Recep Tayyip Erdogan University

Faculty of Engineering and Architecture

Computer Engineering

CE100- Algorithms and Programming II

Syllabus

Spring Semester, 2021-2022

Instructor	**Asst. Prof. Dr. Uğur CORUH**
Contact Information	ugur.coruh@erdogan.edu.tr
Office Number	**F-301**
Google Classroom Code	**bafwwt6**
Lecture Hours and Days	TBD
Lecture Classroom	TBD
Office hours	Meetings will be scheduled over Google Meet with your university account and email and performed via demand emails. Please send emails with the subject starts with [CE100] tag for the fast response and write formal, clear, and short emails.
Lecture and Communication Language	English
Theory/Laboratory Course Hour Per Week	3/2 hours
Credit	4
Prerequisite	CE103- Algorithms and Programming I
Corequisite	TBD
Requirement	TBD

**TBD: To Be Defined.*

A. Course Description

This course is a continuation of the ***Algorithms and Programming I*** course. In this course learned programming skills in Algorithms and Programming I course met with common problems and their solution algorithms. This lecture is about analyzing and understanding how algorithms work for common issues. The class will be based on expertise sharing and guiding students to find learning methods and practice for algorithm and programming topics. By making programming applications and projects in the courses, the learning process will be strengthened by practicing rather than theory.

B. Course Learning Outcomes

After completing this course satisfactorily, a student will be able to:

- Interpret a computational problem specification and algorithmic solution and implement a C/C++, Java or C# application to solve that problem.
- Argue the correctness of algorithms using inductive proofs and invariants.
- Understand algorithm design steps
- Argue algorithm cost calculation for time complexity and asymptotic notation
- Analyze recursive algorithms complexity
- Understand divide-and-conquer, dynamic programming and greedy approaches.
- Understand graphs and graph related algorithms.
- Understand hashing and encryption operations input and outputs.

C. Course Topics

- Algorithms Basics, Pseudocode
- Algorithms Analysis for Time Complexity and Asymptotic Notation
- Sorting Problems (Insertion and Merge Sorts)
- Recursive Algorithms
- Divide-and-Conquer Analysis (Merge Sort, Binary Search)
- Matrix Multiplication Problem
- Quicksort Analysis
- Heaps, Heap Sort and Priority Queues
- Linked Lists, Radix Sort and Counting Sort
- Convex Hull
- Dynamic Programming
- Greedy Algorithms
- Graphs and Graphs Search Algorithms
    - Breadth-First Search
    - Depth-First Search and Topological Sort
- Graph Structure Algorithms
    - Strongly Connected Components
    - Minimum Spanning Tree
- Disjoint Set Operations
- Single Source Shortest Path Algorithm
- Q-Learning Shortest Path Implementation
- Network Flow and Applications
- Hashing and Encryption

D. Textbooks and Required Hardware

This course does not require a coursebook. If necessary, you can use the following books and open-source online resources.

- *Paul Deitel and Harvey Deitel. 2012. C How to Program (7th. ed.). Prentice Hall Press, USA.*
- *Intro to Java Programming, Comprehensive Version (10th Edition) 10th Edition by Y. Daniel Liang*
- *Introduction to Algorithms, Third Edition By Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein*
- *Problem Solving and Program Design in C, J.R. Hanly, and E.B. Koffman, 6th Edition.*
- *Robert Sedgewick and Kevin Wayne. 2011. Algorithms (4th. ed.). Addison-Wesley Professional.*
- *Harvey M. Deitel and Paul J. Deitel. 2001. Java How to Program (4th. ed.). Prentice Hall PTR, USA.*
- *Paul Deitel and Harvey Deitel. 2016. Visual C# How to Program (6th. ed.). Pearson.*
- *Additional Books TBD*

During this course, you should have a laptop for programming practices. You will have your development environment, and you will use this for examination and assignments also classroom practices.

