BRACU CP Workshop

Day 1

Arman Ferdous, MD Mazed Hossain

BRAC University, Dhaka

9 April 2025

Intro

- What is competitive programming?

Intro

- What is competitive programming?
- Why you should be interested in it?

Intro

The Big Competitions

- IOI & ICPC

Intro

The Big Competitions

- IOI & ICPC
- Bangladesh won it's first ever Gold Medal at IOI in 2024!

Intro

The Big Competitions

- IOI & ICPC
- Bangladesh won it's first ever Gold Medal at IOI in 2024!
- We are yet to win any medal at the ICPC WF.

Intro

The Contest Environment

- 1 PC, 3 Members, a dozen problems and hundreds of balloons.

Intro

Your Roadmap

- How can you reach the topmost level?

Intro

Your Roadmap

- How can you reach the topmost level?
- Come to our selection contests, get selected as part of a team. Communication is the key here.

Intro

Your Roadmap

- How can you reach the topmost level?
- Come to our selection contests, get selected as part of a team. Communication is the key here.
- Participate at the ICPC Dhaka Regional Contest.

Intro
Your Roadmap

- How can you reach the topmost level?
- Come to our selection contests, get selected as part of a team. Communication is the key here.
- Participate at the ICPC Dhaka Regional Contest.
- Dhaka Regional Champion $\rightarrow$ directly qualified for WF!

Intro
Your Roadmap

- How can you reach the topmost level?
- Come to our selection contests, get selected as part of a team. Communication is the key here.
- Participate at the ICPC Dhaka Regional Contest.
- Dhaka Regional Champion $\rightarrow$ directly qualified for WF!
- Around top 25 $\rightarrow$ qualified for the ICPC Asia West Continent Championship.

Intro
Your Roadmap

- How can you reach the topmost level?
- Come to our selection contests, get selected as part of a team. Communication is the key here.
- Participate at the ICPC Dhaka Regional Contest.
- Dhaka Regional Champion $\rightarrow$ directly qualified for WF!
- Around top 25 $\rightarrow$ qualified for the ICPC Asia West Continent Championship.
- ICPC Asia West top 10 $\rightarrow$ qualified for the World Finals.

Intro

Your Roadmap

- How can you reach the topmost level?
- Come to our selection contests, get selected as part of a team. Communication is the key here.
- Participate at the ICPC Dhaka Regional Contest.
- Dhaka Regional Champion $\rightarrow$ directly qualified for WF!
- Around top 25 $\rightarrow$ qualified for the ICPC Asia West Continent Championship.
- ICPC Asia West top 10 $\rightarrow$ qualified for the World Finals.
- Reach World Final top 12 $\rightarrow$ get Bangladesh's first ever medal.

BUT WHAT WILL YOU BECOME TOMORROW?
薬局
Next chapter hits Oct. 11!

Mandatory Haikyuu shoutout!

Where to Practice

Online Judges

- Codeforces, Codechef, Atcoder.

CODEFORCES Sponsored by TON

CODECHEF

AtCoder

Where to Practice
Online Judges

- Codeforces, Codechef, Atcoder.
- Participate in contests regularly (2-3x a week). Make sure to upsolve!

The Highest Solved Problem In CF

Watermelon - 4A

- Two friends want to divide a watermelon that weights w $(1 \leq w \leq 100)$ kilograms.

The Highest Solved Problem In CF

Watermelon - 4A

- Two friends want to divide a watermelon that weights w $(1 \leq w \leq 100)$ kilograms.
- They want to divide the watermelon in such a way that each of the two parts weighs an even number of kilos, but it is not necessary for the parts to be equal.

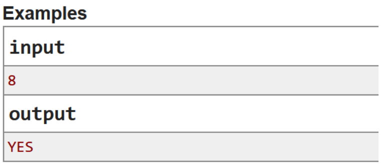
The Highest Solved Problem In CF

Watermelon - 4A

- Two friends want to divide a watermelon that weights w $(1 \leq w \leq 100)$ kilograms.
- They want to divide the watermelon in such a way that each of the two parts weighs an even number of kilos, but it is not necessary for the parts to be equal.
- Given w determine whether such division is possible or not. Print "YES" or "NO".

The Highest Solved Problem In CF

Watermelon - 4A

- Two friends want to divide a watermelon that weights w $(1 \leq w \leq 100)$ kilograms.
- They want to divide the watermelon in such a way that each of the two parts weighs an even number of kilos, but it is not necessary for the parts to be equal.
- Given w determine whether such division is possible or not. Print "YES" or "NO".
- 

Examples

```
input
8
output
YES
```

Verdicts

Online Judges

- Your codes may receive one of many verdicts upon submitting a solution.

