OCR GCSE (J277)

# Module O: Robust programs

Workbooklet · 7 parts · 149 marks

O3 is IGCSE (Cambridge) only and not on J277.

## Contents


| Part | Topic | Marks |
|---|---|---|
| O1 | Defensive design: anticipating misuse | 21 |
| O2 | Input validation | 22 |
| O4 | Authentication in programs | 22 |
| O5 | Why we test, and iterative vs final testing | 20 |
| O6 | Syntax errors vs logic errors | 22 |
| O7 | Choosing test data | 21 |
| O8 | Writing a test plan | 21 |
| | **Total** | **149** |

MODULE O ROBUST PROGRAMS · O1

# Defensive design: anticipating misuse

Original practice questions · 21 marks · ~25 minutes · OCR J277 spec 2.3.1

Name _____ Date _____

**Instructions.** Answer all questions in the spaces provided. The marks for each part are shown in brackets [ ]. All code on this sheet is written in OCR Exam Reference Language. Where you write code, you may use OCR Exam Reference Language or a high-level programming language that you have studied. The total for this worksheet is **21 marks**.

## 1

**Total: 4 marks**

Anticipating misuse is one of the defensive design considerations a programmer should think about.

**(a)** State what is meant by **anticipating misuse** when designing a program. **[2]**

**(b)** A council's staff portal asks each member of staff for a password. Someone may try to get into another person's account by guessing the password over and over again.
Describe **one** way the programmer could design the program to anticipate this misuse. **[2]**

**2** **Total: 6 marks**

A bike hire kiosk at a railway station runs a program. The customer is asked to enter:
- the number of hours of hire: a whole number from 1 to 8
- the type of bike, chosen from a menu: 1 = city bike, 2 = electric bike, 3 = tandem
- their surname, so the booking can be found later.

**(a)** Complete the table. For each input, give **one** example of an invalid value that a customer might enter and that the program should anticipate. **[3]**

| Input | Example of an invalid value |
|---|---|
| Number of hours of hire | |
| Type of bike (menu choice) | |
| Surname | |

**(b)** Describe how the program should deal with an invalid value when a customer enters one. **[3]**

**3** **Total: 6 marks**

An environmental agency measures the depth of a river in metres at several points. This program works out the average depth.

```
01 total = 0
02 count = int(input("How many readings?"))
03 for i = 1 to count
04     depth = float(input("Enter depth in metres"))
05     total = total + depth
06 next i
07 average = total / count
08 print("Average depth: " + str(average))
```

**(a)** State the value of `count` that would make line 07 stop the program with an error. **[1]**

**(b)** Give **one other** value that a user could enter for `count` that the program does not anticipate. **[1]**

**(c)** Write code to replace line 02. Your code must: **[4]**

- ask the user how many readings there are
- if the value is less than 1, output an error message and ask again
- keep asking, as many times as needed, until a value of 1 or more is entered.

You may assume that the user always enters a whole number.

**4** **Total: 5 marks**

A snack vending machine has six slots, numbered 1 to 6 on the front of the machine. The names of the snacks are stored in an array called `snacks`. The array is 0-indexed, so the snack in slot 1 is stored in `snacks[0]`.

```
01 array snacks = ["Crisps", "Apple", "Flapjack", "Nuts", "Raisins", "Popcorn"]
02 slot = int(input("Enter slot number 1 to 6"))
03 print("You chose " + snacks[slot - 1])
```

**(a)** A customer enters **7**. Explain why line 03 would cause an error. **[2]**

**(b)** Write code to replace line 03 so that the program: **[3]**

- outputs the name of the chosen snack only if the slot number is from 1 to 6
- outputs an error message for any other slot number.

**Total for this worksheet: 21 marks**

MODULE O ROBUST PROGRAMS · O2

# Input validation

Original practice questions · 22 marks · ~25 minutes · OCR J277 spec 2.3.1

Name _____ Date _____

**Instructions.** Answer all questions in the spaces provided. The marks for each part are shown in brackets [ ]. The total for this worksheet is **22 marks**. All code on this sheet is written in OCR Exam Reference Language.

