Topic 2.1.1 – Computational thinking

Activity 1: Key terms

Using the options provided, match each description to the correct computational thinking method.

Difficulty level:

A	Decomposition
B	Abstraction
C	Algorithmic thinking

The removal of unnecessary elements so that the important points remain.	The process of creating a list of instructions are that are designed to solve a problem.	Breaking down a (large) complex problem into smaller and more manageable sub-tasks.

Topic 2.1.1 – Computational thinking (Answers)

Activity 1: Key terms

Difficulty level:

Using the options provided, match each description to the correct computational thinking method.

A	Decomposition
B	Abstraction
C	Algorithmic thinking

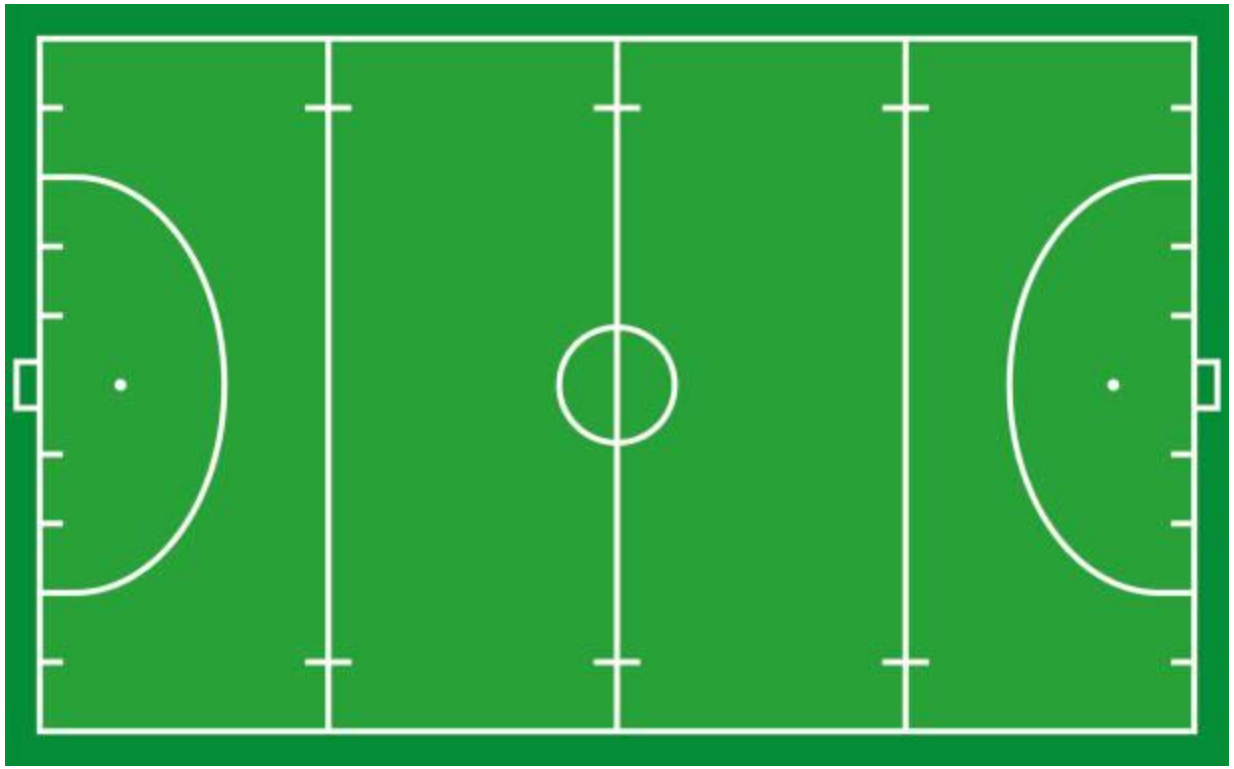
B	C	A
The removal of unnecessary elements so that the important points remain.	The process of creating a list of instructions are that are designed to solve a problem.	Breaking down a (large) complex problem into smaller and more manageable sub-tasks.

Topic 2.1.1 – Computational thinking

Activity 2: Abstraction

Difficulty level:

Gareth is the director of Landale Ice Hockey team and wants to create a program to store the results of each game they play and the name of the goal scorers.



What would be removed from this program?