Verdicts
Online Judges

- Your codes may receive one of many verdicts upon submitting a solution.
- Accepted (AC) Problem solved!

Verdicts

Online Judges

- Your codes may receive one of many verdicts upon submitting a solution.
- Accepted (AC) Problem solved!
- Wrong Answer (WA): Your output was incorrect.

Verdicts

Online Judges

- Your codes may receive one of many verdicts upon submitting a solution.
- Accepted (AC) Problem solved!
- Wrong Answer (WA): Your output was incorrect.
- Time Limit Exceed (TLE): Your code did not finish within the allotted time.

Verdicts

Online Judges

- Your codes may receive one of many verdicts upon submitting a solution.
- Accepted (AC) Problem solved!
- Wrong Answer (WA): Your output was incorrect.
- Time Limit Exceed (TLE): Your code did not finish within the allotted time.
- Run Time Error (RTE): Your code crashed while running.

Verdicts

- Your codes may receive one of many verdicts upon submitting a solution.
- Accepted (AC) Problem solved!
- Wrong Answer (WA): Your output was incorrect.
- Time Limit Exceed (TLE): Your code did not finish within the allotted time.
- Run Time Error (RTE): Your code crashed while running.
- Memory Limit Exceed (MLE): Too much memory consumption.

Verdicts
Online Judges

- Your codes may receive one of many verdicts upon submitting a solution.
- Accepted (AC) Problem solved!
- Wrong Answer (WA): Your output was incorrect.
- Time Limit Exceed (TLE): Your code did not finish within the allotted time.
- Run Time Error (RTE): Your code crashed while running.
- Memory Limit Exceed (MLE): Too much memory consumption.
- Presentation Error (PE): Your output was off by whitespaces.

How to fix?
Online Judges

- WA: Debug your code. (incorrect logic/idea, typos, etc.)

How to fix?
Online Judges

- WA: Debug your code. (incorrect logic/idea, typos, etc.)
- TLE: Improve your code (too slow solution/unintentional inf loop).

How to fix?
Online Judges

- WA: Debug your code. (incorrect logic/idea, typos, etc.)
- TLE: Improve your code (too slow solution/unintentional inf loop).
- RTE: Only happens on out of bounds access or division by 0.

How to fix?
Online Judges

- WA: Debug your code. (incorrect logic/idea, typos, etc.)
- TLE: Improve your code (too slow solution/unintentional inf loop).
- RTE: Only happens on out of bounds access or division by 0.
- MLE: Calculate your memory uses.

How to fix?

Online Judges

- WA: Debug your code. (incorrect logic/idea, typos, etc.)
- TLE: Improve your code (too slow solution/unintentional inf loop).
- RTE: Only happens on out of bounds access or division by 0.
- MLE: Calculate your memory uses.
- PE: Check your output formatting (cases, spaces, newlines).

Time Complexity

- You will get a formal introduction to time complexity from CSE221 course.
  But for our purposes, we can simply think about the order of magnitude a code takes to run.

Time Complexity

Example 1

-
  ```
  // Take n, m input from the user.
  sum = 0;
  for (int i = 0; i < n; ++i)
      for (int j = 0; j < m; ++j)
          for (int k = 0; k < 5; ++k)
              sum += i * j * k;
  ```
  The above code's runtime depends on the value of n, m.

Time Complexity
Example 1

-
  ```
  // Take n, m input from the user.
  sum = 0;
  for (int i = 0; i < n; ++i)
      for (int j = 0; j < m; ++j)
          for (int k = 0; k < 5; ++k)
              sum += i * j * k;
  ```
  The above code's runtime depends on the value of n, m.
- We say the code has a time complexity $\mathcal{O}(nm)$.

Time Complexity

Example 1

-
  ```
  // Take n, m input from the user.
  sum = 0;
  for (int i = 0; i < n; ++i)
      for (int j = 0; j < m; ++j)
          for (int k = 0; k < 5; ++k)
              sum += i * j * k;
  ```
  The above code's runtime depends on the value of n, m.
- We say the code has a time complexity $\mathcal{O}(nm)$.
- To measure the approximate worst-case run time: Plug in the values of each term in the complexity so that it evaluates as largest. Divide by 10^8 to find time in seconds.

Time Complexity

Example 2

-
  ```python
  # A python example
  n = int(input())     # length of array
  arr = [int (x) for x in input().split()]

  for i in range(n):
      print(sum(a) - a[i])
  ```

Time Complexity

Example 2

-
  ```
  # A python example
  n = int(input())     # length of array
  arr = [int (x) for x in input().split()]

  for i in range(n):
      print(sum(a) - a[i])
  ```
- Builtin functions can be useful, but make sure you know their complexity. This code runs in $\mathcal{O}(n^2)$.

