Welcome to Programming Fundamentals with C++

This course is designed to teach you the fundamental concepts of programming using C++. We'll explore problem-solving methodologies and build a solid foundation for your programming journey. Get ready to design, analyze, and decompose complex problems!

Course Overview: Mastering C++ Fundamentals

This course aims to master fundamental programming concepts using C++. We'll cover key topics, grading, and the schedule. There are helpful resources such as textbooks and online platforms like tutorials point and w3schools.

Versatility

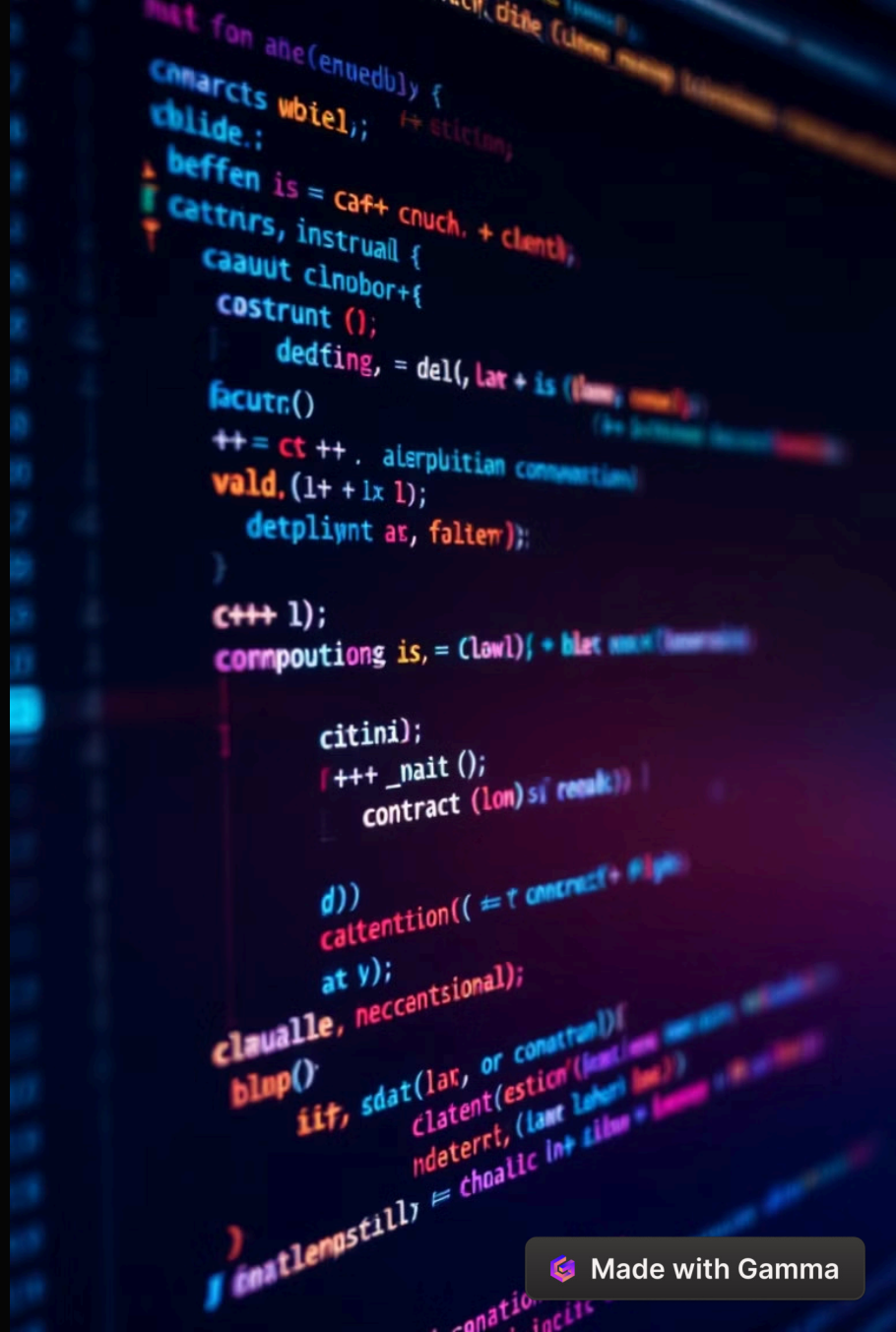
Highly versatile language

Performance

Excellent performance

Relevance

Industry relevance



The Problem-Solving Methodology

A step-by-step guide to solve any problem.