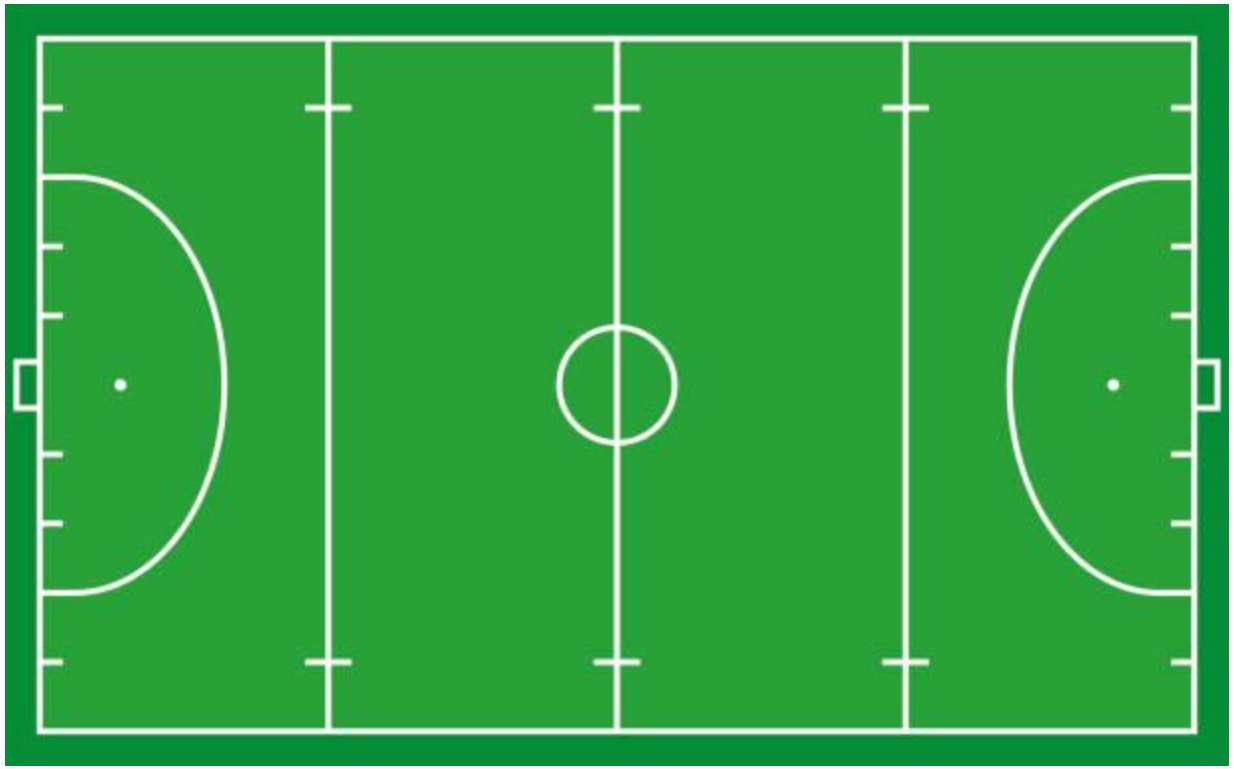
What important parts would remain in this program?

Topic 2.1.1 – Computational thinking (Answers)

Activity 2: Abstraction

Difficulty level:

Gareth is the director of Landale Ice Hockey team and wants to create a program to store the results of each game they play and the name of the goal scorers.



What would be removed from this program?

Minute of the goal, Attendance, Referee, Any fouls/penalties
Any other reasonable example.

What important parts would remain in this program?

