting Days/Time: TF: 16:00 - 17:50; W: 17:00-17:50			
Instructor Office Hours: MT: 14:00-16:00	Course Format: In person delivery		
Course Prerequisites Computer Science 120	Course Corequisites English 099		
Transferability to: visit bctransferguide.ca			

Course Description:

A rigorous introduction to computing science and computer programming, suitable for students who already have some background in computing science and programming. Students will learn the fundamental concepts of computing science and develop basic skills in software development.

Topics include: history of computing science; review of elementary programming; data types and control structures; fundamental algorithms; abstract data types; elementary data structures; basic object-oriented programming and software design; elements of empirical and theoretical algorithmic; computability and complexity; design, specification and program correctness.

Additional Course Details:

Required Texts/Readings/Learning Resources:

Textbook: *Problem Solving with C++,* 10/E, Walter Savitch, ISBN-10: 0133591743 • ISBN- 13: 9780133591743 ©2018 Addison-Wesley

Course Learning Outcomes:

Upon successful completion of this course the student will be able to:

1. Given a code fragment, describe its purpose in plain English, and trace its execution.
2. Adapt an existing code fragment to change its behaviour.
3. Define the term "pseudocode" and the term "algorithm".
4. List various desirable properties of an algorithm.
5. Use pseudocode and/or diagrams to describe the steps involved in solving simple problems.
6. Given a simple problem, create an algorithm to solve it.
7. Modify conditional structures in a short program.
8. Modify iterative structures in a short program.
9. Write well-structured, well-documented, well-commented readable code.
10. Describe the role of documentation and comments.
11. Use language-appropriate idioms and write meaningful, well-structured external documentation.
12. Design, implement, evaluate, and remove errors from a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, basic conditional and iterative structures, and functions.
13. Describe the syntax and semantics of conditional structures available in [a language].
14. Use conditional structures available in [a language].
15. Choose proper conditional and/or iterative constructs for a given programming task and justify your choice.
16. Describe the syntax and semantics of iteration structures available in a language.
17. Use iterative structures available in a language.
18. Apply decomposition techniques to break a program into smaller pieces (where each piece has a specific purpose or responsibility).
19. Explain the role of pseudocode and diagramming in decomposing problems.
20. Define the term "formal parameter" and the term "actual parameter".
21. Find function and given a code fragment, identify formal and actual parameters of a function.
22. Describe the role of formal and actual parameters of a function.
23. Trace the execution of a program (e.g., desk checking).
24. Describe and use strategies for removing syntax errors, logic errors, and runtime errors.
25. Interpret error messages (e.g. compiler, run-time) and identify their causes.
26. Define the term "iteration" and the term "recursion".

27. Recognise algorithms as being iterative or recursive.
28. Define the term "loop invariant".
29. Use pointers/references in [an imperative language].
30. Describe the advantages and disadvantages of using pointers/references.
31. Describe the risks of using pointers/references (e.g., dangling pointers, memory leaks).
32. Implement list data structures using both index-based and reference/pointer techniques.
33. Define "time complexity" and "space complexity".
34. Given a code fragment, find and derive its time and/or space complexity.
35. Compare and contrast code fragments based on their time and/or space complexity.
36. Define "big O", "big Omega", and "big Theta".
37. Compare and contrast big O, big Omega, and big Theta notations.
38. Use complexity to estimate the time taken to execute code fragments.
39. Explain the differences between best-, worst-, and average-case complexity analysis.
40. Describe why best-case complexity analysis is rarely relevant and how worst-case complexity analysis may never be encountered in practice.
41. Given a list and a target, explain how the sequential search attempts to find the target.
42. Recall and derive the Big O value for a sequential search.
43. Given an algorithm, compute its worst-case asymptotic complexity.
44. Define the term "abstraction" and the term "implementation".
45. Differentiate between an abstraction and an implementation.
46. Describe list data structures along with their public-interface specifications.

Course Content/Schedule*

Week	Topic(s)	Readings	Assessments	Briefly describe list (via number) the outcomes linked to the assessments.
1	Introduction to CS and Programming; Environment setup	Lecture notes		1
2	Introduction to your coding enviroment; Your first C/C++ programme	Lecture notes	Lab 0 due;	1, 2
3	Your first C/C++ programme; Conditions and Loops	Lecture notes		2, 3
4	Functions	Lecture notes	Assignment 1 due;	2, 3
5	Functions; Array	Lecture notes	Quiz 1;	2, 3, 4
6	Pointers	Lecture notes	Assignment 2 due;	2, 3, 4, 5
7	Review - Midterm	Lecture notes	Midterm;	
8	Algorithms and Complexity	All covered content		3, 4
9	Algorithms and Complexity	Lecture notes	Assignment 3 due;	3, 4
10	Data Structure	Lecture notes	Quiz 2;	3, 4
11	Class, Object Oriented Programming	Lecture notes	Assignment 4 due;	2, 4, 5
12	Class, Object Oriented Programming	Lecture notes	Assignment 5 due;	2, 4, 5
13	FINAL EXAM			

*Timing subject to change

Evaluation Criteria

Evaluation Methods	%	Comments
Quiz	10	
Assignments + Labs	30	
Midterm	20	
Final	40	
Total	**100%**	

Classroom Code of Conduct:

Students will be prepared for any appointments with the instructor or other students – this means logging in and getting out paper, pens, necessary texts and so on before the appointment starts.

1. Students will communicate respectfully when interacting with the instructor or classmates.
2. Students will respectfully communicate with the instructor and classmates in discussion groups, office hours, and in any type of electronic communication.
3. Students will respond to messages/emails from the instructor or other classmates in a timely manner.

Cheating and Plagiarism Policy:

I expect all students to uphold the principle of academic honesty. Cheating and plagiarism (presenting another person's words or ideas as one's own) are not acceptable behaviour at anywhere. Depending on the severity of the offence such acts can result in a grade of zero on the test or assignment, a failing grade (F) in the course, or expulsion. In all cases, the circumstances and the penalty are recorded in the student's file. **Do not share your files with others. Do not let others copy or mimic your files. You may take inspiration, but any work you do must be original.** Failure to comply will result in plagiarism charges to both the party providing assistances, as well as the party receiving.

Academic misconduct not covered in the College's Cheating and Plagiarism Policy, is covered under Academic Policy 2.6 Academic Misconduct. It can be found at the following link: https://www.columbiacollege.ca/about/college-policies/ . You are expected to familiarise yourself with this policy, as it covers serious issues including uploading copyright material, submission of falsified records and other strategies to gain unfair academic advantage. If you are unclear on the contents, please ask for clarification.

Course-Specific Policies:

1. Minimum Final Exam and Lab Grades Policy

Students must achieve 50% in Labs, 50% in the Final exam, and 50% in overall grade to pass the course.

2. Late Submission / Resubmission Policy

If you are affected by personal issues such as sickness, injuries, the passing of a relative, or other traumatising experiences, you should contact an advisor and seek professional help and your instructor will try to accommodate as much as possible. Otherwise, late submissions and resubmission are not allowed beyond the original due.

Grading System

Grade Percentage	Grade Points	Rating
A+ 90-100	4.3	Excellent
A 85-89	4.0	
A - 80-84	3.7	Very Good
B+ 76-79	3.3	
B 72-75	3.0	
B - 68-71	2.7	Good
C+ 64-67	2.3	

C	60-63	2.0	Satisfactory
C-	55-59	1.7	
D	50-54	1.0	Marginal Pass
F	0-49	0.0	Fail
N	Below 50	0.0	Failure for non-completion or non-attendance

Please see the college calendar for more information about grading and related policies.