## 1

Total: 3 marks

**(a)** Tick (✓) **one** box in each row to show whether each statement about input validation is true or false. **[3]**

| Statement | True | False |
|---|---|---|
| Input validation automatically checks that data entered is sensible and follows set rules before the program accepts it. | | |
| If an input passes validation, it must be correct. | | |
| A type check could reject the input `six` when a whole number is expected. | | |

## 2

Total: 6 marks

A community tool library lends garden and DIY tools to its adult members. Members use a program on a kiosk to borrow tools. The program validates each input before it is accepted.

**(a)** Complete the table by identifying the most suitable validation check for each rule. **[4]**

| Input | Rule | Validation check |
|---|---|---|
| Member number | Two capital letters followed by four digits, for example `TL4827` | |
| Number of days to borrow the tool | A whole number from 1 to 14 inclusive | |
| Contact phone number | Exactly 11 characters | |
| Name of the tool to search for | Must not be left blank | |

**(b)** A member enters 20 as the number of days to borrow a tool. **[2]**
Describe how the program should deal with this input.

## 3

**Total: 4 marks**

An escape room takes bookings for teams of 2 to 6 players. This program is meant to keep asking for the number of players until a value from 2 to 6 is entered. It contains a logic error.

```
01 players = int(input("How many players? (2 to 6)"))
02 while players < 2 AND players > 6
03     print("Team size must be 2 to 6")
04     players = int(input("How many players? (2 to 6)"))
05 endwhile
06 print("Booking for " + str(players) + " players")
```

**(a)** Identify the line number that contains the logic error. **[1]**

**(b)** A user enters 9 the first time they are asked. State what the program, as written above, outputs. **[1]**

**(c)** Write a corrected version of the line you identified in part (a). **[1]**

**(d)** Explain why the input on line 04 is inside the loop. **[1]**

## 4

**Total: 3 marks**

A gym locker lets an adult member choose their own code. The program checks the length of the code.

```
01 do
02     code = input("Enter locker code")
03     if code.length != 4 then
04         print("Code must be 4 characters")
05     endif
06 until code.length == 4
07 print("Locker set")
```

**(a)** A member enters these three codes, one each time the program asks: `73`, then `73915`, then `7391`. Complete the table to show what the program outputs after each code is entered. Write **nothing** in a box if there is no output. **[3]**

| Code entered | Output |
|---|---|
| 73 | |
| 73915 | |
| 7391 | |

**5** **Total: 6 marks**

A community radio station has an online message board. Each listener chooses a username when they join. A username is valid only if it:

- is not left blank
- has no more than 12 characters.

**(a)** Write a program that: **[6]**

- asks the user to enter a username
- validates the username using the two rules above
- outputs an error message and asks for the username again each time the username is invalid
- outputs `Welcome` followed by the username once a valid username has been entered.

You must use either: OCR Exam Reference Language, or a high-level programming language that you have studied.

**Total for this worksheet: 22 marks**

MODULE O ROBUST PROGRAMS · O4

# Authentication in programs

Original practice questions · 22 marks · ~25 minutes · OCR J277 specification 2.3.1 (with 2.1.2 and 2.2.3)

Name _____ Date _____

**Instructions.** Answer all questions in the spaces provided. The marks for each part are shown in brackets [ ]. All questions are about a box-office program used by the staff of a small theatre. The total for this worksheet is **22 marks**.

## 1

**Total: 4 marks**

The theatre's box-office program uses defensive design.

