AP COMPUTER SCIENCE PRINCIPLES • EXERCISE SHEET

1.4 Identifying and Correcting Errors

Name: _____ Class: _____ Date: _____ **Total: 9 marks**

Objective

Build the skills to answer exam questions on **identifying and correcting errors**.

You must be able to:

- distinguish a **logic error** 逻辑错误, a **syntax error** 语法错误, a runtime error, and an overflow error
- use **testing** 测试 with varied inputs, including **edge cases** 边界情况
- locate and fix a defect during **debugging** 调试
- verify the corrected program produces the intended output

1 Worked examples

Study these first. Each one shows the method for a question type used later.

Types of error

- **Syntax error** —breaks the language's grammar, so it will not run.
- **Logic error** —runs but gives the **wrong** result.
- **Runtime error** —crashes while running (e.g. dividing by zero).
- **Overflow error** —a value is too large for the bits available.

Testing and debugging

Test with varied inputs, including **edge cases** (boundaries and unusual values). To find a bug, **hand-trace** the code or add temporary **print statements**, then re-test to verify the fix.

2 Practice

2.1 State the difference between a syntax error and a logic error. [2]

2.2 State what edge cases are used for in testing. [1]

2.3 Name one debugging strategy. [1]

---

3 Exam-style questions

3.1 A program runs but produces the wrong answer. This is a [1]

- **A** syntax error
- **B** logic error
- **C** overflow error
- **D** case with no error

3.2 Testing a program with unusual or boundary inputs uses [1]

- **A** edge cases
- **B** comments
- **C** variables
- **D** loops

3.3 A program crashes when it tries to divide by zero.

(a) Name the type of error. [1]

(b) Name one way to find the bug. [1]

(c) State what you should check after fixing it. [1]

4 Go further

- work through the **1.4 Identifying and Correcting Errors** lesson on the **Learn** page;
- read the **Creative Development** section of the AP Computer Science Principles handout on the **Know** page.

Solutions

2.1 a syntax error breaks the language's rules so the program will not run; a logic error runs but gives the wrong result.

2.2 to test the program at boundaries and unusual values where defects often hide.

2.3 hand-tracing the code, or adding print statements / using a debugger (any one).

3.1 B.

3.2 A.

3.3 (a) a runtime error. (b) hand-trace or add print statements to find where it fails (any one). (c) that it now produces the intended output for its requirements.