OCR GCSE Computer Science J277 – Unit 2.3: Producing Robust Programs

Introduction

This revision sheet summarises the key concepts for Unit 2.3: Producing Robust Programs. It explains ways to make code more reliable, maintainable, and protected from errors or misuse.

Defensive Design

- Defensive design anticipates how a program might be misused or go wrong.
- Includes input validation, authentication, and planning for unexpected inputs.

Input Validation

- Ensures that data entered by the user is reasonable before processing.
- Common validation types:
  - Range check: Data is within a set range.
  - Length check: Data has the correct number of characters.
  - Presence check: Data has actually been entered.
  - Type check: Data is the correct type (e.g. number).

Authentication

- Checks a user's identity before giving access to a system.
- Can include usernames and passwords, biometric scans, or two-factor authentication.

Maintainability

- Code should be written so that it is easy to update and understand.
- Techniques include:
  - Comments to explain what the code does.
  - Meaningful variable names.
  - Indentation to show structure.
  - Using constants instead of hardcoding values.

Testing

- Testing helps identify and fix problems in the code.
- Types of test data:
  - Normal: Within expected range.
  - Boundary: At the edges of the valid range.
  - Invalid: Outside the valid range.
- A test plan includes input, expected output, and reason for test.

Types of Errors

- Syntax error: Breaks the rules of the programming language.
- Logic error: Code runs but does not do what was intended.
- Runtime error: Occurs while the program is running (e.g. divide by zero).

Final Checklist

- Understand the purpose and use of input validation.
- Know different authentication methods.
- Be able to write maintainable code.
- Identify syntax, logic, and runtime errors.
- Write and analyse test plans.