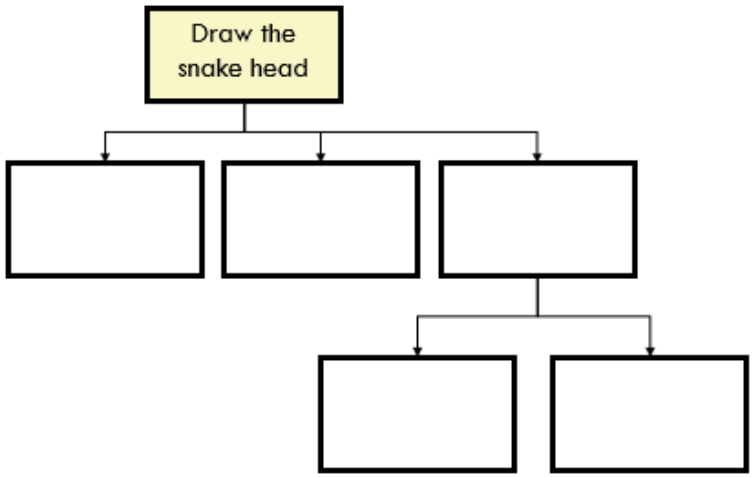
Player name
Match results
Goals scored

Topic 2.1.1 – Computational thinking

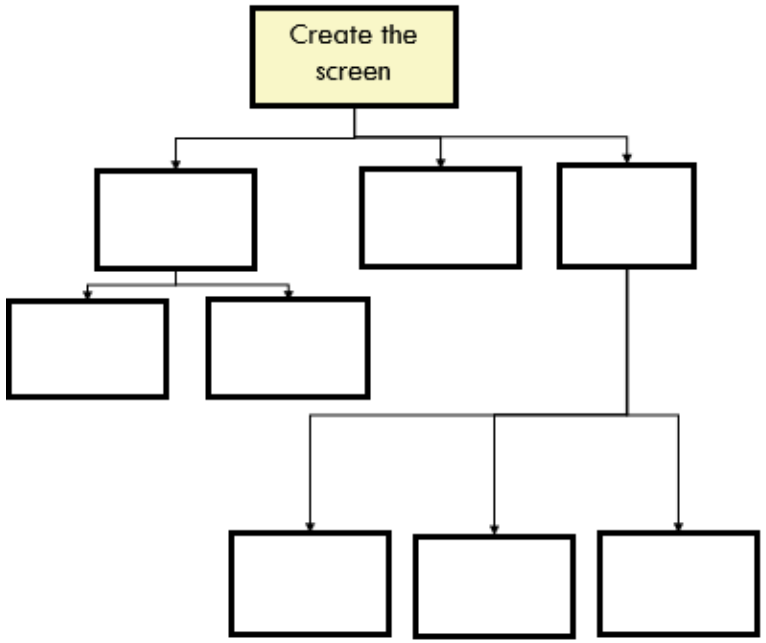
Activity 3: Structure diagram:

Difficulty level:

Structure diagrams are commonly used in problem decomposition. Complete the structure diagrams below for the game Snake using the options provided.



A	Create the eyes
B	Define the shape of the head
C	Identify the colour of the head.
D	Set eye colour
E	Set eye shape.

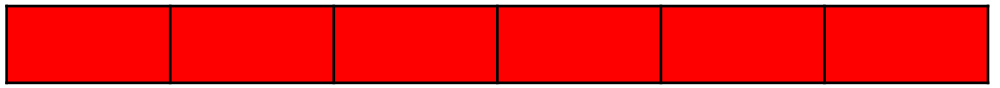


A	Add Background colour
B	Add text
C	Set dimensions
D	Set font colour
E	Set font size
F	Set font style
G	Set the length
H	Set the width

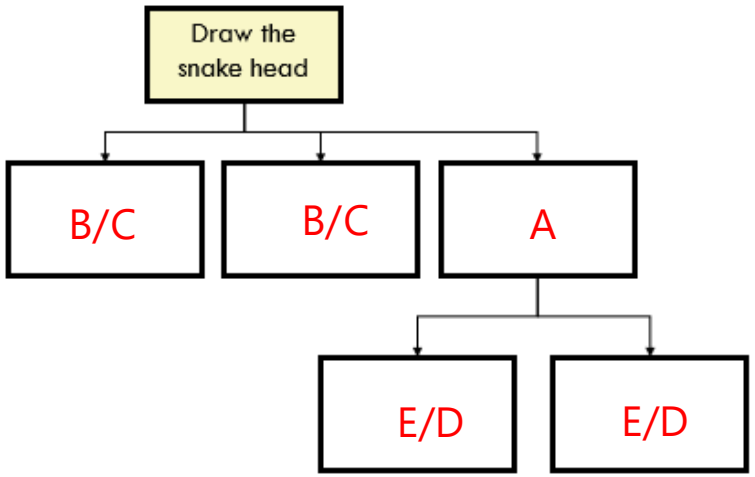
Topic 2.1.1 – Computational thinking (Answers)