**(a)** State what is meant by **authentication**. **[1]**

**(b)** Tick (✓) **one** box in each row to show whether each check is an example of authentication or of input validation. **[3]**

| Check carried out by the program | Authentication | Input validation |
|---|---|---|
| The number of seats booked must be between 1 and 8 | | |
| A staff member must enter their own password again before a booking can be deleted | | |
| The username and password entered are compared with the ones stored for that staff member | | |

**2** **Total: 5 marks**

Staff log in to the program with a username and password. The first version of the login code is shown in **Fig. 1**.

```
01 const storedUser = "boxoffice2"
02 const storedPass = "Curtain#47"
03 user = input("Enter username")
04 pwd = input("Enter password")
05 if user == storedUser AND pwd == storedPass then
06     print("Access granted")
07 else
08     print("Access denied")
09 endif
```

**Fig. 1**

**(a)** Complete the table to show the output from Fig. 1 for each set of inputs. **[3]**

| Username entered | Password entered | Output |
|---|---|---|
| boxoffice2 | Curtain#47 | |
| boxoffice2 | Curtain#74 | |
| boxoffice3 | Curtain#47 | |

**(b)** A programmer changes `AND` to `OR` on line 05. Describe the effect this change would have on who can log in to the program. **[2]**

**3** **Total: 6 marks**

The program is improved so that a staff member has a limited number of attempts to enter the password. The new code is shown in **Fig. 2**.

```
01 attempts = 0
02 loggedIn = false
03 while attempts < 3 AND loggedIn == false
04     pwd = input("Enter password")
05     if pwd == "Curtain#47" then
06         loggedIn = true
07     else
08         print("Try again")
09     endif
10     attempts = attempts + 1
11 endwhile
12 if loggedIn == true then
13     print("Welcome")
14 else
15     print("Locked")
16 endif
```

**Fig. 2**

**(a)** Complete the trace table for Fig. 2 when the passwords entered are `Curtain47`, then `Curtain#74`, then `Curtain#47`. The first row has been completed for you. You may not need to use all the rows. **[4]**

| attempts | loggedIn | pwd | Output |
|---|---|---|---|
| 0 | false | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

**(b)** Explain why the program limits the number of attempts to enter the password. **[2]**

## 4

**Total: 5 marks**

The theatre now has 20 staff accounts. The usernames are stored in the global array `users` and the passwords in the global array `passwords`. The password for `users[i]` is stored in `passwords[i]`. Both arrays have 20 items and are indexed from 0. The first three items are shown below.

| Index | 0 | 1 | 2 |
|---|---|---|---|
| **users** | `"mpatel"` | `"rokafor"` | `"jlind"` |
| **passwords** | `"Velvet!29"` | `"Encore#81"` | `"Stalls$56"` |

**(a)** Complete the function `checkLogin()` so that it: **[5]**

- takes the username and password entered as the parameters `user` and `pwd`
- checks the usernames in the array `users`
- returns `true` if `user` matches a stored username **and** `pwd` matches the password stored at the same index
- returns `false` if no matching username and password pair is found.

You must use either OCR Exam Reference Language or a high-level programming language that you have studied.

```
function checkLogin(user, pwd)









endfunction
```

## 5

**Total: 2 marks**

Other than limiting the number of attempts, describe **one** way the programmer could make it harder for someone who is not a member of staff to log in to the box-office program. **[2]**

**Total for this worksheet: 22 marks**

MODULE O ROBUST PROGRAMS · O5

# Why we test, and iterative vs final testing

Original practice questions · 20 marks · ~25 minutes · OCR J277 spec 2.3.2

Name _____ Date _____

**Instructions.** Answer all questions in the spaces provided. The marks for each part are shown in brackets [ ]. The total for this worksheet is **20 marks**. All code on this sheet is written in OCR Exam Reference Language.

A small museum asks a programmer to write a program for its ticket kiosk. The program is made of separate modules: one works out the cost of the tickets, one takes the payment and one prints a receipt. All the questions on this worksheet are about this program.