Time Complexity
Example 3

-
  ```
  # Take a positive integer x from the user.
  while (x) {
      if (x % 2 == 0) x /= 2;
      else x--;
  }
  ```
  What is the complexity of this code?

Time Complexity

Example 3

-
  ```
  # Take a positive integer x from the user.
  while (x) {
      if (x % 2 == 0) x /= 2;
      else x--;
  }
  ```
  What is the complexity of this code?
- It is $\mathcal{O}(\log_2(x))$. As the base-2 logarithm is so common in computer science, it even has its own symbol $\mathcal{O}(lg(x))$.

Time Complexity

Example 4

- What was the time complexity of our solution to Watermelon problem?

Time Complexity

Example 4

- What was the time complexity of our solution to Watermelon problem?
- Our solution works in constant time. This complexity is denoted as $\mathcal{O}(1)$, which is theoretically the fastest complexity.

Time Complexity

Example 4

- What was the time complexity of our solution to Watermelon problem?
- Our solution works in constant time. This complexity is denoted as $\mathcal{O}(1)$, which is theoretically the fastest complexity.
- There are many more weird complexity functions that you will encounter in your journey. Here is a checklist for you! See if you know any algorithm that works in the following complexities: $\mathcal{O}(n\lg(n)), \mathcal{O}(2^n n^2), \mathcal{O}(n!), \mathcal{O}(n\sqrt{m}), \mathcal{O}(n\lg(\lg(n))), \mathcal{O}(n^{\lg(3)})$.

Memory Complexity

- Similar to time complexity, there is also the concept of memory complexity. You can measure it in the same way, and write it in the big-O notation.

Memory Complexity

- Similar to time complexity, there is also the concept of memory complexity. You can measure it in the same way, and write it in the big-O notation.
- But what we need to memorize are the sizes of all fundamental data types.
  `bool, char` - 1 byte
  `int` - 4 byte `long long` - 8 byte

Memory Complexity

- Similar to time complexity, there is also the concept of memory complexity. You can measure it in the same way, and write it in the big-O notation.
- But what we need to memorize are the sizes of all fundamental data types.
  `bool, char` - 1 byte
  `int` - 4 byte `long long` - 8 byte
- Once you know how many bytes your code uses, divide by 1024^2 to calculate in MB.

The 3 Important Areas

- Doing well in competitive programming and problem solving comes down to ultimately 3 areas.

The 3 Important Areas

- Doing well in competitive programming and problem solving comes down to ultimately 3 areas.
- 1. Observation/Thinking Skills: How well you can come up with own ideas. Hardest to train & teach, and takes a good amount of time to improve.

The 3 Important Areas

- Doing well in competitive programming and problem solving comes down to ultimately 3 areas.
- 1. Observation/Thinking Skills: How well you can come up with own ideas. Hardest to train & teach, and takes a good amount of time to improve.
- 2. Theory: How well you know the common algorithms/data structures/tricks & techniques. Much easier to improve & teach.

The 3 Important Areas

- Doing well in competitive programming and problem solving comes down to ultimately 3 areas.
- 1. Observation/Thinking Skills: How well you can come up with own ideas. Hardest to train & teach, and takes a good amount of time to improve.
- 2. Theory: How well you know the common algorithms/data structures/tricks & techniques. Much easier to improve & teach.
- 3. Coding Skill: How quickly you can convert your ideas to code (without bugs). Must be trained individually.

Why You Are Stuck...

- Many beginners get stuck at lower ratings even after solving hundreds of problems in various online judges.

Why You Are Stuck...

- Many beginners get stuck at lower ratings even after solving hundreds of problems in various online judges.
- Reason? They only learn theory. But they don't work on their observation skills.

Why You Are Stuck...

- Many beginners get stuck at lower ratings even after solving hundreds of problems in various online judges.
- Reason? They only learn theory. But they don't work on their observation skills.
- How do you even train this? By keeping a structured method that you apply on every single problem, it becomes a routine work.

Simple Puzzle 1

Water Bottle Puzzle

- There is a 3L bottle, and a 5L bottle. You have an infinite supply of water. Can you measure 4L of water? The bottles have no marks on them, so your only options are to fill them up entirely/empty them out/switch from one bottle to another.

Simple Puzzle 2

Yet Another Puzzle With River & Boat

- A man is walking down the village road with a tiger, a goat and a bundle of grass. Soon he arrives at the river bank where there is one tiny boat that can carry him and another animal or grass at a time.
  Here is the problem: Left alone, the tiger will eat the goat. And similarly, the goat will eat the grass bundle. How is he going to take all three across the river safely?