E. Grading System

Midterm and Final grades will be calculated with the weighted average of the project or homework-based examinations. Midterm grades will be calculated between term beginning to the midterm week, and Final grades will be calculated between Midterm and Final week homeworks or projects as follow.

$$a_n = Homework\ or\ Project\ Weight$$

$$HW_n = Homework\ or\ Project\ Points$$

$$n = Number\ of\ Homework\ or\ Project$$

$$Grade = \frac{(a_1HW_1 + a_2HW_2 + \cdots + a_nHW_n)}{n}$$

Homework/Exam	Weight
$Midterm$	$\%40$
$Final$	$\%60$

$$Passing\ Grade = \frac{40 * Midterm_{Grade} + 60 * Final_{Grade}}{100}$$

F. Instructional Strategies and Methods

The basic teaching method of this course will be planned to be face-to-face in the classroom, and support resources, homeworks, and announcements will be shared over google classroom. Students are expected to be in the university. This responsibility is very important to complete this course with success. If pandemic situation changes and distance education is required during this course, this course will be done using synchronous and asynchronous distance education methods. In this scenario, students are expected to be in the online platform, zoom, or meet at the time specified in the course schedule. Attendance will be taken.

G. Late Homework

Throughout the semester, assignments must be submitted as specified by the announced deadline. Your grade will be reduced by 10% of the full points for each calendar day for overdue assignments.

Overdue assignments will not be accepted after three (3) days.

Unexpected situations must be reported to the instructor for late homeworks by students.

H. Course Platform and Communication

Google Classroom will be used as a course learning management system. All electronic resources and announcements about the course will be shared on this platform. It is very important to check the course page daily, access the necessary resources and announcements, and communicate with the instructor as you needed to complete the course with success.

I. Academic Integrity, Plagiarism & Cheating

Academic Integrity is one of the most important principles of RTEÜ University. Anyone who breaches the principles of academic honesty is severely punished.

It is natural to interact with classmates and others to "study together". It may also be the case where a student asks to help from someone else, paid or unpaid, better understand a difficult topic or a whole course. However, what is the borderline between "studying together" or "taking private lessons" and "academic dishonesty"? When is it plagiarism, when is it cheating?

It is obvious that looking at another student's paper or any source other than what is allowed during the exam is cheating and will be punished. However, it is known that many students come to university with very little experience concerning what is acceptable and what counts as "copying", especially for assignments.

The following are attempted as guidelines for the Faculty of Engineering and Architecture students to highlight the philosophy of academic honesty for assignments for which the student will be graded. Should a situation arise which is not described below, the student is advised to ask the instructor or assistant of the course whether what they intend to do would remain within the framework of academic honesty or not.

a. What is acceptable when preparing an assignment?

- Communicating with classmates about the assignment to understand it better
- Putting ideas, quotes, paragraphs, small pieces of code (snippets) that you find online or elsewhere into your assignment, provided that
    - these are not themselves the whole solution to the assignment,

    - you cite the origins of these
- Asking sources for help in guiding you for the English language content of your assignment.
- Sharing small pieces of your assignment in the classroom to create a class discussion on some controversial topics.
- Turning to the web or elsewhere for instructions, for references, and solutions to technical difficulties, but not for direct answers to the assignment
- Discussing solutions to assignments with others using diagrams or summarized statements but not actual text or code.
- Working with (and even paying) a tutor to help you with the course, provided the tutor does not do your assignment for you.

b. What is not acceptable?

    - Asking a classmate to see their solution to a problem before submitting your own.
- Failing to cite the origins of any text (or code for programming courses) that you discover outside of the course's lessons and integrate into your work
    - Giving or showing a classmate your solution to a problem when the classmate is struggling to solve it.

J. Expectations

You are expected to attend classes on time by completing weekly course requirements (readings and assignments) during the semester. The main communication channel between the instructor and the students will be emailed. Please send your questions to the instructor's email address about the course via the email address provided to you by the university. ***Ensure that you include the course name in the subject field of your message and your name in the text field***. In addition, the instructor will contact you via email if necessary. For this reason, it is very important to check your email address every day for healthy communication.

K. Lecture Content and Syllabus Updates

If deemed necessary, changes in the lecture content or course schedule can be made. If any changes are made in the scope of this document, the instructor will inform you about this.

