

# GCSE OCR

Computer Science  
J277

3

## Errors and testing

Unit 8  
Logic and languages



PG ONLINE

# Objectives

- Understand the purpose of testing including:
  - Iterative testing
  - Final/terminal testing
- Identify syntax and logic errors
- Select and use suitable test data including:
  - Normal
  - Boundary
  - Invalid
  - Erroneous

# Starter

```
function max(a,b,c)
    if a >= b AND a >= c
        return a
    elif b >= a AD b >= c
        return b
    else
        return a
    endif
endfunction
```

```
num1 = input("Enter number: ")
num2 = input("Enter number: ")
num3 = input("Enter number: ")

maxNum = max(num1, num2, nu3)

print "maximum number is: "
print(maxNum)
```

- What are the **six** mistakes in the code above

# Starter

## Answers

```
function max(a,b,c)
  if a >= b AND a > c
    return a
  elif b >= a AND b >= c
    return b
  else
    return a
  endif
endfunction
```

Diagram annotations for the first code block:

- A red box highlights the word "AND" in the first `if` condition, with a red line pointing to the word "AND" above it.
- A red box highlights the variable `a` in the `return a` statement of the first `if` branch, with a red line pointing to the variable `a` below it.
- A red box highlights the variable `a` in the `return a` statement of the `else` branch, with a red line pointing to the variable `a` below it.

```
num1 = input("Enter number: ")
num2 = input("Enter number: ")
num3 = input("Enter number: ")

maxNum = max(num1, num2, num3)

print("maximum number is: ")
print(maxNum)
```

Diagram annotations for the second code block:

- A red box highlights the opening quote character `"` at the end of the first `input` statement, with a red line pointing to the quote character above it.
- A red box highlights the opening quote character `"` at the end of the second `input` statement, with a red line pointing to the quote character above it.
- A red box highlights the variable `num3` in the `max` function call, with a red line pointing to the variable `num3` below it.
- A red box highlights the opening parenthesis `(` in the `print` statement, with a red line pointing to the parenthesis below it.





# Syntax errors

- A syntax error is one where the code written doesn't conform to the rules of the language
  - The compiler doesn't know how to translate the program into machine code so will give the programmer a syntax error
  - The program cannot be run until all syntax errors are fixed
  - Which of the errors in the starter were syntax errors?

```
function max(a,b,c)
  if a >= b AND a >= c
    return a
  elif b >= a AD b >= c
    return b
  else
    return a
  endif
endfunction
```

Diagram illustrating syntax errors in the code above:

- A vertical line labeled "AND" points to the word "AND" in the first `if` condition.
- A box around "AD" in the `elif` condition has a vertical line pointing to "c" below it.
- A box around "a" in the `return a` statement of the `else` block has a vertical line pointing to "a" below it.

```
num1 = input("Enter number: ")
num2 = input("Enter number: " )
num3 = input("Enter number: ")

maxNum = max(num1, num2, nu3) num3

print "maximum number is: "
print(maxNum)
```

Diagram illustrating syntax errors in the code above:

- A box around the closing parenthesis of the first `input` function call has a vertical line pointing to a double quote above it.
- A box around the opening parenthesis of the second `input` function call has a vertical line pointing to a double quote above it.
- A box around "nu3" in the `max` function call has a vertical line pointing to "num3" to its right.
- A box around the opening parenthesis of the `print` statement has a vertical line pointing to a double quote above it.
- A box around the opening parenthesis of the `print` statement has a vertical line pointing to a double quote above it.

