**First Solo Competition Answers**

Debugging Questions:

1. **Ans:** The issue causing incorrect output is in the loop of the calculateAverage function. The loop condition i <= size is incorrect. It should be i < size because array indexing starts from 0, so it should iterate from 0 to size - 1.
2. **Ans:** Even after fixing the loop issue, the program may output a lower than expected average due to the data types used in the calculation. In particular, the calculation sum / size might perform integer division, truncating the decimal part of the result. To fix this, we need to ensure that the division performs floating-point division.
3. **Ans:** The use of int for the sum variable in calculateAverage could lead to potential issues if the array contains very large numbers. This is because the maximum value that an int variable can hold is limited, and if the sum exceeds this maximum value, it will cause overflow, leading to undefined behavior.
4. **Ans:** To handle an empty array without resulting in undefined behavior or a crash, we need to check if the size of the array is zero before calculating the average. If the size is zero, we can return an appropriate value, such as NaN (Not a Number) or some sentinel value, to indicate that the average cannot be calculated.

The modified code is as below:

#include <stdio.h>

double calculateAverage(int numbers[], int size) {

if (size == 0) {

return 0.0;

}

double sum = 0.0;

for (int i = 0; i < size; i++) {

sum += numbers[i];

}

return sum / size;

}

int main() {

int numbers[] = {5, 10, 15, 20, 25};

int size = sizeof(numbers) / sizeof(numbers[0]);

double average = calculateAverage(numbers, size);

printf("Average: %.2f\n", average); // Print average with two decimal places

return 0;

}