1

Define

Clearly state the problem

2

Analyze

Understand constraints

3

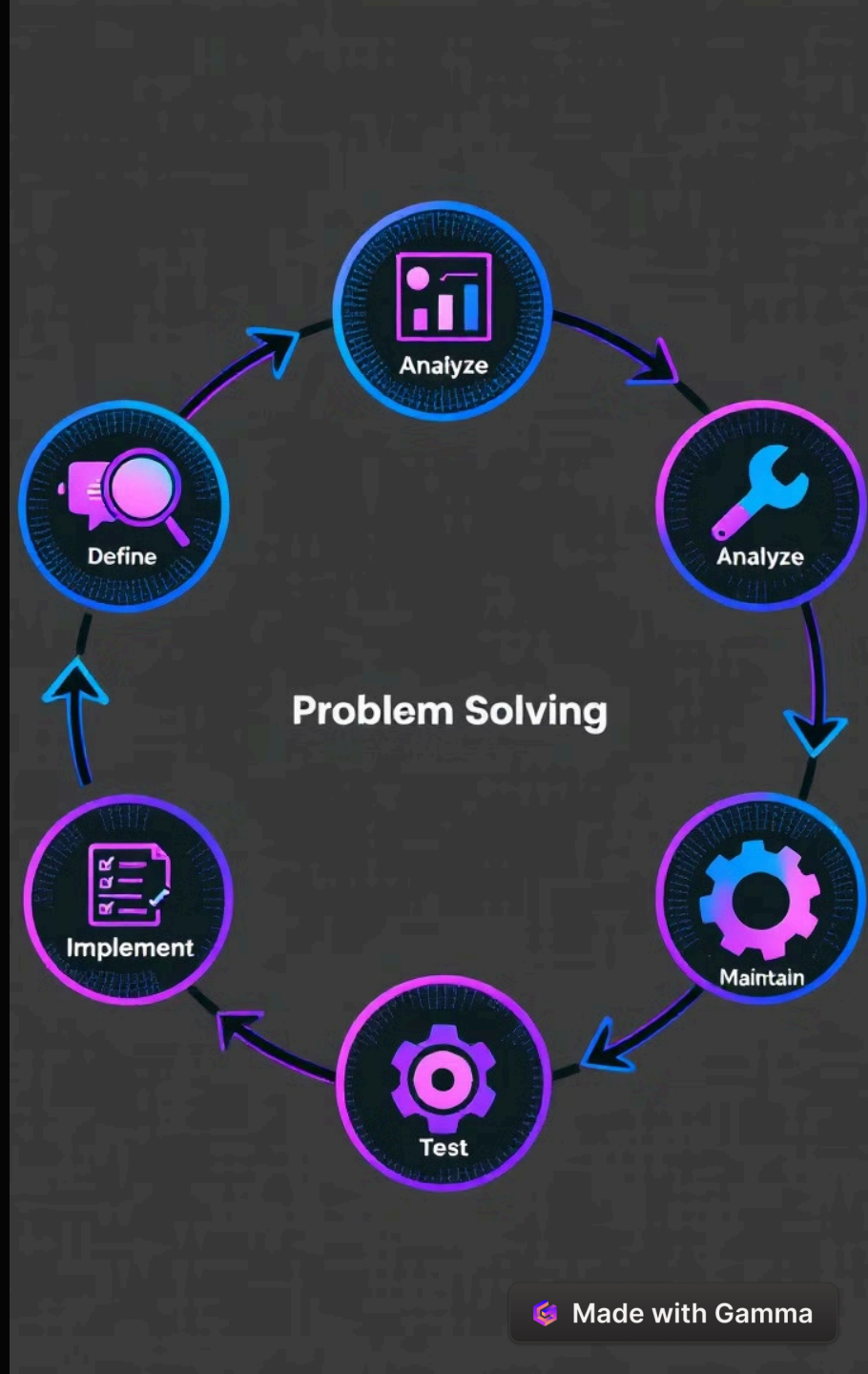
Design

Develop a logical plan

4

Implement

Translate into C++ code



Design: Crafting Solutions with Algorithms

An algorithm is a step-by-step procedure for solving a problem. A good algorithm needs clarity, efficiency, and correctness. Here's the algorithm for calculating the area of a triangle given its base and height.