Simple Puzzle 3

Hats

- Alice and Bob are in prison and the warden plays a game with them. The warden puts on a hat (White/Black) on each of their heads while they were not looking. Now Alice and Bob are in front of each other and can see the other person's hat, but can't see their own.
  They must guess together what their own hat color is. If either one of them is correct the warden will let them go. Find a strategy for Alice & Bob.

Takeaway

- All these puzzles, once you start thinking (or putting in the slightest effort), they start to open up.

Takeaway

- All these puzzles, once you start thinking (or putting in the slightest effort), they start to open up.
- If you don't try at all, the options will always seem overwhelming. But if you calmly take it one step at a time, you can find the solutions without even realizing.

Takeaway

- All these puzzles, once you start thinking (or putting in the slightest effort), they start to open up.
- If you don't try at all, the options will always seem overwhelming. But if you calmly take it one step at a time, you can find the solutions without even realizing.
- These problems are "Linear" in a sense - "Following the most obvious route leads you to the solution".

Takeaway

- All these puzzles, once you start thinking (or putting in the slightest effort), they start to open up.
- If you don't try at all, the options will always seem overwhelming. But if you calmly take it one step at a time, you can find the solutions without even realizing.
- These problems are "Linear" in a sense - "Following the most obvious route leads you to the solution".
- Harder problems will have more choices (and not so trivial ones) to make and thus take more time. But to be a good problem solver, you must be willing to take your shot.

A Framework That Works

- Unfortunately there is no single way to think. To make it even more complicated, different problem areas (i.e. dynamic programming, graphs, strings etc.) require you to think in different ways.

A Framework That Works

- Unfortunately there is no single way to think. To make it even more complicated, different problem areas (i.e. dynamic programming, graphs, strings etc.) require you to think in different ways.
- But from our experience, the following techniques are very useful in solving problems.

R for Reduction

- Refers to reducing the problem to a smaller scale.

R for Reduction

- Refers to reducing the problem to a smaller scale.
- It can be as simple as reducing the maximum limits of the problem.

R for Reduction

- Refers to reducing the problem to a smaller scale.
- It can be as simple as reducing the maximum limits of the problem.
- For example if a geometry problem asks you to find something in 3D, consider solving the 2D (or even 1D) version first?

R for Reduction

- Refers to reducing the problem to a smaller scale.
- It can be as simple as reducing the maximum limits of the problem.
- For example if a geometry problem asks you to find something in 3D, consider solving the 2D (or even 1D) version first?
- A very common theme in problems is to start by defining something very weird. And then the tasks require you to calculate something about that. You should always try to think about that defining property first (reduced problem).

R for Reduction

Codeforces - 2070A

- A number is *FizzBuzz* if it has the same remainders modulo 3 and modulo 5.

R for Reduction

Codeforces - 2070A

- A number is *FizzBuzz* if it has the same remainders modulo 3 and modulo 5.
- Given n $(0 \leq n \leq 10^9)$, print how many `FizzBuzz` integers are there from 0 to n. There will be upto 10^4 testcases.

E for Extreme Case Analysis

- Idea is to look a the maximum/minimum or any other extreme point of interests.

E for Extreme Case Analysis

- Idea is to look a the maximum/minimum or any other extreme point of interests.
- A problem asks you to find maximum profit? Figure out what is the theoretically highest you can achieve if everything went your way. Can you always achieve it?

E for Extreme Case Analysis

- Idea is to look a the maximum/minimum or any other extreme point of interests.
- A problem asks you to find maximum profit? Figure out what is the theoretically highest you can achieve if everything went your way. Can you always achieve it?
- Thinking about upper/lower bounds and conditions that must hold for them.

E for Extreme Case Analysis

- Idea is to look a the maximum/minimum or any other extreme point of interests.
- A problem asks you to find maximum profit? Figure out what is the theoretically highest you can achieve if everything went your way. Can you always achieve it?
- Thinking about upper/lower bounds and conditions that must hold for them.
- Often the necessary conditions to reach them is sufficient.

E for Extreme Case Analysis

Codechef - PDEL

- A string is palindrome if it reads the same from backwards. You are given a string S. In a single move you can delete any non-palindromic substring from it (can't delete the whole string).
  Find the minimum number of moves to turn S into a palindrome, or report it is impossible to do so.
  Example: racecar $\rightarrow$ 0
  Example: modem $\rightarrow$ 1
  Example: my $\rightarrow$ impossible.