## 1

Total: 4 marks

**(a)** The table describes four tests that the programmer carries out. Tick (✓) **one** box in each row to show whether each test is iterative testing or final/terminal testing. **[4]**

| Test | Iterative | Final/terminal |
|---|---|---|
| The cost module is tested as soon as it has been written, before the payment module is started. | | |
| The finished program is run on the kiosk to check that it does everything the museum asked for. | | |
| The receipt module is tested again after an error in it has been corrected. | | |
| Just before the kiosk opens to the public, the programmer checks that all three modules work together. | | |

## 2

Total: 2 marks

**(a)** State **two** purposes of testing the kiosk program. **[2]**

1

2

**3** **Total: 4 marks**

**(a)** Describe what is meant by **iterative** testing. **[2]**

**(b)** Describe what is meant by **final/terminal** testing. **[2]**

**4** **Total: 6 marks**

The museum's rule for group tickets is:

- groups of 10 or more people pay £9 per person
- smaller groups pay £12 per person.

The programmer writes this function for the cost module. The parameter `people` is always a whole number of 1 or more.

```
01 function groupCost(people)
02     if people > 10 then
03         cost = people * 9
04     else
05         cost = people * 12
06     endif
07     return cost
08 endfunction
```

As soon as the function is written, the programmer tests it on its own. The table shows the tests.

| Test | Value of `people` | Expected result (£) | Actual result (£) |
|---|---|---|---|
| 1 | 4 | 48 | 48 |
| 2 | 10 | | 120 |
| 3 | 25 | | 225 |

**(a)** Complete the table by writing the expected result for Test 2 and Test 3. **[2]**

**(b)** Identify the test that shows there is an error in the function. **[1]**

**(c)** Identify the line number that contains the error and write the corrected line of code. **[2]**

Line number .....

Corrected line

**(d)** State what the programmer should do after correcting the error. **[1]**

**5** **Total: 4 marks**

Every module of the kiosk program has passed iterative testing. The museum manager says that final/terminal testing is therefore not needed.

**(a)** Explain **two** reasons why the programmer should still carry out final/terminal testing on the kiosk program. **[4]**

1

2

**Total for this worksheet: 20 marks**

MODULE O ROBUST PROGRAMS · O6

# Syntax errors vs logic errors

Original practice questions · 22 marks · ~25 minutes · OCR J277 specification 2.3.2 (with 2.1.2)

Name _____ Date _____

**Instructions.** Answer all questions in the spaces provided. The marks for each part are shown in brackets [ ]. Where you write a corrected line of code, you may use OCR Exam Reference Language or a high-level programming language that you have studied. The total for this worksheet is **22 marks**.

## 1

Total: 4 marks

**(a)** Describe what is meant by a **syntax error**. **[2]**

**(b)** Describe what is meant by a **logic error**. **[2]**

## 2

Total: 4 marks

A café is writing a program for its loyalty cards. Each line of code below contains **one** error.

**(a)** Tick (✓) **one** box in each row to show whether the error is a syntax error or a logic error. **[4]**

| What the line should do | Line of code | Syntax error | Logic error |
|---|---|---|---|
| Output the message Card full | `print("Card full"` | | |
| Work out a 20% discount on the price | `discount = price * 0.02` | | |
| Check whether the customer has 10 or more stamps | `if stamps > 10 then` | | |
| End the if statement | `endfi` | | |

**3** **Total: 4 marks**

A bike-hire shop charges £8 for each hour a bike is hired. A programmer writes a program to work out the cost of a hire.

**(a)** When the programmer first tries to run the program, the translator reports an error on line 12 and the program does not run. **[1]**
State the type of error this shows.

**(b)** After this error is corrected, the program runs. A customer who hires a bike for 2 hours is charged £24. **[1]**
State the type of error this shows.

