

Topic 1.1: Introduction to Algorithms, Programming, and Compilers

Mini-FRQ — From Algorithm to Running Program

A new member of the Riverside Coding Club is writing a program to plan water for a hike. Use the two Java versions below to answer Parts A and B.

*Total points: 5. Use evidence from the sources. Allowed task verbs: **Complete** · **Describe** · **Determine** · **Explain** · **Identify** · **Trace** · **Write**.*

Evidence Sources

The programmer typed this and clicked Run, but it would not start.

Source 1 — Version that will not run

```
public class ClubWelcome {  
    public static void main(String[] args) {  
        System.out.println("Welcome to the Riverside Coding Club!")  
    }  
}
```

After fixing Source 1, the programmer wrote this. It runs and prints 16.

Source 2 — Version that runs but is wrong

```
int hikers = 12;  
int perBottle = 4;           // each bottle serves 4 hikers  
int bottlesNeeded = hikers + perBottle;  
System.out.println(bottlesNeeded); // prints 16, but the club has 12 hikers
```

Questions

Part A. Algorithms and sequencing.

- i. Write** a numbered-step algorithm, in plain language, that computes how many full bottles of water are needed if each bottle serves 4 hikers and there are a given number of hikers. Sequencing must be clear. (2 points)

Part B. Programming errors. Use Source 1 and Source 2.

i. Identify the type of error in Source 1 and name the tool that reports it. *(1 point)*

ii. Identify the type of error in Source 2, given that it runs but prints 16 for 12 hikers. *(1 point)*

iii. Explain why a program can pass the compiler and still be wrong, using these two sources as evidence. *(1 point)*