Course Schedule Overview

Weeks	Dates	Subjects	Other Tasks
Week 1	TBD	Course Plan and Communication Grading System, Assignments and Exams. Algorithms Basics, Pseudocode Algorithm Cost Calculation for Time Complexity. Worst, Average and Best Case Summary Sorting Problem (Insertion and Merge Sort Analysis)	TBD
Week 2	TBD	Solving Recurrences (Recursion Tree, Master Method and Back-Substitution)	TBD

		Divide-and-Conquer Analysis (Merge Sort, Binary Search) Recurrence Solution	
Week 3	TBD	RAM (Random Access Machine Model) Asymptotic Notation (Big O, Big Teta, Big Omega,Small o,Small omega) Matrix Multiplication (Traditional,Recursive,Strassen)	TBD
Week 4	TBD	Quicksort and Analysis (Hoare and Lomuto Partitioning, Recursive Sorting) Randomized Quicksort and Selection (Recursive,Medians) Heaps (Max / Min Heap, Heap Data Structure, Iterative and Recursive Heapify, Extract-Max, Build Heap) Heap Sort, Priority Queues, Linked Lists, Radix Sort,Counting Sort	TBD
Week 5	TBD	Convex Hull (Divide & Conquer) Dynamic Programming (Fibonacci Numbers) Divide-and-Conquer (DAC) vs Dynamic Programming (DP) Development of a DP Algorithms Matrix-Chain Multiplication and Analysis	TBD
Week 6	TBD	Elements of Dynamic Programming Recursive Matrix Chain Order Memoization (Top-Down Approach, RMC, MemoizedMatrixChain, LookupC) Dynamic Programming vs Memoization Longest Common Subsequence (LCS) Most Common Dynamic Programming Interview Questions	TBD
Week 7	TBD	Greedy Algorithms and Dynamic Programming Differences Greedy Algorithms (Activity Selection Problem, Knapsack Problems)	TBD
Week 8	**20.11.2021** **28.11.2021**	**Midterm**	TBD
Week 9	TBD	Heap Data Structure Heap Sort Huffman Coding	TBD
Week 10	TBD	Introduction to Graphs Graphs and Representation BFS (Breath-First Search) DFS (Depth-First Search) Topological Order	TBD

		SCC (Strongly Connected Components) MST Prim Kruskal	
Week 11	TBD	Disjoint Sets and Kruskal Relationships Single-Source Shortest Paths (Bellman-Ford,Dijkstra) Q-Learning Shortest Path Max-Flow Min-Cut (Ford-Fulkerson,Edmond's Karp,Dinic)	TBD
Week 12	TBD	Crypto++ Library Usage Hashing and Integrity Control Cryptographic Hash Functions (SHA-1,SHA-256,SHA-512,H-MAC) Checksums(MD5,CRC32)	TBD
Week 13	TBD	Symmetric Encryption Algorithms (AES, DES, TDES) Symmetric Encryption Modes (ECB, CBC) Asymmetric Encryption Key Pairs (Public-Private Key Pairs) Signature Generation and Validation	TBD
Week 14	TBD	OTP Calculation(Time-based, Counter-based) File Encryption and Decryption and Integrity Control Operations	TBD
Week 15	TBD	Review	TBD
Week 16	**17.01.2022 30.01.2022**	**Final**	TBD

Course Schedule Details

Week-1

1. Course Plan and Communication
2. Grading System, Home works, and Exams.
3. Algorithms
    a. Algorithm Basics
    b. Introduction to Analysis of Algorithms
        i. Algorithm Basics
        ii. Flowgorithm
        iii. Pseudocode
        iv. Sorting Problem
        v. Insertion Sort Analysis
        vi. Algorithm Cost Calculation for Time Complexity
        vii. Worst, Average, and Best Case Summary
        viii. Merge Sort Analysis

Week-2

1. Solving Recurrences
    a. Recursion Tree
    b. Master Method
    c. Back-Substitution
2. Divide-and-Conquer Analysis
    a. Merge Sort
    b. Binary Search
    c. Merge Sort Analysis
    d. Complexity
3. Recurrence Solution

Week-3

1. RAM (Random Access Machine Model)
2. Asymptotic Notation
    a. Big O Notation
    b. Big Teta Notation
    c. Big Omega Notation
    d. Small o Notation
    e. Small omega Notation
3. Matrix Multiplication
    a. Traditional
    b. Recursive
    c. Strassen

Week-4

1. Quicksort
    a. Hoare Partitioning
    b. Lomuto Partitioning
    c. Recursive Sorting
2. Quicksort Analysis
3. Randomized Quicksort
4. Randomized Selection
    a. Recursive
    b. Medians
5. Heaps
    a. Max / Min Heap
    b. Heap Data Structure
    c. Heapify
        i. Iterative
        ii. Recursive
    d. Extract-Max
    e. Build Heap
6. Heap Sort
7. Priority Queues
8. Linked Lists
9. Radix Sort
10. Counting Sort

