# Coding Set 7

## Problem 1 — printTwo (Function template)

Objective: Write a function template printTwo that prints two values (of the same type) separated by a space.
Input: First token: type (int/double/string), then two values.
Output: The two values separated by a space.
Example:
Input: int 5 7 → Output: 5 7

---

## Problem 2 — sumArray (Function template)

Objective: Implement sumArray as a function template that returns sum of n elements of a vector. Types allowed: int or double.
Input: First line: type and n. Second line: n numbers.
Output: Sum (same type as input).
Example:
Input:

int 3

1 2 3

Output:

6

---

## Problem 3 — MinMaxPair (Simple class template)

Objective: Create a class template MinMax<T> that stores two values (minVal, maxVal) and has a constructor taking two T values and a print() method to print them as min max.
Input: type and two values.
Output: min max (use the given values as they are — no need to compute min/max).
Example:
Input: double 2.5 7.1 → Output: 2.5 7.1

---

## Problem 4 — scaleVector (Function template)

Objective: Implement scaleVector<T>(vector<T>& v, T factor) that multiplies every element by factor. Read vector, scale, print result.
Input: type n, then n elements, then factor.
Output: scaled elements space-separated.

**Example:**
**Input:**

**int 3**

**1 2 3**

**2**

**Output:**

**2 4 6**

---

**Problem 5 — StringBox (Class template with string only)**

**Objective: Implement Box<T> but test with T = string. Class stores a value, set and get. Read one string, store it, then print from get().**
**Input: one word (no spaces).**
**Output: the same word.**
**Example:**
**Input: Hello → Output: Hello**

---

**Problems 6-10 — Exception Handling (Easy)**

**Problem 6 — safeDiv (simple)**

**Objective: Read two integers a b. If b==0 print Error (without throwing), otherwise print a/b.** ***(This is warm-up — no throw required.)***
**Input: a b**
**Output: result or Error**
**Example: 10 0 → Error**

---

**Problem 7 — throwOnZero (throw/catch basic)**

**Objective: Read integer x. If x==0 throw the string "Zero" and catch it in main to print Caught Zero. Otherwise print OK.**
**Input: single integer.**
**Output: Caught Zero or OK**
**Example: 0 → Caught Zero**

---

**Problem 8 — parsePositive (validation with exception)**

**Objective: Read one integer as string. If it is negative (starts with -), throw std::invalid_argument and catch to print Negative not allowed. Otherwise print the number. (You may use stoi inside try-catch.)**

**Input: a string representing an integer.**
**Output: number or Negative not allowed**
**Example: -5 → Negative not allowed**

---

**Problem 9 — boundedPush (stack with exception message)**

**Objective: Implement very small stack with capacity cap (cap ≤ 5). Read cap, then m commands (push x or pop). On push when full print Full (use exception or if-check). On pop when empty print Empty, otherwise print popped value.**
**Input example:**

**2 4**

**push 1**

**push 2**

**push 3**

**pop**

**Output:**

**Full**

**2**

---

**Problem 10 — openTxt (simple file-type check)**

**Objective: Read a filename string. If it ends with .txt print OK; else throw and catch a custom exception and print Not txt. Implement a small custom exception class with what() returning "Not txt".**
**Input: filename (single token).**
**Output: OK or Not txt**
**Example: notes.txt → OK ; data.csv → Not txt**