

# Task 7 Programming Skill Guide

Task 7: Programming Practical Test - Unit 4 - 7.5%

## Task 7 Programming Skill Guide

This guide helps you prepare for the programming practical test. The goal is not to memorise one answer. The goal is to be able to read a short scenario, build the logic, test it and explain it.

### The Five-Minute Reading Routine

When the test starts, spend the first few minutes marking up the task:

1. Circle the **inputs** the user must enter.
2. Underline the **processing** the program must perform.
3. Box the **outputs** the program must display.
4. Mark each required **decision** with `IF`.
5. Mark each repeated action with `LOOP`.
6. Write the main variable names before coding.

### Input, Processing And Output

| Program part | What it means                                        | Example                                        |
|--------------|------------------------------------------------------|------------------------------------------------|
| Input        | Data entered by the user or provided to the program. | study minutes, score, name, menu choice        |
| Processing   | Work done by the program.                            | calculate total, compare score, update counter |
| Output       | Information shown by the program.                    | total time, feedback, final score              |

Strong practical work keeps these three parts connected. If the program asks for a value but never uses it, the logic is weak. If the program calculates a value but never outputs it, the user cannot benefit from it.

### Data Types

| Data type | Use when                         | Example values                                   |
|-----------|----------------------------------|--------------------------------------------------|
| Integer   | The value is a whole number.     | 0, 12, 45                                        |
| Real      | The value may include a decimal. | 3.5, 72.25                                       |
| Boolean   | The value is true or false.      | <code>isBalanced</code> , <code>hasPassed</code> |
| Character | A single character is stored.    | <code>Y</code> , <code>N</code> , <code>A</code> |

In many practical tests, student names and longer labels are stored as text/string values in the programming language even though the syllabus list names character as the core data type. Use the type supported by the class programming environment and explain the purpose clearly.

## Variable Naming

Good variable names explain what is stored:

- theoryMinutes
- practiceMinutes
- totalMinutes
- balancedCount
- menuChoice

Weak variable names hide meaning:

- x
- thing
- num1
- stuff

## Control Structures

### Sequence

Sequence means the instructions run in order.



### One-Way Selection

One-way selection performs an action only if the condition is true.



### Two-Way Selection

Two-way selection chooses between two paths.



### Multi-Way Selection

Multi-way selection chooses between more than two paths.



## Fixed Iteration

Fixed iteration repeats a known number of times.



## Test-First Iteration

Test-first iteration checks the condition before the loop body runs.



## Test-Last Iteration

Test-last iteration runs at least once before checking the condition.



## Testing

| Test data type | Purpose                              | Example for allowed score 0-10 |
|----------------|--------------------------------------|--------------------------------|
| Normal         | Checks expected valid input.         | 6                              |
| Boundary       | Checks values at or near the edge.   | 0, 10, -1, 11                  |
| Invalid        | Checks data that should be rejected. | text, blank, negative number   |

## Trace Tables

A trace table tracks variable values as an algorithm runs. Use it to find logic errors.

Example algorithm:



Trace table:

| count | minutes | total |
|-------|---------|-------|
| start | -       | 0     |
| 1     | 10      | 10    |
| 2     | 15      | 25    |
| 3     | 20      | 45    |

# Error Types

| Error type     | What it means                                 | Example                                                   |
|----------------|-----------------------------------------------|-----------------------------------------------------------|
| Syntax error   | The code breaks language rules.               | missing bracket, misspelled keyword                       |
| Run-time error | The program starts but crashes while running. | division by zero, invalid conversion                      |
| Logical error  | The program runs but gives the wrong result.  | using <code>&lt;</code> when <code>&lt;=</code> is needed |

## Practical Test Advice

- Build the simplest working version first.
- Test after each feature.
- Use comments to explain decisions, calculations and loop purpose.
- If a feature partly works, leave evidence of what works.
- If you cannot fix an error, write what the error is and what you tried.