# Syntax errors

Answers

- All the errors are syntax errors except return a should be return c
  - This is because they all don't conform to the grammar of the language, so the compiler doesn't know how to translate them

```
function max(a,b,c)
    if a >= b AND a >= c
        return a
    elif b >= a AD b >= c
        return b
    else
        return a
    endif
endfunction
```


```
num1 = input("Enter number: ")
num2 = input("Enter number: ")
num3 = input("Enter number: ")
maxNum = max(num1, num2, nu3) num3
print "maximum number is: "
print(maxNum)
(
```



# Logical errors

- The remaining error is a logical error
  - The program will run, but it won't work as the programmer intended
  - It will give the wrong maximum number if *c* is greatest

```
function max(a,b,c)
  if a >= b AND a >= c
    return a
  elif b >= a AD b >= c
    return b
  else
    return a
  endif
endfunction
```



```
num1 = input("Enter number: ")
num2 = input("Enter number: ")
num3 = input("Enter number: ")

maxNum = max(num1, num2, nu3)

print "maximum number is: "
print(maxNum)
```



# Worksheet 3

- Now complete **Task 1** on **Worksheet 3**



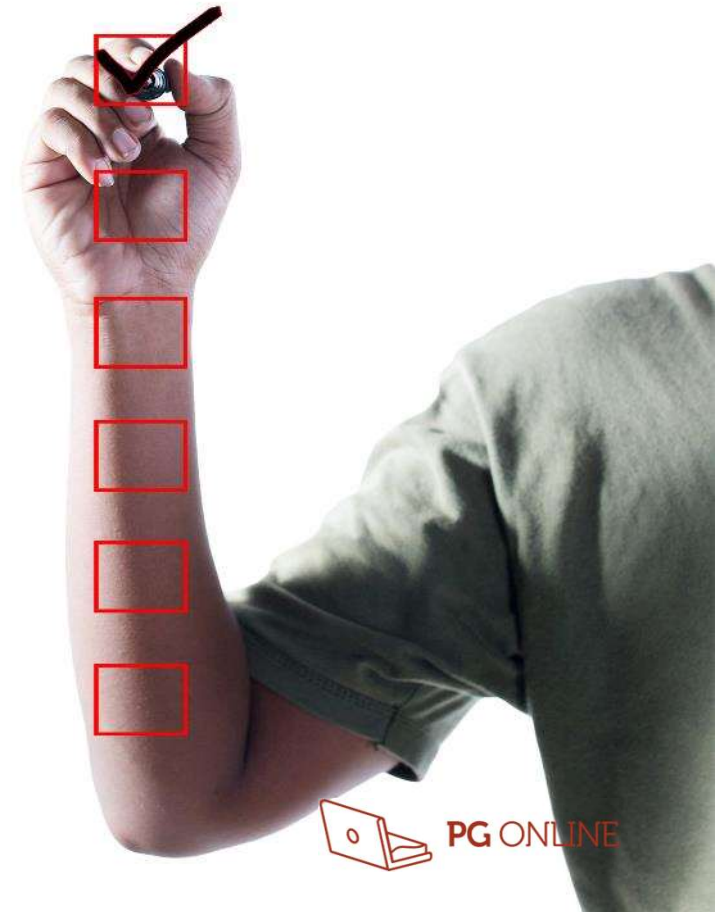


# Testing

- It is important that programs are fully tested to find logical errors
- Before testing, a test plan is created which will test for:
  - Normal data
  - Boundary data
  - Invalid data
  - Erroneous data
- What do you think each of these types of test mean?

# Test data

- Suppose the user is asked to enter a number between 1 and 100
  - What test data would you use to test your program?



# Tests

Answers

- Program: enter a number between 1 and 100
- Some examples of test data:
  - **Normal data:** 5 (checks a single digit), 14 (checks two digits)
  - **Boundary data:** 1 (checks lowest valid data), 100 (checks highest valid data)
  - **Invalid data:** -50 (checks negative numbers), 173 (checks data of the correct type that is invalid), 0 (checks just below lower boundary), 101 (checks just above upper boundary)
  - **Erroneous:** "ade", "#\$" (checks data of the wrong type is rejected)



# Finding logic errors

- A **trace table** is useful for tracing through a program in order to find a logic error
  - The value of each variable is recorded as it changes
- Another useful technique is to insert **print** statements at various points in the program to check the values of variables

