Competitive Programming and Mathematics Society

# Programming Workshop #0

## Competitive Programming 101

**Patrick Moore and Ryan Ong**

# Welcome

CPMSOC

- There will be a programming workshop in weeks 4, 6, 8.
- Each workshop will run for up to two hours.
- Each workshop (including this one) will have an accompanying problem set, where you can practise what has been taught.
- These workshops are targeted at people who have some knowledge of programming, not necessarily competitive programming
- Please feel free to ask questions at any time.

# Today's Workshop

CPMSOC

1. Welcome
2. What is competitive programming?
3. What do we learn in competitive programming?
4. Competitions and Training
5. Time Complexity
6. IMC Coding Competitions
7. Wrap up

# What is competitive programming?

CPMSOC

**Definition**

Competitive programming involves individuals or teams working to solve computer science problems under a time limit.

Each problem statement contains a couple main parts which will help explain the problem, and explain how you will submit your work.

- Problem Description
- Input and Output Descriptions
- Sample Inputs and Outputs
- Subtasks and Constraints

# A Basic Competitive Programming Problem

CPMSOC

- **Statement:** Tom makes $A$ million dollars per year working as a trading intern at IMC Trading. However, he has to spend $B$ million dollars per year on pizza for his CPMSoc workshops. What is his net profit per year (in millions of dollars) after working as a trading intern and spending money on pizza?
- **Input** Two integers, $A$ and $B$ ($0 \leq A, B \leq 10$).
- **Output:** A single integer, Tom's net profit per year (in millions of dollars).
- **What is the question asking?**

# A Basic Competitive Programming Problem

CPMSOC

- **Statement:** Tom makes $A$ million dollars per year working as a trading intern at IMC Trading. However, he has to spend $B$ million dollars per year on pizza for his CPMSoc workshops. What is his net profit per year (in millions of dollars) after working as a trading intern and spending money on pizza?
- **Input** Two integers, $A$ and $B$ ($0 \leq A, B \leq 10$).
- **Output:** A single integer, Tom's net profit per year (in millions of dollars).
- **What is the question asking?** Print $A - B$, given $A$ and $B$.

# Submitting to a problem

CPMSOC

## Implementation (C or C++)

```c
#include <stdio.h>
int main() {
    // Read input
    int a, b;
    scanf("%d %d", &a, &b);

    // Compute answer
    int ans = a - b;

    // Print output
    printf("%d\n", ans);
}
```

### Running locally

- Compile: g++ source.cpp
- Run: ./a.out

### Online shell

- Go to the website cpp.sh

### Submitting

- Submit file or copy/paste code to the judging site.
- Your file will be judged automatically! You may receive 0, partial or 100 points for your submission

# What do we learn in competitive programming?



Competititve programming is not only fun, but very educational too

- Implementation / Maths
- Data Structures
- Algorithms
- Graph Theory
- Dynamic Programming
- Computational Geometry

Knowledge of these fields can be useful to solve some more advanced competitive programming problems, but many problems can be solved without them.

# Competitions and Training

CPMSOC

There are many ways to train for competitive programming competitions.

- Learn data structures and algorithms ( cp-algorithms.com and many others )
- Practice problems from previous competitions
- Timed practice competitions (These run all the time)

Most competitive programmers would agree that the best way is to train is to simply do practice problems! Start easy and work your way up.

# Competitions and training

CPMSOC

- International Collegiate Programming Contest
  - Divisionals/Regionals/World Finals
  - ANZAC League
- Big Companies
  - Google Code Jam/Kickstart
  - Facebook Hacker Cup
  - Amazalgo
- Society Competitions
  - CSESoc + Industry sponsors
  - CPMSoc (hopefully)

- Online platforms (for training and/or regular contests)
  - AtCoder
  - Codeforces
  - Topcoder
  - CodeChef
  - ORAC
  - USACO
  - Project Euler
  - HackerRank
  - LeetCode
  - Probably more...

# Time Complexity

CPMSOC

As competitive programmers, our programs often have to run within defined memory and time limits. It's sometimes easy to write a program that solve a given problem, but very difficult to write a program that solves that problem fast.

## Definition

- We say two algorithms have the same time complexity if, as the input parameters get large, the number of operations in one approaches a constant factor of the number of operations in the other.

# Time Complexity

CPMSOC

As competitive programmers, our programs often have to run within defined memory and time limits. It's sometimes easy to write a program that solve a given problem, but very difficult to write a program that solves that problem fast.

## Definition

- We say two algorithms have the same time complexity if, as the input parameters get large, the number of operations in one approaches a constant factor of the number of operations in the other.
- This allows us to compare algorithms meaningfully, while preserving the useful ambiguity of what constitutes an "operation".

# Time Complexity

CPMSOC

As competitive programmers, our programs often have to run within defined memory and time limits. It's sometimes easy to write a program that solve a given problem, but very difficult to write a program that solves that problem fast.

