Programming Assignment 3

Filename: `pa3_surname.c`

Long ago, in the ancient land of *Numerica*, mathematicians protected their treasures inside sacred vaults. Each vault is secured by a **secret numerical code**. This code was cleverly hidden within a sequence of ten positive integers engraved on the vault's surface. To unlock a vault, one must first uncover the hidden patterns within these numbers.

Your team has taken on the challenge of decoding these vaults. Through research, you have discovered that each secret code can be determined using a special formula:

```
secret code = mode + count of unique numbers + min + max + range
```

Because there are many vaults to analyze, you are tasked with automating the decoding process. Each vault provides a sequence of ten integers, and for each sequence, your program must:

- Determine the **mode**, which is the number that appears most frequently.
- Count how many **unique numbers** are present in the sequence.
- Identify the **minimum** and **maximum** values.
- Compute the **range**, which is the difference between the largest and smallest numbers.

Your task is to complete the given C program so that it calculates and displays the secret code for each vault sequence. You can access the starter code here. The following function prototypes are provided that define the functions you need to implement:

```
int get_mode(int arr[], int size);
int count_unique(int arr[], int size);
int get_min(int arr[], int size);
int get_max(int arr[], int size);
int get_range(int arr[], int size);
```

Important You must not modify any other parts of the program. Your task is only to complete the definitions of the five functions listed above. Do not add or introduce any helper functions. Failure to follow these restrictions will result in point deductions.

Input

The first line of input contains an integer $0 \leq \mathrm{T} \leq 100$, representing the number of vaults to decode. Each of the next T lines describes a vault's sequence of ten positive integers, each less than $1{,}000$. Each vault's sequence is guaranteed to have a mode, and that mode is unique. There will always be exactly one value that occurs most frequently.

Output

For each vault, the program should output one line containing the vault number, followed by a colon, a space, and then the secret code.

Sample Input

```
5
1 2 3 4 5 6 7 8 9 1
1 1 1 1 1 1 1 1 1 2
9 9 9 9 1 8 8 8 8 8
10 20 30 40 50 60 90 90 90 20
1 100 1 100 1 100 1 100 1 101
```

Sample Output

```
1: 28
2: 7
3: 29
4: 277
5: 206
```

Grading Rubric

Criteria	0	5	10
Compiling the Source Code on the Eustis Server *The solution developed incorporated only concepts covered in class (except those explicitly allowed in the instructions).*	The source code could not be compiled on the Eustis server. Therefore, the entire submission is not graded anymore and will be marked as 0/100. **OR** The solution included code/s OR function/s not covered in class. Therefore, the entire submission is not graded anymore and will be marked as 0/100.	*(Not Applicable)*	The source code can be compiled on the Eustis server. **AND** The solution included code/s AND function/s that were covered in class.
Function Implementation *For each of the required functions. The max possible score will be reweighted to 80. Your score will be multiplied by 0.80.*	The function was not implemented.	The function was either unfinished (but can still be compiled) **OR** incorrect. However, some effort was exerted.	The function was correctly implemented.
Code Readability *Refer to the style guide here.*	The code was not very readable **AND** lacked meaningful comments. **OR** The required comment for homework submissions is missing.	Either the source code was not very readable because of poor indentation **OR** there was a lack of meaningful comments throughout the source code.	The source code was readable because proper indentation was followed. Further, relevant and meaningful comments were incorporated throughout the source code.