

# GCSE OCR

Computer Science  
J277

6

**Interpret, correct  
and complete  
algorithms**

Unit 6  
Algorithms



PG ONLINE



# Objectives

- Understand the purpose of a given algorithm and how an algorithm works
- Understand how to determine the correct output of an algorithm for a given set of data
- Understand how to identify and correct errors in algorithms
- Create and use of trace tables to follow an algorithm

# Starter

- When programming, it is common to make mistakes
  - What techniques can a programmer use to find mistakes in their programs?



# Starter

Answers

- There are a number of ways
  - The first is to **use a trace table** – this is where the programmer goes through the code, line by line, writing down the values of variables
  - Alternatively **IDEs (Integrated development environments) can be used**
  - These have the ability to set breakpoints, step through code and watch variables. This is similar to trace tables and allows the computer to do the processing



# Trace tables

- Sometimes you may be given an algorithm in the form of a flowchart or pseudocode, and asked to determine its purpose
- One way to do this is to use a **trace table**
  - Trace tables are used to determine the outputs from a program as it runs
  - They enable a programmer to find errors in their programs

# Using a trace table

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

```
num = 3
n = 0
while n < 4
    num = num + n
    n = n + 1
endwhile
print(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 from the code below?

```
num = 3
n = 0
while n < 4
    num = num + n
    n = n + 1
endwhile
print(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 from the code below?

```
num = 3
n = 0
while n < 4
    num = num + n
    n = n + 1
endwhile
print(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 from the code below?

```
num = 3
n = 0
while n < 4
    num = num + n
    n = n + 1
endwhile
print(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 from the code below?

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





# Determining 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 count = 1 to 3
  base = input()
  height = input()
  area = (base*height)/2
  total = total + area
next count
result = total / 3
print(result)
```

total	count	base	height	x	output
0	1	8	3	12	

# Determining 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 count = 1 to 3
  base = input()
  height = input()
  area = (base*height)/2
  total = total + area
next count
result = total / 3
print(result)
```

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

# Determining 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 count = 1 to 3
  base = input()
  height = input()
  area = (base*height)/2
  total = total + area
next count
result = total / 3
print(result)
```

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



# Determining the function of an algorithm

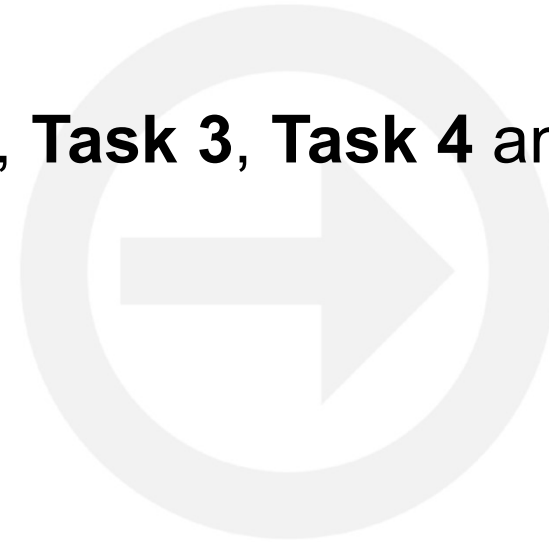
- The algorithm finds the average of the areas of the triangles
  - Assume the user enters values 8, 3, 6, 5, 10, 6

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

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

# Worksheet 6

- Now complete **Task 1, Task 2, Task 3, Task 4** and **Task 5** on **Worksheet 6**



# Finding errors

- Trace tables are often used to find errors
  - The Fibonacci sequence takes the previous two numbers to find the next number – e.g. 0, 1, 1, 2, 3, 5, 8, 13...
  - Look at the following code and complete the trace table for it

```
num1 = 0
num2 = 1
print(num1)
print(num2)
for i = 1 to 5
    newNum = num1 + num2
    print(newNum)
    num1 = num2
    num2 = num1 + num2
endfor
```

i	newNum	num1	Num2	output
		0	1	0 1



# Finding errors

Answers

- The trace table shows the output will be:  
**0, 1, 1, 3, 6, 12, 24** – it should be 0, 1, 1, 2, 3, 5, 8
  - Change one line of code below to fix the problem
  - Create another table and check your algorithm

```
num1 = 0
num2 = 1
print(num1)
print(num2)
for i = 1 to 5
    newNum = num1 + num2
    print(newNum)
    num1 = num2
    num2 = num1 + num2
endfor
```

i	newNum	num1	Num2	output
		0	1	0 1
1	1	1	2	1
2	3	2	4	3
3	6	4	8	6
4	12	8	16	12
5	24	16	32	24



# Finding errors

Answers

- The trace table shows the output will be:  
**0, 1, 1, 3, 6, 12, 24** – it should be 0, 1, 1, 2, 3, 5, 8
  - Change one line of code below to fix the problem
  - Create another table and check your algorithm

```
num1 = 0
num2 = 1
print(num1)
print(num2)
for i = 1 to 5
    newNum = num1 + num2
    print(newNum)
    num1 = num2
    num2 = newNum
endfor
```

i	newNum	num1	Num2	output
		0	1	0 1
1	1	1	2	1
2	3	2	4	3
3	6	4	8	6
4	12	8	16	12
5	24	16	32	24



# Errors

- The original Fibonacci program worked, but gave a result that wasn't intended by the programmer
  - This is an example of a **logical error**
- The below program has three **syntax errors** – what are they?

```
name = input("Type in your name")
if name = "George"
    print("hello" name)
else
    print("Your name isn't George")
endif
```

# Errors

## Answers

- Corrected code:

```
name = input("Type in your name")
```

Missing quote mark to end string

```
if name == "George"
```

== for equality (= means assignment)

```
    print("hello" + name)
```

+ needed to concatenate two strings

```
else
```

```
    print("Your name isn't George")
```

```
endif
```





# Plenary

- In pairs answer the following:
  - How is a trace table used to help find errors in a program?
  - Two types of error are syntax errors and logical errors. Explain what both of these mean with an example



# Plenary

Answers

- How is a trace table used to help find errors in a program?
  - Variable names and outputs are put in columns
  - The programmer traces through the program line by line updating the values of variables and outputs
  - A row is used for each iteration
- Two types of error are syntax errors and logical errors
  - Syntax errors - doesn't follow the rules of the language  
e.g. `print("hello)` - this has no final `"` for a string
  - Logical errors – the logic of the program is incorrect  
e.g. `average = ((num1 + num2) / 3)`



## 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.