

C++ for Absolute Beginners

30 Hands-On Exercises — Organised by Topic

Sections	
1–9	Very Basics — output, variables, types, casting
10–30	Operators & Syntax — conditionals, loops, strings, arrays, structs

Each exercise includes a description, a helpful hint, and the expected output. Try to solve the exercise on your own before reading the hint!

Table of Contents

Section 1 — Very Basics (exercises 1–9)

- 01. Hello, World! (Output) ★■ ■
- 02. Print Your Name (Output) ★■ ■
- 03. Sum of Two Numbers (Variables) ★■ ■
- 04. Read Two Integers (Input) ★■ ■
- 05. int vs float Division (Variable Types) ★■ ■
- 06. double Precision Pi (Variable Types) ★■ ■
- 07. Overflow Demonstration (Variable Types) ★★ ■
- 08. char Arithmetic (Variable Types) ★★ ■
- 09. Type Conversion Chain (Variable Types) ★★★

Section 2 — Operators & Syntax (exercises 10–30)

- 10. Even or Odd (Conditionals) ★■ ■
- 11. Maximum of Two Numbers (Conditionals) ★■ ■
- 12. Leap Year Checker (Logical Operators) ★★ ■
- 13. Login Gate (Logical Operators) ★★ ■
- 14. PIN Change Validator (Logical Operators) ★★ ■
- 15. Countdown with a Loop (Loops (for)) ★★ ■
- 16. Sum 1 to N (Loops) ★★ ■
- 17. Multiplication Table (Loops) ★★ ■
- 18. Factorial (Loops) ★★ ■
- 19. Guess the Number (Loops (while)) ★★ ■
- 20. Collatz Conjecture (Loops (while)) ★★★
- 21. FizzBuzz (Conditionals) ★★ ■
- 22. Reverse a String (Strings) ★★ ■
- 23. Count Vowels (Strings) ★★ ■
- 24. Palindrome Check (Strings) ★★★
- 25. Simple Array Average (Arrays) ★★ ■
- 26. Find Minimum in Array (Arrays) ★★ ■
- 27. Bubble Sort (Arrays) ★★★
- 28. Basic Struct — Student Record (Structs) ★★ ■
- 29. Array of Structs — Temperature Log (Structs) ★★★
- 30. Struct with Comparison (Structs) ★★★

Section 1 — Very Basics

Exercise 1 of 30

Hello, World!



Topic: Output / Program Structure

Description

Write a C++ program that prints the message **Hello, World!** to the screen followed by a newline.

Hint

Use `std::cout` and remember to include `<iostream>`.

Expected Output

Hello, World!

Your Solution



Hint

Expected Output

Your Solution

★ ☐ ☐

Description

Hint

Expected Output

Float division: 3.5

★ ☐ ☐

Description

Hint

Expected Output

Your Solution



★★■

Description

Hint

Expected Output

Your Solution

★ ★ ★

Description

Hint

Expected Output

Your Solution

Exercise 10 of 30

★ ☐ ☐

Description

Hint

Expected Output

Your Solution

★★■

Description

Hint

Expected Output

Your Solution

★★■

Description

Hint

Expected Output

Your Solution

Ask the user to enter a current PIN and a new PIN. Enforce these rules using !=:

- Print **PIN changed successfully** only when both rules pass, otherwise print a specific error message for whichever rule failed.

Use `newPin != currentPin` to catch reuse, and `confirm != newPin` to catch a mistyped confirmation. Check each condition with its own `if` so you can give a precise message.

```
Current PIN: 1234
New PIN: 1234
Error: new PIN must differ from current PIN
---
Current PIN: 1234
New PIN: 5678
Confirm PIN: 9999
Error: PINs do not match
---
Current PIN: 1234
New PIN: 5678
Confirm PIN: 5678
PIN changed successfully
```



Hint

Expected Output

Your Solution

★★■

Description

Hint

Expected Output

Sum = 15

Multiplication Table



Topic: Loops / Nested

Description

Print the multiplication table for numbers 1 through 5. Each row should show: $1 \times 1 = 1$ $1 \times 2 = 2$...

Hint

Use nested for loops — outer for rows, inner for columns.

Expected Output

$$\begin{array}{lll} 1 \times 1 = 1 & 1 \times 2 = 2 & \dots \\ 2 \times 1 = 2 & 2 \times 2 = 4 & \dots \end{array}$$

Your Solution



Hint

Expected Output

Your Solution

★★■

Set a secret number *secret* = 42. Ask the user to keep guessing an integer until they get it right. After each wrong guess, print **Too low!** or **Too high!** as appropriate. When correct, print **Correct!** **You got it in N guesses.**

Use a *while* (*guess* $\neq$ *secret*) loop. Keep a counter variable that increments each iteration.

```
Guess: 20
Too low!
Guess: 60
Too high!
Guess: 42
Correct! You got it in 3 guesses.
```





Hint

Expected Output

Your Solution

Read a single word from the user and print it reversed.

Hint

Loop from the last character index down to 0 using `s.length() - 1`.

Expected Output

```
Enter a word: hello
olleh
```

Your Solution

★★■

Description

Hint

Expected Output

Your Solution



Description

Hint

Expected Output

Your Solution

★★■

Description

Hint

Expected Output

Your Solution



Hint

Expected Output

Min: 1

Your Solution

★ ★ ★

Description

Hint

Expected Output

Sorted: 1 2 3 5 8 9



Hint

Expected Output

Your Solution

★ ★ ★

Description

Print all readings, then compute and print the average temperature, and print the hour at which the maximum temperature was recorded.

Initialise with: Reading $\log[5] = \{\{6, 12.5\}, \{9, 17.3\}, \dots\}$. Loop with `for (int i = 0; i < 5; i++)` and access `log[i].hour` and `log[i].temperature`. Track the max the same way as finding a minimum — seed with the first element.

```
06:00 -> 12.5
09:00 -> 17.3
12:00 -> 22.1
15:00 -> 20.8
18:00 -> 15.4
Average: 17.62
Max temperature at hour 12: 22.1
```



Define a *struct Student* with fields: *name* (string) and *grade* (int). Create two students, fill in their details, and print the one with the higher grade using the ternary operator.

Access struct members with the dot operator: `s.name`. Use `(s1.grade > s2.grade) ? s1 : s2` to pick the better student.

Top student: Bob (95)