# Using a trace table

- The value of each variable is recorded as it changes
- What value is output in the code below?

```
num = 3
n = 0
while n < 4
    num = num + n
    n = n + 1
endwhile
output(num)
```

num	n	n < 4	OUTPUT
3	0	TRUE	



# Using a trace table

- The value of each variable is recorded as it changes
- What value is output in the code below?

```
num = 3
n = 0
while n < 4
    num = num + n
    n = n + 1
endwhile
output(num)
```

num	n	n < 4	OUTPUT
3	0	TRUE	
3	1	TRUE	

# Using a trace table

- The value of each variable is recorded as it changes
- What value is output in the code below?

```
num = 3
n = 0
while n < 4
    num = num + n
    n = n + 1
endwhile
output(num)
```

num	n	n < 4	OUTPUT
3	0	TRUE	
3	1	TRUE	
4	2	TRUE	

# Using a trace table

- The value of each variable is recorded as it changes
- What value is output in the code below?

```
num = 3
n = 0
while n < 4
    num = num + n
    n = n + 1
endwhile
output(num)
```

num	n	n < 4	OUTPUT
3	0	TRUE	
3	1	TRUE	
4	2	TRUE	
6	3	TRUE	



# Using a trace table

- The value of each variable is recorded as it changes
- What value is output in the code below?

```
num = 3
n = 0
while n < 4
    num = num + n
    n = n + 1
endwhile
output(num)
```

num	n	n < 4	OUTPUT
3	0	TRUE	
3	1	TRUE	
4	2	TRUE	
6	3	TRUE	
9	4	FALSE	9



# Creating a trace table

- A trace table is useful for
  - Determining the purpose of an algorithm
  - Finding the output of an algorithm
  - Finding errors in an algorithm
- To draw a trace table, make a column for each variable used, in the order in which they appear
- You don't need to fill in a value for a variable which does not change in a particular row



# Determine the function of an algorithm

- Complete the trace table for the algorithm and state its function:
  - Assume the user enters values 8, 3, 6, 5, 10, 6

```
total = 0
for i = 1 to 3
  base = input()
  height = input()
  x = (base * height)/2
  total = total + x
next i
result = total / 3
print(result)
```

total	i	base	height	x	output
0	1	8	3	12	



# Determining the function of this algorithm

- Complete the trace table for the algorithm and state its function:
  - Assume the user enters values 8, 3, 6, 5, 10, 6

```
total = 0
for i = 1 to 3
  base = input()
  height = input()
  x = (base * height)/2
  total = total + x
next i
result = total / 3
print(result)
```

total	i	base	height	x	output
0	1	8	3	12	
12	2	6	5	15	



# Determining the function of this algorithm

- Complete the trace table for the algorithm and state its function:
  - Assume the user enters values 8, 3, 6, 5, 10, 6

```
total = 0
for i = 1 to 3
  base = input()
  height = input()
  x = (base * height)/2
  total = total + x
next i
result = total / 3
print(result)
```

total	i	base	height	x	output
0	1	8	3	12	
12	2	6	5	15	
27	3	10	6	30	

# Determining the function of this algorithm

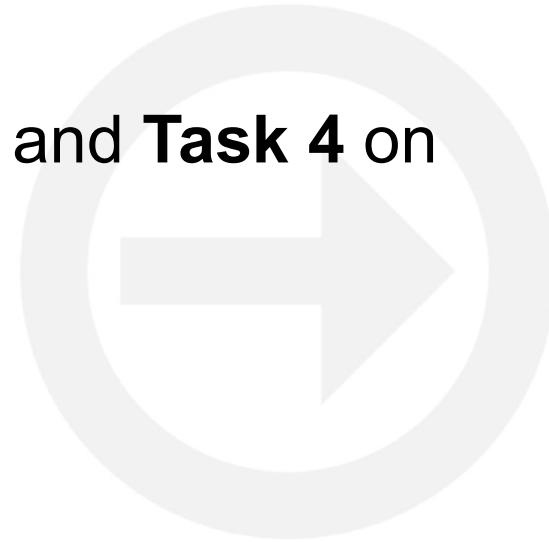
- The algorithm finds the average of the areas of the triangles
  - Assume the user enters values 8, 3, 6, 5, 10, 6

```
total = 0
for i = 1 to 3
  base = input()
  height = input()
  x = (base * height)/2
  total = total + x
next i
result = total / 3
print(result)
```

