# README

# What is this?

Welcome to **Duck Duck Code!**

This is a collection of IO training resources collated by me, Rui Yuan, aka oolimry the duck.

I have coached a range of schools (NUSH, RGS, ACJC, ANDSS, VS), as well the 2024 IOI team and 2024, 2025 EGOI team for Singapore. These materials are collated from my experience in IO coaching, and now I'm publicly releasing them so more people can benefit from them.

# How to use this?

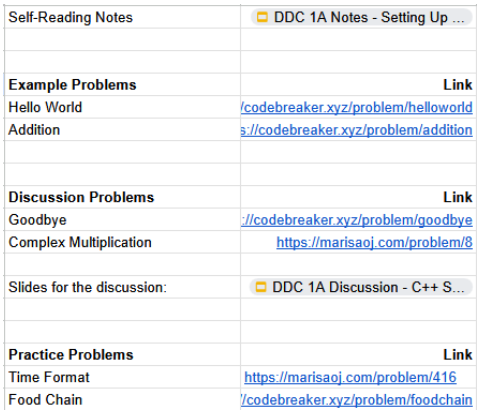
This set of resources is designed for anyone who is starting Competitive Programming. No knowledge of programming is required to begin, and is intended to train the fundamentals of Competitive Programming, so you can train yourself effectively for the future.

This set of resources is intended to be used in one of two ways. First, it can be used as a self-study guide. Second, it can be used as a lesson plan for Computing Teachers, CCA or Interest group leaders who want to conduct IO training in their schools.

Duck Duck Code is organized into units - each unit consists of the following:

1. A set of notes introducing a technique or way of thinking. Typically this contains 1 or 2 example problems, and only covers the technique on a surface level.
2. A problemset consisting:
    a. The example problems from the notes
    b. Discussion problems
    c. Practice problems
3. Solution slides discussing the discussion problems. In these slides, the emphasis is on exploring different ways of using this technique in more complex scenarios.

The units are organized into sections. The section boundaries are somewhat arbitrary but generally reflect a point where you have enough knowledge about certain topics to practice on that range of topics.



| | |
|---|---|
| Self-Reading Notes | DDC 1A Notes - Setting Up ... |
| | |
| | |
| **Example Problems** | **Link** |
| Hello World | /codebreaker.xyz/problem/helloworld |
| Addition | s://codebreaker.xyz/problem/addition |
| | |
| | |
| **Discussion Problems** | **Link** |
| Goodbye | ://codebreaker.xyz/problem/goodbye |
| Complex Multiplication | https://marisaoj.com/problem/8 |
| | |
| Slides for the discussion: | DDC 1A Discussion - C++ S... |
| | |
| | |
| **Practice Problems** | **Link** |
| Time Format | https://marisaoj.com/problem/416 |
| Food Chain | /codebreaker.xyz/problem/foodchain |

Example of a unit in the summary sheet

Each section has a summary sheet that contains everything you should need to know. For example, this is section 1's sheet DDC Section 1 Sheet

**Self-Study**

The following is the recommended way to use the materials

1. Read the self-reading notes
2. Code out the example problems without referring to the notes
3. Look at the discussion problems and think about them for at least 15 minutes each
    a. It is not necessary to solve them, but to attempt to solve them by making observations and such
4. Read the discussion problem slides
5. Code out all problems you didn't solve
6. Attempt some of the practice problems, generally I consider you should move on if you can solve at least half the practice problems in that sheet. It's okay to not solve all the practice problems, as they can generally get tougher as you go down the sheet

**The self-reading notes alone are not sufficient to learn the topic**. Please attempt the discussion problems seriously, and then read the discussion slides once you've solved them or spent sufficient time playing with the problem and you're reading to know the solution.

The self-reading notes are just an introduction to the topic, not a comprehensive guide. You'll find detailed ways to apply these techniques in the discussion problems, and you'll learn even more from attempting the practice problems.

**Trainers**

I'll describe the intended way to use the materials, which is how I designed the units. Of course, you're free to deviate and experiment with how you see fit. It is planned for a session lasting about 1.5 to 2 hours long

1. (Before session) Send the notes for them to self read
2. (First 15 mins of session) either have them self read, or use it to clarify doubts about notes
3. (Next 45 minutes to 1 hour) attempt the discussion problems
    a. Make sure you tell them that the intention is for them to think about the problems and read all of them, not to code them out
4. (Next 20 minutes to 30 minutes) discuss the problems' solutions, you can use the Discussion Slides provided.
5. (Remaining Time) continue working on the problemset

If your sessions are two 1h sessions a week, you can consider splitting the session in the middle of part 3.

I understand that preparing and conducting a lesson is extremely draining, and it is difficult to juggle conducting training for others with training yourself. The purpose of this format is that you don't have to literally teach people and engage your trainees. You just give them the notes to self-read, and then your main job is to discuss the problems in the problemset, which I think is easier than teaching a whole new topic.

For Trainers is a folder containing some tips on how to do training, but it's still a work in progress! Don't be alarmed if it's empty or incomplete.

# Who can use this?

**Self-Study**

Anyone from Singapore Schools! As Codebreaker is mainly a Singapore judge not a global judge, please do not share the link of this outside the context of Singapore Schools

Do be patient with codebreaker accounts enabled. To ensure you're more consistently enabled, make sure you enter your school and name properly, else you'll risk not being enabled.

**Trainers**

If you are a CCA or interest group leader, I'm very proud of you for doing something selfless like this, and you're doing a great service to the community. I hope this set of materials are designed to hopefully make your life easier. School teachers are welcome to use these materials too.

If you are coaching schools for money, I appreciate your effort to spread IO as well. Though since this is technically for work/commercial purposes, I would appreciate it if you let me know first that you're intending to use my materials for your work purposes. You can email this account, or text me if you know me personally.

**If you are a private tutor (either 1 to 1 or tuition centre), these materials are off limits**. I firmly believe that paid, private tuition is counterproductive to the IO community, and that would go against the mission of Duck Duck Code.

If you see these materials being used in private tuition, please let me know. I don't think there's much I can do but at least I think I have enough of a reputation in this community to call them out publicly if I feel like it.

# Getting Started

To get started, you should head over to section 1! Section 1. You should start with the first set of self-reading notes: DDC 1A Notes - Setting Up + C++ Syntax

If you're looking for something else, here's the folder to the full content Full Content

This is our discord server! https://discord.gg/jRWd38AFv4

# Directory

Inside there you'll find the following:

Section 1:
- Focus on **solving problems** and writing **correct code**
- Introduction to C++ Syntax
- Problem Solving Techniques

Section 2:
- Focus on designing **fast algorithms**
- Big O, searching algorithms, data structures, greedy, etc

Problem Solving Spreadsheet:

- A spreadsheet of problems which require minimal prerequisite knowledge, and tests your problem solving abilities
- Problems which I (subjectively) consider the most elegant problems that I know of
- **WARNING: THESE CAN GET VERY HARD.** I think someone who has finished the entire section 2 might find the problems at difficulty 5 or 6 challenging already.

Additionally, the following folders are still a work in progress (as of Apr 2026)

📁 Section 3 (WIP)
- Focus on more specific and advanced algorithms
- Graphs, Dynamic Programming and Segment Trees
- **I don't recommend touching this until you can comfortably solve difficulty 4+ problems in section 2**
- Target: those aiming for a gold medal in NOI

📁 For Trainers (WIP)
- Some advice from me and other trainers on how to effectively train