Activity 3: Structure diagram:

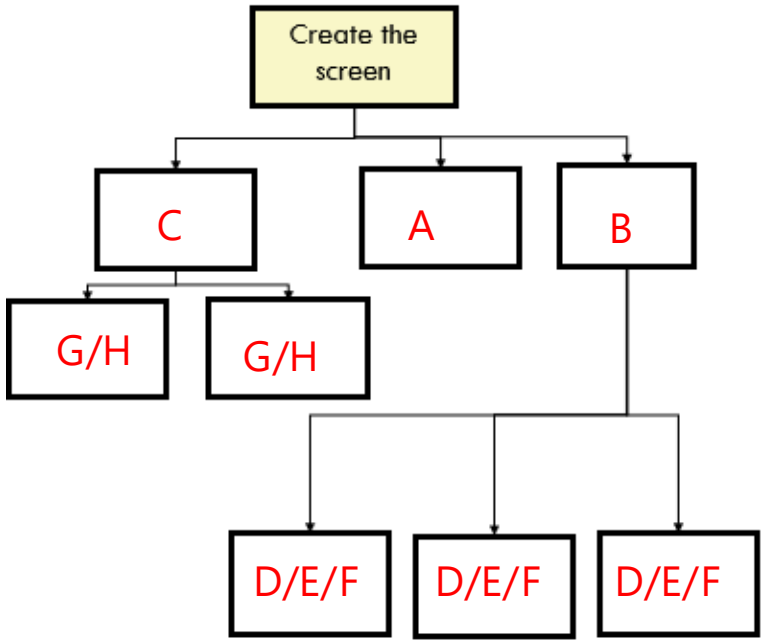
Difficulty level:



Structure diagrams are commonly used in problem decomposition. Complete the structure diagrams below for the game Snake using the options provided.



A	Create the eyes
B	Define the shape of the head
C	Identify the colour of the head.
D	Set eye colour
E	Set eye shape.



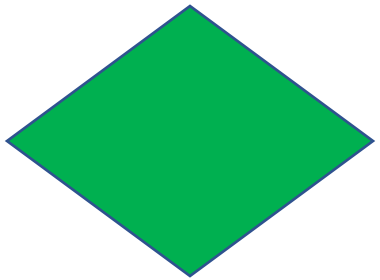
A	Add Background colour
B	Add text
C	Set dimensions
D	Set font colour
E	Set font size
F	Set font style
G	Set the length
H	Set the width

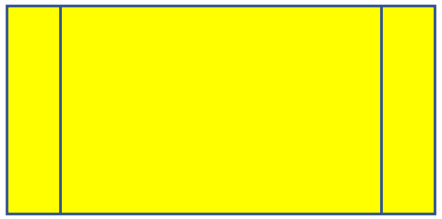
Topic 2.1.2 – Designing, creating and refining algorithms

Activity 1: Flow chart symbols

Difficulty level:

Name each flow chart symbol using the options below.





A	Start/Stop
B	Input/Output
C	Process
D	Decision
E	Subroutine
F	Line

Topic 2.1.2 – Designing, creating and refining algorithms (Answers)

Activity 1: Flow chart symbols

Difficulty level:

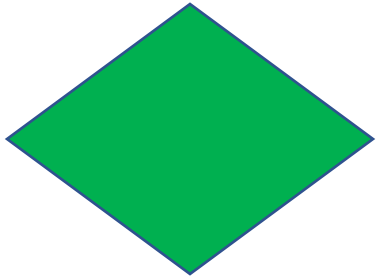
Name each flow chart symbol using the options below.

F

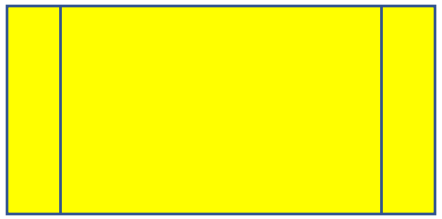
A

B

C



D



E

A	Start/Stop
B	Input/Output
C	Process
D	Decision
E	Subroutine
F	Line

Topic 2.1.2 – Designing, creating and refining algorithms

Activity 2: Correcting algorithms

Difficulty level:

Look at the code below and identify all the errors you can see and suggest a fix for each one.

```
import random
num = random.randint(1,20)
for x in range(10):
    print("num")
    if num < 10:
        print("Greater than 10")
    else:
        prnit("Less than 10")
```

Line	Syntax or Logic error?	Error Description	Solution

A Syntax error is….