E for Extreme Case Analysis

Codechef - PDEL

- A string is palindrome if it reads the same from backwards. You are given a string S. In a single move you can delete any non-palindromic substring from it (can't delete the whole string).
  Find the minimum number of moves to turn S into a palindrome, or report it is impossible to do so.
  Example: racecar $\rightarrow$ 0
  Example: modem $\rightarrow$ 1
  Example: my $\rightarrow$ impossible.
- Limits: $N \leq 2 \times 10^5$.

E for Extreme Case Analysis

Codechef - PDEL

-

C for Counting Contribution

- Whenever you need to calculate the sum of anything complicated, you can think about which term appears how many times (i.e. their contribution) and add them up collectively.

$$x_1 + x_2 + \ldots + x_n = \sum_{\text{unique x}} x \times f(x)$$

C for Counting Contribution

- Whenever you need to calculate the sum of anything complicated, you can think about which term appears how many times (i.e. their contribution) and add them up collectively.

$$x_1 + x_2 + ... + x_n = \sum_{\text{unique x}} x \times f(x)$$

- Sounds like a silly thing to mention. But it is one of the most applied general technique in probability/math problems.

C for Counting Contribution

Sample Problem

- Given two arrays A and B of length n. Find the following value:

$$\sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} a_i b_j$$

I for Inverting the Problem

- Thinking about the problem from the opposite direction.

I for Inverting the Problem

- Thinking about the problem from the opposite direction.
- Suppose a problem asks you to maintain an initially filled data structure, where deletions occurs. You can invert the problem and solve for insert only.

I for Inverting the Problem

- Thinking about the problem from the opposite direction.
- Suppose a problem asks you to maintain an initially filled data structure, where deletions occurs. You can invert the problem and solve for insert only.
- Although we are only writing "inverting", this can be anything that modifies the rules of your problem. Remember: Don't play by the rules!

I for Inverting the Problem

Sample Problem

- N ants are on a 1m length rod. Each ant has a position $0 \leq x_i \leq 1$ which is a real value, and a direction it is facing (either L or R).
  All ants start moving at speed 1m/s, at time = 0.

I for Inverting the Problem

Sample Problem

- N ants are on a 1m length rod. Each ant has a position $0 \leq x_i \leq 1$ which is a real value, and a direction it is facing (either L or R).
  All ants start moving at speed 1m/s, at time $= 0$.
- When two ants collide, they change direction and start moving in the opposite direction instantly.
  When an ant reaches one of the ends of the rod, it falls off.
  Find the time it takes for **all ants** to fall off.

P for Pattern Hunting

- As it says, find anything, hope that there is a pattern that your eyes can figure out.

P for Pattern Hunting

- As it says, find anything, hope that there is a pattern that your eyes can figure out.
- Can help you solve problems that you were not meant to solve.

P for Pattern Hunting

- As it says, find anything, hope that there is a pattern that your eyes can figure out.
- Can help you solve problems that you were not meant to solve.
- At a higher level, you can often sense that there must be a pattern and build your ideas on top of it. Who knows, maybe you were correct all along!?

P for Pattern Hunting

LightOJ - 1136

- There is a sequence $1, 12, 123, ..., 123456789, 12345678910, ...$. Given A and B, find the number of integers divisible by 3 from the Ath term to the Bth term.

P for Pattern Hunting

LightOJ - 1136

- There is a sequence $1, 12, 123, \ldots, 123456789, 12345678910, \ldots$. Given A and B, find the number of integers divisible by 3 from the Ath term to the Bth term.
- T (≤ 10000) testcases, $1 \leq A \leq B < 2^{31}$.

E for Escape

- It is okay to switch to a different problem.

E for Escape

- It is okay to switch to a different problem.
- Find your sweet spot to switch from problem to problem to not waste too much time during a contest.

E for Escape

- It is okay to switch to a different problem.
- Find your sweet spot to switch from problem to problem to not waste too much time during a contest.
- Many people shadow others from the ranklist during a contest. While it is a popular strategy it has it's drawbacks.

Your RECIPE

- R = Reduction
  E = Extrem Case Analysis
  C = Counting Contribution
  I = Inverting the Problem
  P = Pattern Hunting
  E = Escape

Your RECIPE

- R = Reduction
  E = Extrem Case Analysis
  C = Counting Contribution
  I = Inverting the Problem
  P = Pattern Hunting
  E = Escape
- By no means it is a set rule. It is just a set of techniques that you can think around. Having your toolbox well organized is the best way to train and get efficient with solving problems.

Up Next

- Starting from next day we will start looking into some well developed tricks to get you started. We will again return to just thinking procedures at the end once we all know about some more topics.
  Till then, have fun!

Good Bye!