total	i	base	height	x	output
0	1	8	3	12	
12	2	6	5	15	
27	3	10	6	30	
57	4				19

# Worksheet 3

- Now complete **Task 2**, **Task 3** and **Task 4** on **Worksheet 3**





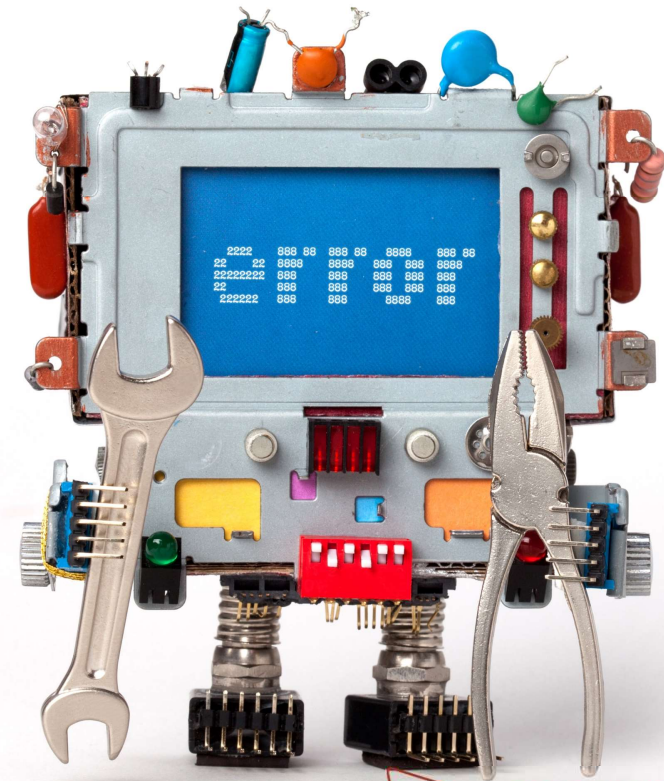
# Iterative and final testing

- There are two types of testing:
  - **Iterative testing** tests modules and parts of a program as the program is developed
  - **Final testing** (also known as terminal testing) tests the whole program at the end of production
- In iterative testing, the programmer will usually test the code with knowledge of how it works
  - In final testing, the program may be tested by the programmer or end user – as they are testing the whole program, they won't necessarily know how it works



# Plenary

- In a pair, answer the following questions
  - Name **four** types of test data that can be used when testing
  - Explain what a syntax error is
  - Give **three** examples of a syntax error
  - Explain with an example what a logic error is



# Plenary

Answers

- Name **four** types of test data:
  - Normal, boundary, invalid, erroneous
- Explain what a syntax error is:
  - An error where the programming code written doesn't conform to the rules/grammar of the language
- Give **three** examples of a syntax error:
  - `print("hello)`      `14 = age`      `remainder = 10 MD 3`
- Explain with an example what a logic error is:
  - Where the program compiles/runs but it doesn't do what the programmer intended



## **Copyright**

© 2020 PG Online Limited

The contents of this unit are protected by copyright.

This unit and all the worksheets, PowerPoint presentations, teaching guides and other associated files distributed with it are supplied to you by PG Online Limited under licence and may be used and copied by you only in accordance with the terms of the licence. Except as expressly permitted by the licence, no part of the materials distributed with this unit may be used, reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic or otherwise, without the prior written permission of PG Online Limited.

## **Licence agreement**

This is a legal agreement between you, the end user, and PG Online Limited. This unit and all the worksheets, PowerPoint presentations, teaching guides and other associated files distributed with it is licensed, not sold, to you by PG Online Limited for use under the terms of the licence.

The materials distributed with this unit may be freely copied and used by members of a single institution on a single site only. You are not permitted to share in any way any of the materials or part of the materials with any third party, including users on another site or individuals who are members of a separate institution. You acknowledge that the materials must remain with you, the licencing institution, and no part of the materials may be transferred to another institution. You also agree not to procure, authorise, encourage, facilitate or enable any third party to reproduce these materials in whole or in part without the prior permission of PG Online Limited.