**(c)** Describe how the programmer could find the cause of the error in part (b). **[2]**

**4** **Total: 5 marks**

A laundrette charges £2 for each kilogram (kg) of washing, with a **minimum** charge of £6. Any charge below £6 is raised to £6. Weights are entered in whole kilograms.
The program below contains **one** syntax error and **one** logic error.

```
01 kg = int(input("Enter weight of washing in kg"))
02 charge = kg * 2
03 if charge > 6 then
04     charge = 6
05 endif
06 print("Charge: " + str(charge) + " pounds"
```

**(a)** Identify the line number of the **syntax** error and write the corrected line. **[2]**

| Line number | Corrected line |
|---|---|
| | |

**(b)** Identify the line number of the **logic** error and write the corrected line. **[2]**

| Line number | Corrected line |
|---|---|
| | |

**(c)** The programmer corrects **only** the syntax error. State the output when the weight **5** is entered. **[1]**

## 5

**Total: 5 marks**

An office air-quality monitor takes three readings. The program below should input the three readings and output their average (mean).

```
01 total = 0
02 for i = 1 to 3
03     reading = int(input("Enter reading"))
04     total = total + i
05 next i
06 average = total / 3
07 print(average)
```

**(a)** Complete the trace table when the readings **12**, **9** and **15** are entered, in that order. You may not need to use all the rows. **[3]**

| i | reading | total | average | Output |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

**(b)** The program contains a logic error. Identify the line number of the logic error and write the corrected line. **[2]**

**Total for this worksheet: 22 marks**

MODULE O ROBUST PROGRAMS · O7

# Choosing test data

Original practice questions · 21 marks · ~25 minutes · OCR J277 specification 2.3.2

Name _____ Date _____

**Instructions.** Answer all questions in the spaces provided. The marks for each part are shown in brackets [ ]. Use the OCR names for types of test data: **normal**, **boundary**, **invalid** and **erroneous**. Where you are asked for test data, give an actual value, not a description. The total for this worksheet is **21 marks**.

## 1

**Total: 3 marks**

**(a)** State what is meant by **normal** test data. **[1]**

**(b)** Explain the difference between **invalid** test data and **erroneous** test data. **[2]**

## 2

**Total: 5 marks**

An indoor climbing centre takes bookings for groups. The booking program asks for the group size. The group size must be a **whole number** from **2 to 12** inclusive.

**(a)** Complete the table. For each value of test data, write the type of test data and whether the program should **accept** or **reject** it. **[5]**

| Test data (group size) | Type of test data | Accept or reject? |
|---|---|---|
| 8 | | |
| 2 | | |
| 12 | | |
| 20 | | |
| "ten" | | |

**3** **Total: 4 marks**

A cycling club's website asks each new member to choose a username. A username must be from **6 to 12 characters** long (inclusive).

**(a)** Give **two** different usernames that are **boundary** test data and that should be accepted. **[2]**

**(b)** Give **one** username that is **invalid** test data for this rule. Explain why it is invalid test data rather than erroneous test data. **[2]**

**4** **Total: 5 marks**

A shop's fridge has a temperature setting. The setting must be a whole number from **1 to 8** inclusive. A programmer has written this code to validate the setting.

```
01 valid = false
02 while valid == false
03     setting = int(input("Enter setting (1 to 8)"))
04     if setting > 1 and setting <= 8 then
05         valid = true
06     else
07         print("Setting must be 1 to 8")
08     endif
09 endwhile
10 print("Setting saved")
```

**(a)** The program is run four times. A different value is entered each time: **5**, **8**, **1** and **9**. Identify the value that shows the program does **not** work correctly. **[1]**

**(b)** Describe what the program does when the value you identified in (a) is entered, and what it should do instead. **[2]**

**(c)** State the type of test data that found this error. **[1]**

**(d)** Give **one** example of **erroneous** test data for this program. **[1]**

**5** **Total: 4 marks**

A bakery's online ordering program asks how many loaves are wanted. Only whole numbers from **1 to 50** inclusive should be accepted. The programmer tests the program using only these three values: **10**, **20** and **30**.

**(a)** Explain why this testing is **not** enough. Include examples of other test data the programmer should use. **[4]**

**Total for this worksheet: 21 marks**

MODULE O ROBUST PROGRAMS · O8

# Writing a test plan

Original practice questions · 21 marks · ~25 minutes · OCR J277 specification 2.3.2

Name _____ Date _____

**Instructions.** Answer all questions in the spaces provided. All code is written in OCR Exam Reference Language. Use the OCR names for types of test data: **normal**, **boundary**, **invalid** and **erroneous**. Write actual values as test data, not descriptions. The marks for each part are shown in brackets [ ]. The total for this worksheet is **21 marks**.