Week-5

1. Convex Hull (Divide & Conquer)
2. Dynamic Programming
    a. Introduction
        i. Divide-and-Conquer (DAC) vs Dynamic Programming (DP)
        ii. Fibonacci Numbers
            1. Recursive Solution
            2. Bottom-Up Solution
        iii. Optimization Problems
        iv. Development of a DP Algorithms
    b. Matrix-Chain Multiplication
        i. Matrix Multiplication and Row Columns Definitions
        ii. Cost of Multiplication Operations (pxqxr)
        iii. Counting the Number of Parenthesizations
        iv. The Structure of Optimal Parenthesization
            1. Characterize the structure of an optimal solution
            2. A Recursive Solution
                a. Direct Recursion Inefficiency.
            3. Computing the optimal Cost of Matrix-Chain Multiplication
            4. Bottom-up Computation
        v. Algorithm for Computing the Optimal Costs
            1. MATRIX-CHAIN-ORDER
        vi. Construction and Optimal Solution
            1. MATRIX-CHAIN-MULTIPLY
        vii. Summary

Week-6

1. Elements of Dynamic Programming
    a. Optimal Substructure
    b. Overlapping Subproblems
2. Recursive Matrix Chain Order Memoization
    a. Top-Down Approach
    b. RMC
    c. MemoizedMatrixChain
        i. LookupC
    d. Dynamic Programming vs Memoization Summary
3. Dynamic Programming
    a. Problem-2 : Longest Common Subsequence
        i. Definitions
        ii. LCS Problem
        iii. Notations
        iv. Optimal Substructure of LCS
            1. Proof Case-1
            2. Proof Case-2
            3. Proof Case-3
        v. A recursive solution to subproblems (inefficient)
        vi. Computing the length of and LCS
            1. LCS Data Structure for DP
            2. Bottom-Up Computation
        vii. Constructing and LCS

            1. PRINT-LCS
            2. Back-pointer space optimization for LCS length
4. Most Common Dynamic Programming Interview Questions
    a. Problem-1: Longest Increasing Subsequence
        i. https://www.geeksforgeeks.org/longest-increasing-subsequence-dp-3/
        ii. https://en.wikipedia.org/wiki/Longest_increasing_subsequence#:~:text=In%20computer%20science%2C%20the%20longest,not%20necessarily%20contiguous%2C%20or%20unique.
        iii. https://www.youtube.com/watch?v=22s1xxRvy28&ab_channel=StableSort
    b. Problem-2: Edit Distance
        i. https://www.geeksforgeeks.org/edit-distance-dp-5/
        ii. https://www.youtube.com/watch?v=tU2f2JwHmfQ&feature=youtu.be&ab_channel=PrepForTech
        iii. Recursive
            1. https://www.youtube.com/watch?v=8Q2IEIY2pDU&ab_channel=BenLangmead
        iv. DP
            1. https://www.youtube.com/watch?v=0KzWq118UNI&ab_channel=BenLangmead
            2. https://www.youtube.com/watch?v=eAVGRWSryGo&ab_channel=BenLangmead
    c. Problem-3: Partition a set into two subsets such that the difference of subset sums is minimum
        i. https://www.geeksforgeeks.org/partition-a-set-into-two-subsets-such-that-the-difference-of-subset-sums-is-minimum/
    d. Problem-4: Count number of ways to cover a distance
        i. https://www.geeksforgeeks.org/count-number-of-ways-to-cover-a-distance/
    e. Problem-5: Find the longest path in a matrix with given constraints
        i. https://www.geeksforgeeks.org/find-the-longest-path-in-a-matrix-with-given-constraints/
    f. Problem-6: Subset Sum Problem
        i. https://www.geeksforgeeks.org/subset-sum-problem-dp-25/
    g. Problem-7: Optimal Strategy for a Game
        i. https://www.geeksforgeeks.org/optimal-strategy-for-a-game-dp-31/
    h. Problem-8: 0-1 Knapsack Problem
        i. https://www.geeksforgeeks.org/0-1-knapsack-problem-dp-10/
    i. Problem-9: Boolean Parenthesization Problem
        i. https://www.geeksforgeeks.org/boolean-parenthesization-problem-dp-37/
    j. Problem-10: Shortest Common Supersequence
        i. https://www.geeksforgeeks.org/shortest-common-supersequence/
        ii. https://en.wikipedia.org/wiki/Shortest_common_supersequence_problem
    k. Problem-11: Partition Problem
        i. https://www.geeksforgeeks.org/partition-problem-dp-18/
    l. Problem-12: Cutting a Rod
        i. https://www.geeksforgeeks.org/cutting-a-rod-dp-13/
    m. Problem-13: Coin Change
        i. https://www.geeksforgeeks.org/coin-change-dp-7/

