2.3 Producing Robust Programs: 10-minute Pack

OCR J277 · Calculators not allowed · about 10 minutes

Total Marks

Name: _____ Date: ____ / ____ / _____

Answer ALL questions. Show all your working.

1 A programmer is choosing the most suitable input validation check for different fields on an online form.

(a) Which check is most suitable to make sure a UK postcode is entered in the correct pattern, e.g. SW1A 1AA? **(1)**

- ☐ A) Presence check
- ☐ B) Format check
- ☐ C) Range check
- ☐ D) Length check

(b) Which check ensures that a customer's date of birth field is not left empty? **(1)**

- ☐ A) Presence check
- ☐ B) Format check
- ☐ C) Range check
- ☐ D) Check digit

(c) A 'number of tickets' field must only accept whole numbers from 1 to 8. Which check identifies that 12 has been entered? **(1)**

- ☐ A) Length check
- ☐ B) Format check
- ☐ C) Range check
- ☐ D) Presence check

(Total for Question 1 is 3 marks)

© Revision Library | revisionlibrary.co.uk

2 Study the following pseudocode, which calculates the mean of five numbers stored in an array:

```
total <- 0
count <- 0
FOR i <- 1 TO 5
total <- total + n[i]
count <- count + 1
NEXT i
a <- total / count
OUTPUT a
```

(a) State two maintainability techniques, other than adding comments, that could be used to improve the readability of this code without changing what it does. **(2)**

(b) Rewrite the line 'a <- total / count' using a more meaningful identifier name for a. **(1)**

(c) The array n stores a list of numbers. Suggest one more suitable identifier name for this array. **(1)**

(Total for Question 2 is 4 marks)

---

3 A programmer runs their test plan. One of the tests has failed: the expected result and the actual result produced by the program do not match. Describe the steps the programmer should take, after this failed test, to resolve the problem.

(Total for Question 3 is 3 marks)

© Revision Library | revisionlibrary.co.uk

Mark scheme · 2.3 Producing Robust Programs: 10-minute Pack

Question 1

(a) B) Format check **B1**

(a) Answer: B) Format check

(b) A) Presence check **B1**

(b) Answer: A) Presence check

(c) C) Range check **B1**

(c) Answer: C) Range check

Question 2

(a) Consistent/correct indentation of the FOR loop body **B1**

(a) Using meaningful/descriptive identifier names instead of a, n and i **B1**

(a) Answer: Indenting the lines inside the FOR loop; using meaningful identifier names instead of a, n and i.

(b) Any meaningful name replacing a, e.g. average <- total / count **B1**

(b) Answer: average <- total / count

(c) Any suitable meaningful name, e.g. numbers, scores or marks **B1**

(c) Answer: e.g. numbers

Question 3

Use IDE debugging tools (e.g. breakpoints, single stepping) or trace the code to locate the line(s) where the actual result diverges from the expected result **M1**

Identify and correct the fault/logic error in the code **M1**

Re-run the failed test (and ideally the whole test plan) to confirm the fix works and no new faults have been introduced **A1**

Answer: Use IDE debugging tools such as breakpoints and single stepping (or trace the code by hand) to find exactly where the program's behaviour diverges from what was expected; correct the fault in the code; then re-run the failed test, and ideally the whole test plan again, to confirm the program now produces the expected result and that no new errors have been introduced.

© Revision Library | revisionlibrary.co.uk