## 1

**Total: 3 marks**

**(a)** A programmer writes a test plan before testing a program. **[2]**
Other than the test data, give **two** items of information that are recorded for each test in a test plan.

1

2

**(b)** State why each test in a test plan has an expected result. **[1]**

## 2

**Total: 5 marks**

A tool-hire shop uses a program to work out the cost of hiring a tool.

- The user enters the number of days, as a whole number.
- Only 1 to 14 days (inclusive) are allowed.
- If the number of days is allowed, the program outputs the cost, which is £12 per day.
- Otherwise, the program outputs `Enter 1 to 14 days` and asks again.

Complete the test plan for this program. **[5]**

| Test | Test data (days) | Type of test | Expected result |
|---|---|---|---|
| 1 | 6 | Normal | Accepted, outputs cost £72 |
| 2 | 14 | | |
| 3 | | Boundary | Accepted, outputs cost £12 |
| 4 | 0 | | Rejected, outputs `Enter 1 to 14 days` |
| 5 | | Erroneous | Rejected, outputs `Enter 1 to 14 days` |

## 3

**Total: 3 marks**

A parcel locker uses this function to decide which size of locker a parcel goes in. Weights, in kg, are checked before the function is called, so the function only receives weights above 0 and up to 30.

```
01 function lockerSize(weight)
02     if weight <= 2 then
03         size = "Small"
04     elseif weight <= 10 then
05         size = "Medium"
06     else
07         size = "Large"
08     endif
09     return size
10 endfunction
```

There are three routes through the `if` statement. **[3]**
Complete the test plan so that each route is tested once. Give a different weight for each test and the value the function should return.

| Test | Test data (weight in kg) | Expected result (value returned) |
|---|---|---|
| 1 | | |
| 2 | | |
| 3 | | |

**4** **Total: 6 marks**

A greenhouse controller reads the temperature in °C as a whole number.

- Readings from 0 to 50 (inclusive) are accepted. Any other reading outputs `Reading rejected`.
- For an accepted reading of 28 or more, the program outputs `Fan on`.
- For an accepted reading below 28, the program outputs `Fan off`.

```
01 temp = int(input("Enter temperature"))
02 if temp < 0 OR temp > 50 then
03     print("Reading rejected")
04 elseif temp > 28 then
05     print("Fan on")
06 else
07     print("Fan off")
08 endif
```

**(a)** The programmer runs the tests in this test plan on the code above. **[3]**
Complete the **Actual result** and **Pass/Fail** columns for tests 2, 3 and 4. Test 1 has been done for you.

| Test | Test data | Type of test | Expected result | Actual result | Pass/Fail |
|---|---|---|---|---|---|
| 1 | 35 | Normal | Fan on | Fan on | Pass |
| 2 | 28 | Normal | Fan on | | |
| 3 | 51 | Invalid | Reading rejected | | |
| 4 | 0 | Boundary | Fan off | | |

**(b)** Identify the line number that contains the error. Refine the line so that the program meets all three bullet points above. **[2]**

Line number

Refined line

**(c)** State what the programmer should do with the test plan after refining the code. **[1]**

**5** **Total: 4 marks**

A boat-tour company's booking program accepts group sizes from 2 to 10 people (inclusive), as a whole number. A new programmer wrote this test plan for the group size input.

| Test | Test data | Type of test | Expected result |
|---|---|---|---|
| 1 | 5 | Normal | It works |
| 2 | 7 | Normal | Accepted |
| 3 | 12 | Invalid | Error |

Identify **two** weaknesses in this test plan. For each weakness, describe how the test plan should be improved. **[4]**

Weakness 1

Improvement

Weakness 2

Improvement

**Total for this worksheet: 21 marks**