n. Problem-14: Word Break Problem
    i. https://www.geeksforgeeks.org/word-break-problem-dp-32/
o. Problem-15: Maximum Product Cutting
    i. https://www.geeksforgeeks.org/maximum-product-cutting-dp-36/
p. Problem-16: Dice Throw
    i. https://www.geeksforgeeks.org/dice-throw-dp-30/
q. Problem-17: Box Stacking Problem
    i. https://www.geeksforgeeks.org/box-stacking-problem-dp-22/
r. Problem-18: Egg Dropping Puzzle
    i. https://www.geeksforgeeks.org/egg-dropping-puzzle-dp-11/

Week-7

1. Greedy Algorithms and Dynamic Programming Differences
2. Greedy Algorithms
    a. Activity Selection Problem
    b. Knapsack Problems
        i. The 0-1 knapsack problem
        ii. The fractional knapsack problem

Week-8 (Midterm)

Week-9

1. Heap Data Structure
2. Heap Sort
3. Huffman Coding

Week-10

1. Introduction to Graphs
2. Graphs and Representation
3. BFS (Breath-First Search)
4. DFS (Depth-First Search)
    a. in-order
    b. post-order
    c. pre-order
5. Topological Order
6. SCC (Strongly Connected Components)
7. MST
    a. Prim
    b. Kruskal

Week-11

1. Disjoint Sets and Kruskal Relationships
2. Single-Source Shortest Paths
    a. Bellman-Ford
    b. Dijkstra
3. Q-Learning Shortest Path
4. Max-Flow Min-Cut
    a. Ford-Fulkerson

    b. Edmond's Karp
    c. Dinic

Week-12

1. Crypto++ Library Usage
2. Hashing and Encryption
    a. Integrity Control
        i. Hash Values
            1. Cryptographic Hash Functions
                a. SHA-1
                b. SHA-256
                c. SHA-512
            2. Checksums
                a. MD5
                b. CRC32
            3. Hash Algorithms
                a. SHA-1
                b. SHA-256
                c. SHA-512
                d. H-MAC

Week-13

1. Symmetric Encryption Algorithms
    a. AES
        i. https://formaestudio.com/portfolio/aes-animation/
    b. DES
        i. http://desalgorithm.yolasite.com/
    c. TDES
        i. https://en.wikipedia.org/wiki/Triple_DES
2. Symmetric Encryption Modes
    a. https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation
    b. ECB
    c. CBC
3. Asymmetric Encryption
    a. Key Pairs (Public-Private Key Pairs)
4. Signature Generation and Validation

Week-14

1. OTP Calculation
    a. Time-based
    b. Counter-based
2. File Encryption and Decryption and Integrity Control Operations

Week-15

1. Review

Week-16 (Final)

Links

https://www.youtube.com/watch?v=1PkcbE3zrYo&ab_channel=RIMMS-NUST

https://github.com/LetsTrie/Code-Library-Of-Others/blob/master/sgtlaugh/Hunt-Szymanski.cpp

https://www.interviewbit.com/

https://visualgo.net/en

https://github.com/Savjee/savjeecoin-frontend

https://www.youtube.com/watch?v=AQV0WNpE_3g&ab_channel=SimplyExplained

https://algorithm-visualizer.org/

https://github.com/algorithm-visualizer/algorithm-visualizer

https://github.com/algorithm-visualizer/algorithms

https://github.com/algorithm-visualizer/server

https://github.com/algorithm-visualizer/slideshow

https://github.com/algorithm-visualizer/tracers.js

https://github.com/algorithm-visualizer/tracers.py

https://github.com/algorithm-visualizer/extractor.cpp

https://github.com/algorithm-visualizer/tracers.cpp

https://github.com/algorithm-visualizer/tracers.java

https://github.com/algorithm-visualizer/extractor.java