1 **Base Input**

Get the base of a triangle

2 **Height Input**

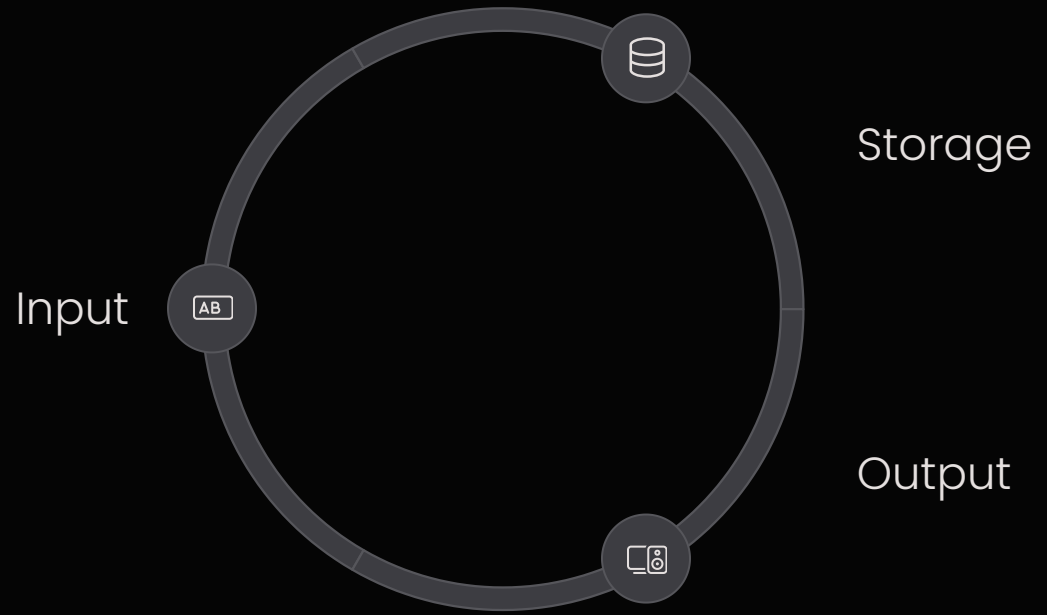
Get the height of a triangle

3 **Calculate**

Multiply base by height then divide by 2

Analyze: Breaking Down Complex Problems

Decompose complex problems into smaller, manageable sub-problems, such as building a calculator app with components like input, storage and output. Simplifies development, enhances maintainability, and facilitates teamwork.



Decompose: Calculating the Average

Write a program to calculate the average of N numbers.

1. Get N
2. Read N
3. Calculate Sum
4. Divide
5. Display

Algorithms: The Heart of Programming Logic

A finite sequence of well-defined instructions to solve a specific problem. They must be unambiguous, feasible, and produce the correct output.

