

**Recep Tayyip Erdogan University**

**Faculty of Engineering and Architecture**

**Computer Engineering**

CE103- Algorithms and Programming I

**Syllabus**

**Fall Semester, 2020-2021**

| Instructor | Asst. Prof. Dr. Uğur CORUH |
| --- | --- |
| Contact Information | [ugur.coruh@erdogan.edu.tr](mailto:ugur.coruh@erdogan.edu.tr) |
| Google Classroom Code | **ouw44uk** |
| Lecture Hours and Days | Tuesday:13:00 – 15:45 (Course)  Wednesday: 10:00 – 11:45 (Lab) |
| Office hours | Meetings will be scheduled over Google Meet with your university account and email and performed via demand emails. |
| Lecture and Communication Language | English |

1. **Course Scope and Goals**

This course goal is to develop algorithm and programming expertise from scratch in a powerful way to provide a high-quality career path for students. The lecture 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. At the end of this course, students will be able to:

* Identify the types of computation and describe digital computation in detail.
* Use language features used in current programming languages.
* Program in different language paradigms and evaluate their relative benefits.
* Understand the key concepts in the implementation of common features of programming languages.
* Describe the types of programming languages and the paradigms.
* Understand the concept of basic data types, such as int, float, bool, and container data types, such as list, tuple, string.
* Differentiate computational evaluation of expressions from mathematical evaluation due to concepts such as overflow, side effect, and truncation.
* Understand and apply structured programming concepts by elaborating on sequential, selective, repetitive structures, such as statements, if/switch/case statements, for/while statements and functions.
* Interpret a computational problem specification and algorithmic solution and implement a C/C++ solution, Java or C#, to solve that problem.
* Argue the correctness of algorithms using inductive proofs and invariants.
* Analyze worst-case running times of algorithms using asymptotic analysis.
* Describe the divide-and-conquer paradigm and explain when an algorithmic design situation calls for it.
* Derive and solve recurrences describing the performance of divide-and-conquer algorithms.

1. **Course Topics**

* Integrated Development Environments
* Continues Integration and Continues Development Processes
* Software Development Principles
* Programming Expertise (C/C++, Java, C#)
* Algorithm Design and Asymptotic Analysis
* Recursive Sorting Algorithms
* Linear Sorting Algorithms
* Heaps

1. **Textbooks and Required Hardware**

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

* *C How to Program, 7/E. Deitel & Deitel. 2013, Prentice-Hall.*
* *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.*

You should have a laptop for programming practices.

1. **Grading System**

| **Homework/Exam** | **Weight** | **Explanation** |
| --- | --- | --- |
| Homework | %15 | Students will be asked to program, run, and present custom algorithms. |
| Midterm | %25 | Take-home exam. (Report) |
| Final | %60 | Project-based exam with online real-time project presentations will be done |

1. **Instructional Strategies and Methods**

The basic teaching method of this course will be done by using synchronous and asynchronous distance education methods. Students are expected to be in the classroom (online platform, zoom, or meet) at the time specified in the course schedule. Attendance will be taken.

1. **Late Homework**

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

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

1. **Module 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.

1. **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 to 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 to all students that looking at another student’s paper or any source other than what is clearly 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 students of the Faculty of Engineering and Architecture, 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 doing would remain within the framework of academic honesty or not.

* 1. **What is acceptable when preparing an assignment?** 
     + Communicating with classmates about the assignment in order 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 on classroom to create a class discussion on some controversial topics.
     + Turning to the web or elsewhere for instructions, for references, and for solutions to technical difficulties, but not for direct solutions 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.
  2. **What is not acceptable?**
     + - Asking a classmate to see his or her 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 own lessons and integrate into your own work
       - Giving or showing to a classmate your solution to a problem when the classmate is struggling to solve it.

1. **Expectations**

During the semester, you are expected to attend classes on time by completing weekly course requirements (readings and assignments). The main communication channel between the instructor and the students will be emailed. Please send your questions about the course via the email address provided to you by the university and to the email address given above. ***Make sure 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.

1. **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 content of this document, the instructor will inform you about this.

**Module Schedule:**

| **Weeks** | **Dates** | **Subjects** | **Other Tasks** |
| --- | --- | --- | --- |
| Week 1 | 13.10.2020  14.10.2020 | Course outline and logistics, An overview of course | Resource Sharing |
| Week 2 | 20.10.2020  21.10.2020 | Introduction to Programming Languages and an Overview of the Computer System with Number Systems | Online Practices |
| Week 3 | 27.10.2020  28.10.2020 | Introduction to Development Environments  and Code Version Systems with CI/CD Processes | Setup and Configuration Practices |
| Week 4 | 03.11.2020  04.11.2020 | Pseudocode / Flow Chart - Primitive Data Types, Operations, Statements and Expressions (C/C++/ Java/C#) | Programming Practices |
| Week 5 | 10.11.2020  11.11.2020 | Conditional Statements  Selection Statements  Repetition Statements  Loops  Functions | Programming Practices |
| Week 6 | 17.11.2020  18.11.2020 | Strings, Arrays, and Pointers | Programming Practices |
| Week 7 | 24.11.2020  25.11.2020 | Review | Programming Practices |
| Week 8 | 01.12.2020  02.12.2020 | **Midterm** | |
| Week 9 | 08.12.2020  09.12.2020 | Introduction: analyzing algorithms, designing algorithms | Programming Practices |
| Week 10 | 15.12.2020  16.12.2020 | Asymptotic notation | Programming Practices |
| Week 11 | 22.12.2020  23.12.2020 | Divide and conquer: Strassen | Programming Practices |
| Week 12 | 29.12.2020  30.12.2020 | Randomized quicksort: analysis | Programming Practices |
| Week 13 | 05.01.2021  06.01.2021 | Medians and order statistics | Programming Practices |
| Week 14 | 12.01.2021  13.01.2021 | Heaps: heapsort, priority queues | Programming Practices |
| Week 15 | 19.01.2021  20.01.2021 | Sorting in linear time | Programming Practices |
| Week 16 | 26.01.2021  27.01.2021 | **Final** | |