# AP CSA Syllabus

**2024-08-05**

**Tutor**: Muhammet Fatih ÇAĞLAR **Phone**: +905452944263 **Mail**: muhammet.caglar@boun.edu.tr

**Note**: This syllabus is prepared based on *AP Computer Science A Course and Exam Description by College Board*.

## Welcome!

This course is equivalent to a first-semester, college-level course in computer science. Computer science involves **problem-solving**, **hardware**, and **algorithms** that help people utilize computers and **incorporate multiple perspectives to address real-world problems** in contemporary life. As the study of computer science continues to evolve, the careful design of the AP Computer Science A course allows students to discover the power of computer science through **rewarding yet challenging** concepts.

## What you should already know?

- **In short, you should be good with ninth grade math.**
- Any significant computer science course builds upon a foundation of mathematical reasoning.
- You should had successfully completed a first-year high school algebra course with a strong foundation of basic linear functions, composition of functions, and problem-solving strategies that require multiple approaches and collaborative efforts.
- You should be able to use a Cartesian (x, y) coordinate system to represent points on a plane.

## What we will do?

The course framework includes two essential components: **Computational Thinking Practices** and **Course Content**. After we have done well with the course, we will acquire five computational thinking skills and four big ideas:

### Skills:

- **Program Design and Algorithm Development**: Determine required code segments to produce a given output.
- **Code Logic**: Determine the output, value, or result of given program code given initial values.
- **Code Implementation**: Write and implement program code.
- **Code Testing**: Analyze program code for correctness, equivalence, and errors.
- **Documentation**: Describe the behavior and conditions that produce identified results in a program.

### Big Ideas:

- **Modularity**: By doing abstractions, break problems down into interacting pieces each with their own purpose.
- **Variables**: *Data* is the basis for reasoning, discussion, or calculation. We use:
  - **variables** to store data.
  - **data structures** to organize multiple data when program complexity increases.
  - **algorithms** to sort, access, and manipulate this data.
- **Control**: Use **control structures** to:
  - do things in order,
  - make decisions,
  - do the same process multiple times.
- **Impact of Computing**: Be aware of privacy, security, and ethical issues. As programmers, we need to understand how our programs will be used and be responsible for the consequences.

## Tentative Schedule:

### Lesson #1:

- Introduction
- Coding environment setup
- Why programming? Why Java?
- Variables and data types
- Compound assignment operators

### Lesson #2:

- Casting and ranges of variables — UNIT 1 done.
- Practice
- Objects: instances of classes
- Creating and storing objects (instantiation)
- Calling a void method
- Calling a void method with parameters

  — HOMEWORK #1: ~25 MCQ —

### Lesson #3:

- Calling a non-void method
- String objects: concatenation, literals, and more
- String methods
- Wrapper classes: `Integer` and `Double`
- Using the `Math` class — UNIT 2 done.

### Lesson #4:

- Practice on UNIT 2

  — HOMEWORK #2: ~25 MCQ, 1 FRQ —

### Lesson #5:

- Boolean expressions
- **if** statements and control flow
- **if-else** statements
- **else if** statements
- Compound boolean expressions
- Equivalent boolean expressions
- Comparing objects — UNIT 3 done.

  — HOMEWORK #3: ~20 MCQ, 2 FRQ —

### Lesson #6:

- **while** Loops
- **for** loops
- Developing algorithms using strings
- Nested iteration
- Informal code analysis — UNIT 4 done.

### Lesson #7:

- Practice on UNIT 4

  — HOMEWORK #4: ~15 MCQ, 2 FRQ —

### Lesson #8:

- Anatomy of a class
- Constructors
- Documentation with comments
- Accessor methods
- Mutator methods

### Lesson #9:

- Writing methods

- Static variables and methods
- Scope and access
- **this** keyword
- Ethical and social implications of computing systems — UNIT 5 done.

  — HOMEWORK #5: ~25 MCQ, 2 FRQ

### Lesson #10:

- Practice on UNIT 5

### Lesson #11:

- Array creation and access
- Traversing arrays
- Enhanced **for** loop for arrays
- Developing algorithms using arrays — UNIT 6 done.

  — HOMEWORK #6: ~15 MCQ, 2 FRQ

### Lesson #12:

- Introduction to `ArrayList`
- `ArrayList` methods
- Traversing ArrayLists
- Developing algorithms using ArrayLists

### Lesson #13:

- Searching
- Sorting
- Ethical issues around data collection — UNIT 7 done.

  — HOMEWORK #7: ~15 MCQ, 1 FRQ

### Lesson #14:

- Practice on UNIT 7

### Lesson #15:

- 2D arrays

- Traversing 2D arrays — UNIT 8 done.

  — HOMEWORK #8: ~10 MCQ, 1 FRQ

### Lesson #16:

- General review and practice

### Lesson #17:

- Creating superclasses and subclasses
- Writing constructors for subclasses
- Overriding methods
- **super** keyword

### Lesson #18:

- Creating references using inheritance hierarchies
- Polymorphism
- `Object`superclass — UNIT 9 done.

  — HOMEWORK #9: ~15 MCQ, 2 FRQ

### Lesson #19:

- Practice on UNIT 9

### Lesson #20:

- Recursion
- Recursive searching and sorting — UNIT 10 done.

  — HOMEWORK #10: ~10 MCQ, 1 FRQ

### Lesson #21:

- General review and practice
- Practice Exam 1 (⅓)

### Lesson #22:

- General review and practice
- Practice Exam 1 (⅔)

### Lesson #23:

- Practice Exam 1 (3/3)
- Practice Exam 2 (½)

### Lesson #24:

- Practice Exam 2 (2/2)

### Lesson #25:

- Practice Exam 3 (½)

### Lesson #26:

- Practice Exam 3 (2/2)

### Lesson #27:

- Practice Exam 4

### Lesson #28:

- Practice Exam 5

### Lesson #29:

- Practice Exam 6

### Lesson #30:

- Practice Exam 7

**Note**: Practice Exams 1,2,3 will be solved together. 4,5,6,7 will be solved by the student on his/her own and reviewed and discussed by the totor.

## Resources:

- Main resource: AP Clasroom
- Additional Resources:
    - Codecademy: AP Computer Science A
    - Sololearn: Java
    - 5 steps to a 5: AP Computer Science A
    - Linkedin Learning: Java Essential Training: Syntax and Structure & Java Essential Training: Objects and APIs (a little bit out of scope)

## Advices:

- Please be aware of that this course is not a walk in the park. But, it might be likely if you carefully follow the course from very beginning to the end. If you start to study on last two months, it may reaaaaly painful to understand things. This course is based on two things: 1) Understand the concept. 2) Play around with the code. Once you understand, you will see that they are not very complex, but until that time your mind needs to be relaxed and get used to the concepts. So the key is **"Don't study hard, but study smart."**.

- **During lessons**:

    - Bring your computer.
    - Be an active listener.
    - Ask anything you didn't understand completely.
    - Taking notes is up to you. I will give you the notes anyway.

- **After lessons**:

    - Do your homeworks. This is crucial.
    - Keep the files we use organized. (Digitally or not)
    - Make sure you understand everything completely(except for the things I said "are not that important."). Take notes otherwise and ask me when you came again.
    - Repeat the codes we have written in the lesson and play with them.

**GOOD LUCK!**