A logic error is….

Topic 2.1.2 – Designing, creating and refining algorithms (Answers)

Activity 2: Correcting algorithms

Difficulty level:

Look at the code below and identify all the errors you can see and suggest a fix for each one.

```
import random
num = random.randint(1,20)
for x in range(10):
    print("num")
    if num < 10:
        print("Greater than 10")
    else:
        prnit("Less than 10")
```

Line	Syntax or Logic error?	Error Description	Solution
2	Logic	num will generate the same number each time.	Place num inside the loop.
4	Logic	The string num will be printed, not the value.	print(num)
5	Logic	Incorrect operator. Less than has been used.	if num > 10
8	Syntax	The function used is not recognised.	print("Less than 10")

A Syntax error is….

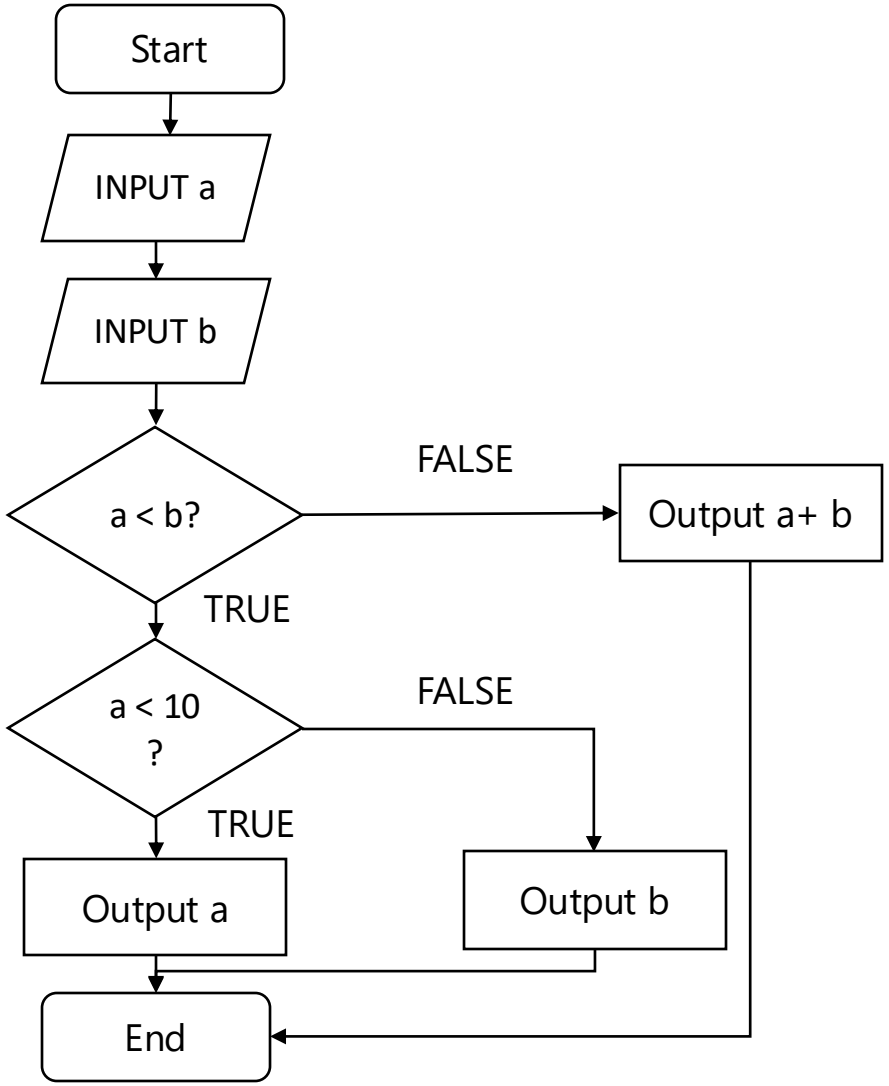
When the code used does not meet the rules of the language.

A logic error is….

