

Topic 1.1: Introduction to Algorithms, Programming, and Compilers

Application Worksheet — Topic 1.1

Complete all four parts. Write in complete sentences where asked.

Part 1 — Match the term to its meaning

Write the letter of the best match next to each term.

- Algorithm — _____
- Compiler — _____
- Syntax error — _____
- Logic error — _____

Choices: (A) A broken language rule the compiler catches before the program runs. (B) A step-by-step process for solving a problem. (C) Checks code for certain errors and translates it so it can run. (D) The program runs but produces the wrong result.

Part 2 — Write an algorithm (worked → guided → on your own)

(a) Worked example. Study this algorithm for finding the largest of three different test scores. Notice each step is unambiguous and the order matters.

Worked example — largest of three scores

1. Look at the first score; call it the largest so far.
2. Compare the second score to the largest so far.
3. If the second score is bigger, it becomes the largest so far.
4. Compare the third score to the largest so far.
5. If the third score is bigger, it becomes the largest so far.
6. The largest so far is the answer.

(b) Your turn — finish it. Complete the two missing steps so this computes the total cost of buying N items at a fixed price each.

Complete steps 2 and 3

1. Start with the number of items, N, and the price per item.
2. _____
3. _____

(c) On your own. Write a numbered-step algorithm — or draw a simple flowchart — that decides whether a student passed, where passing means a score of 60 or higher. Use the box.

Follow-up: Why does the order of the steps matter? Use the word *sequencing*.

Part 3 — Put the build process in order

These four steps describe how source code becomes a running program, but they are scrambled. Number them 1-4 in the correct order.

Number each line 1-4 (they are shown out of order)

_____	The program runs and produces output.
_____	You write the source code in an editor or IDE.
_____	The compiler checks the code and reports any errors it finds.
_____	You fix any errors the compiler reported.

Part 4 — Trace, predict, and classify

For each Java segment, follow the prompts, then name the error type (syntax, logic, or run-time) and how a programmer would discover it.

Segment 1. Will this compile? Why or why not?

Segment 1

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

Segment 2. There are 12 hikers, and each bottle serves 4.

Segment 2 — it compiles and runs

```
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
```

Predict the output of Segment 2: _____

Actual output (after running): _____

Did your prediction match? YES / NO — if not, what did you miss?

Now trace the variables line by line to check your prediction:

Line executed	hikers	perBottle	bottlesNeeded	Output

Segment 3. What happens when this runs with the values shown?

Segment 3

```
int hikers = 12;
int groups = 0;           // no groups were entered yet
int perGroup = hikers / groups; // divide by zero at run time
System.out.println(perGroup);
```

Predict the output of Segment 3: _____

Actual output (after running): _____

Did your prediction match? YES / NO — if not, what did you miss?