## Definition

- We say two algorithms have the same time complexity if, as the input parameters get large, the number of operations in one approaches a constant factor of the number of operations in the other.
- This allows us to compare algorithms meaningfully, while preserving the useful ambiguity of what constitutes an "operation".
- We can denote time complexity with big O notation. For example, $\frac{1}{2}n^2 + n = O(n^2)$. Usually, we can just take the dominating term and remove any constant coefficient.
- We usually care about the time complexity in the worst case.

# Time Complexity - A Game to Demonstrate

CPMSOC

## Example

**Problem:** I'm thinking of an integer, $K$, somewhere between 1 and 1000 inclusive. You may ask me $Q$ queries of a single number. I will tell you if it is lower, higher or equal to $K$. Find out the number I'm thinking of within $Q$ queries.

# Time Complexity - A Game to Demonstrate

CPMSOC

## Example

**Problem:** I'm thinking of an integer, $K$, somewhere between 1 and 1000 inclusive. You may ask me $Q$ queries of a single number. I will tell you if it is lower, higher or equal to $K$. Find out the number I'm thinking of within $Q$ queries.

**Subtasks:** $Q <= 1000$; $Q <= 100$; $Q <= 12$

# Time Complexity - A Game to Demonstrate

CPMSOC

## Example

**Problem:** I'm thinking of an integer, $K$, somewhere between 1 and 1000 inclusive. You may ask me $Q$ queries of a single number. I will tell you if it is lower, higher or equal to $K$. Find out the number I'm thinking of within $Q$ queries.

**Subtasks:** $Q$ <= 1000; $Q$ <= 100; $Q$ <= 12

**Exercises:** What's the most basic way to find $K$? (Solves Subtask 1 only)

# Time Complexity - A Game to Demonstrate

CPMSOC

## Example

**Problem:** I'm thinking of an integer, $K$, somewhere between 1 and 1000 inclusive. You may ask me $Q$ queries of a single number. I will tell you if it is lower, higher or equal to $K$. Find out the number I'm thinking of within $Q$ queries.

**Subtasks:** $Q <= 1000$; $Q <= 100$; $Q <= 12$

**Exercises:** What's the most basic way to find $K$? (Solves Subtask 1 only)

Can you think of a more efficient way to find $K$? (Solves Subtasks 2 or 3)

# Time Complexity - A Game to Demonstrate

CPMSOC

## Example

**Problem:** I'm thinking of an integer, $K$, somewhere between 1 and 1000 inclusive. You may ask me $Q$ queries of a single number. I will tell you if it is lower, higher or equal to $K$. Find out the number I'm thinking of within $Q$ queries.

**Subtasks:** $Q <= 1000$; $Q <= 100$; $Q <= 12$

**Exercises:** What's the most basic way to find $K$? (Solves Subtask 1 only)

Can you think of a more efficient way to find $K$? (Solves Subtasks 2 or 3)

What is the maximum number of questions you will ever have to ask with this new algorithm (The worst case)? Can you define this in terms of the maximum $K$?

# Magic Numbers

CPMSOC

**Definition**

As a very rough guide, a reasonable judging machine can perform 50 to 200 million medium sized operations in a second.

# Magic Numbers

CPMSOC

## Definition

As a very rough guide, a reasonable judging machine can perform 50 to 200 million medium sized operations in a second.

| Maximum $n$ | Appropriate time complexity |
|---|---|
| 10 | $O(n!)$ |
| 20 | $O(n \cdot 2^n)$ |
| 100 | $O(n^3)$ |
| $10^3$ | $O(n^2)$ |
| $10^5$ | $O(n^{3/2})$ or $O(n \log^2 n)$ |
| $10^6$ | $O(n \log n)$ |
| $10^7$ | $O(n)$ |
| $10^9$+ | $O(\log n)$ or $O(1)$ |

# CSESoc x CPMSoc x IMC Competition

CPMSOC

We are doing a collaboration competition with CSESoc and IMC this Saturday, 12th March, 3pm - 5:30pm.

It's hosted online via HackerRank, and you compete in teams of up to 3 to solve problems for 2 hours.

There are 2 divisions, one for relatively experienced programmers, and one for newer programmers.

Many of the problems are very approachable, so most teams should be able to score quite a few points.

$2000 in prizes to be won!

# IMC Coding Competitions

# Wrap up

CPMSOC

- Sign up to HackerRank and have a go at a couple practice problems.
- Get into the 2022 IMC x CSESoc x CPMSoc Coding Competition. (shorturl.at/jxGJO)
  - Vivian and Sarahs Studex tasks
  - Ceces Honeycomb
  - Trading at IMC 1
  - Housing Restrictions
  - Infinity War Spoilers
- Next sessions will be in weeks 6, 8. Hopefully same time and place.
- There will also be Maths sessions in weeks 5, 7, 9. Wednesday 12-2. First one is about number theory!
- More CPMSoc competitions