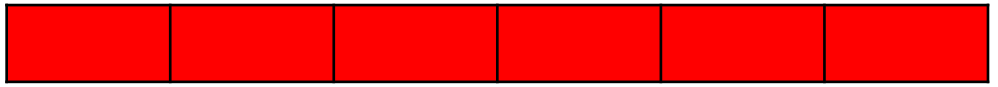
When the program works but not as intended.

Topic 2.1.2 – Designing, creating and refining algorithms

Activity 3: Interpreting an algorithm,



Difficulty level:

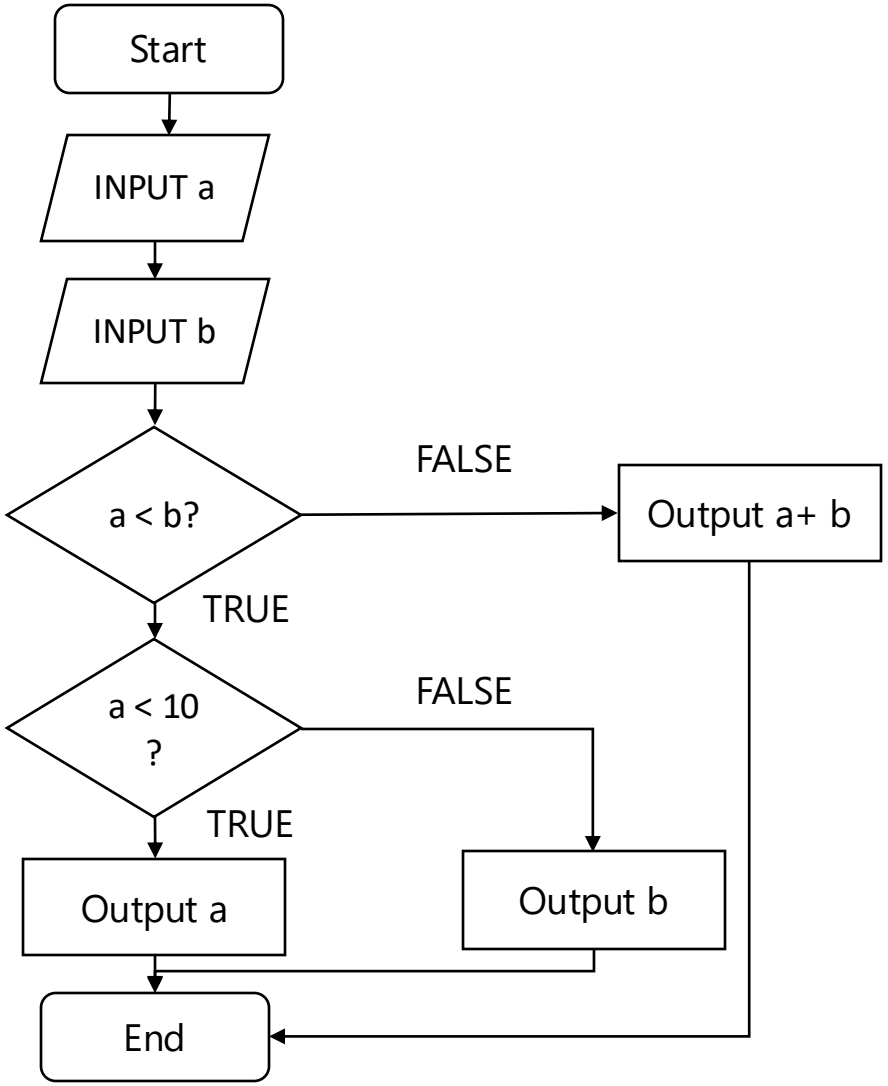


Complete the table to give the output when each of the following set of values are input into the algorithm as a and b.

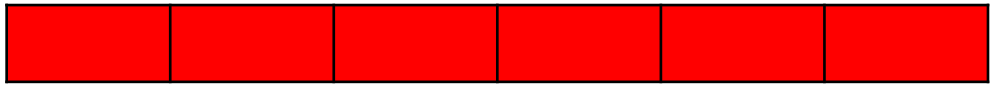
INPUT a	INPUT b	Output
14	11	
7	8	
1	2	
10	3	
7	7	

Topic 2.1.2 – Designing, creating and refining algorithms

Activity 3: Interpreting an algorithm,



Difficulty level:



Complete the table to give the output when each of the following set of values are input into the algorithm as a and b.