Sorting

- Bubble sort
- Insertion sort

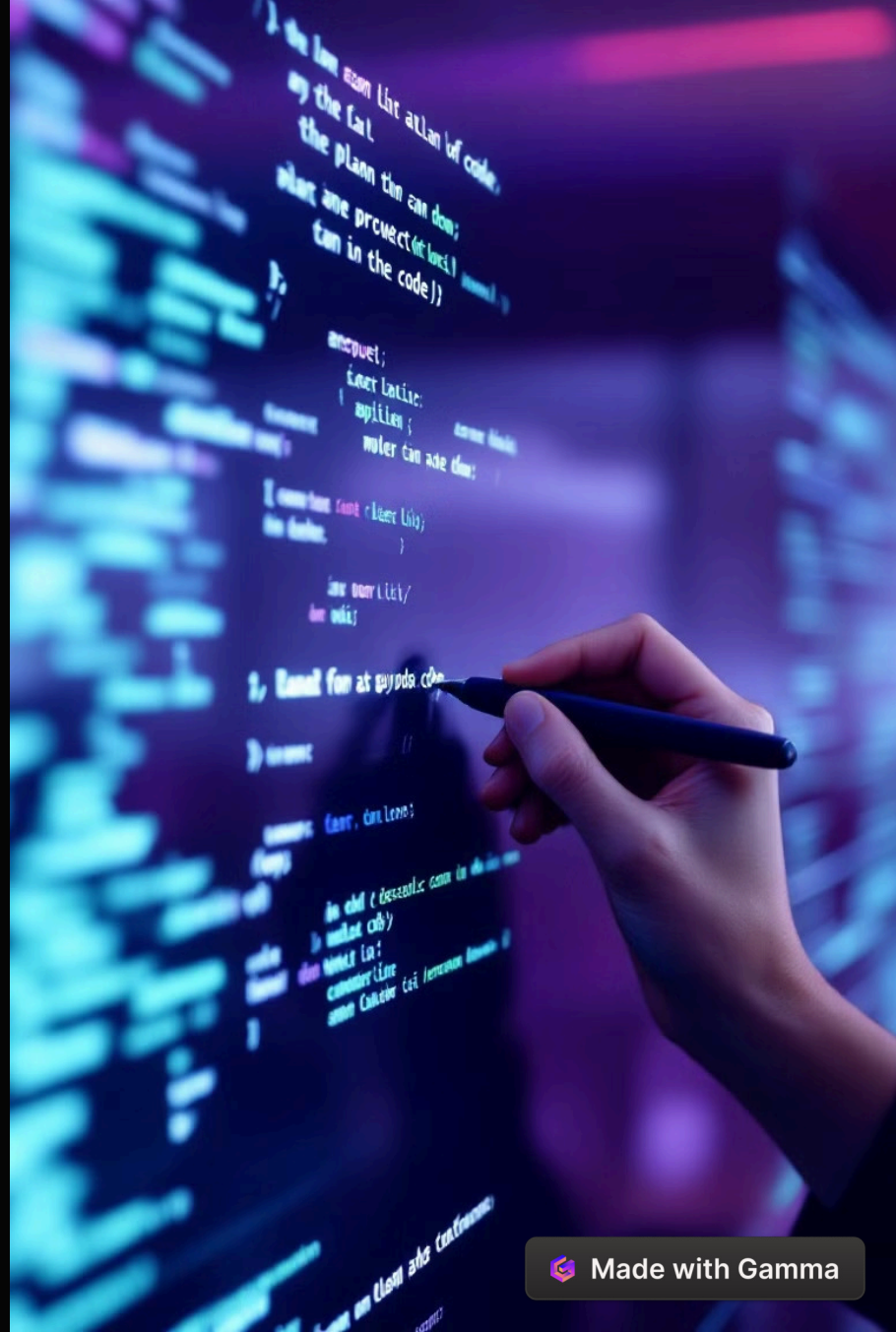
Searching

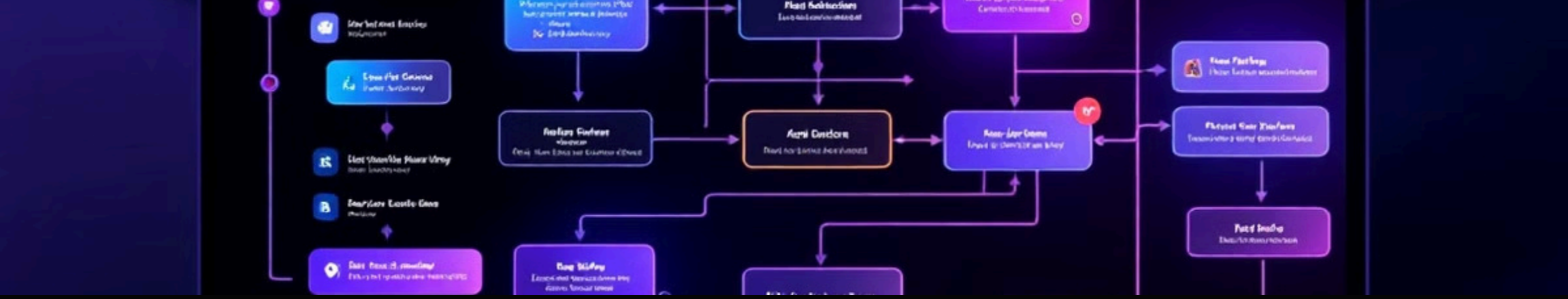
- Linear search
- Binary search

Pseudocode: Bridging the Gap to Code

An informal, high-level description of an algorithm. It outlines the logic of a program before writing actual code, making it easier to understand, facilitate communication, and helps in debugging.

```
Input: array A of numbers
max = A[0]
for each element in A:
    if element > max:
        max = element
Output: max
```





Flow Charts: Visualizing the Process

A diagram that uses symbols to represent the steps of an algorithm. It's easy to visualize the flow of control and helps in identifying logical errors.

Start/End

Process

Decision

Week 1 Recap & Looking Ahead

Key takeaways: Problem-solving methodology, algorithms, pseudocode, and flow charts. Next up are the C++ basics: Variables, data types, and operators. Implement simple algorithms in C++ to reinforce concepts.

C++ Basics

Variables, data types

Practice

Implement algorithms in C++

Next Week

Diving into C++ syntax