INPUT a	INPUT b	Output
14	11	25
7	8	7
1	2	1
10	3	13
7	7	14

Do Now – Searching & Sorting algorithms - 20/04/2022

Activity 1: Match them up

Difficulty level:

Match up each algorithm with the description below.

1. Binary search	**A.** Goes through a list of data a number of times, comparing two items that are side by side to see which is out of order.
2. Linear search	**B.** Repeatedly dividing in half the portion of the list that could contain the item, until you've narrowed down the possible locations to just one.
3. Bubble sort	**C.** Repeatedly divided into two until all the elements are separated individually. Pairs of elements are then compared, placed into order and combined.
4. Insertion sort	**D.** Searching the element sequentially until the element is found in the list.
5. Merge sort	**E.** Compares values in turn, starting with the second value in the list. If this value is greater than the value to the left of it.

Topic 2.1.3 – Searching & Sorting algorithms (Answers)

Activity 1: Match them up

Difficulty level:

Match up each algorithm with the description below.

1. Binary search	A. Goes through a list of data a number of times, comparing two items that are side by side to see which is out of order.
2. Linear search	B. Repeatedly dividing in half the portion of the list that could contain the item, until you've narrowed down the possible locations to just one.
3. Bubble sort	C. Repeatedly divided into two until all the elements are separated individually. Pairs of elements are then compared, placed into order and combined.
4. Insertion sort	D. Searching the element sequentially until the element is found in the list.
5. Merge sort	E. Compares values in turn, starting with the second value in the list. If this value is greater than the value to the left of it.

Searching & sorting algorithms

Activity 2: Searching algorithms

Use the data set provided below to perform a linear and binary search to find the number 14.

Difficulty level:

Linear search					
4	7	8	11	13	14

Binary search					
4	7	8	11	13	14

State **one** pre-requisite of a searching algorithm.	
Definition: A thing that is required as a prior condition for something else to happen or exist.	

Which algorithm is likely to be more efficient with larger sets of data?	
Which algorithm is likely to be more efficient with smaller sets of data?	

Topic 2.1.3 – Searching & sorting algorithms (Answers)

Activity 2: Searching algorithms

Use the data set provided below to perform a linear and binary search to find the number 14.

Difficulty level:

Linear search

4	7	8	11	13	14
4	**7**	8	11	13	14
4	7	**8**	11	13	14
4	7	8	**11**	13	14
4	7	8	11	**13**	14
4	7	8	11	13	**14**

Binary search

4	7	**8**	11	13	14
		11	**13**	14	
			14		

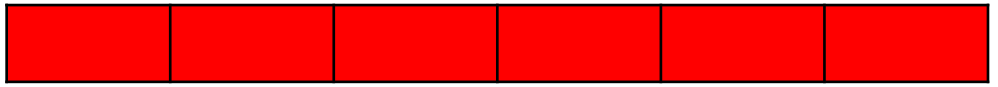
State **one** pre-requisite of a searching algorithm.	
Definition: A thing that is required as a prior condition for something else to happen or exist.	The list of items are in order.

Which algorithm is likely to be more efficient with larger sets of data?	Binary search
Which algorithm is likely to be more efficient with smaller sets of data?	Linear search

Topic 2.1.3 – Searching & sorting algorithms

Activity 3: Sorting algorithms

Difficulty level:



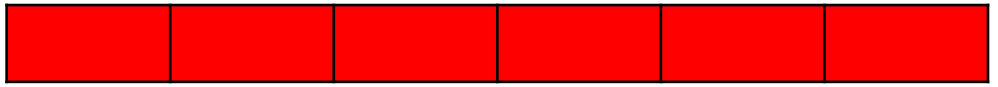
Show the stages of each sorting algorithm using the data sets provided below.

Bubble sort	Insertion sort	Merge sort
5 2 3 12 8 6	4 8 2 7 5 11	1 10 8 6 11 9

Topic 2.1.3 – Searching & sorting algorithms (Answers)

Activity 3: Sorting algorithms

Difficulty level:



Show the stages of each sorting algorithm using the data sets provided below.

Bubble sort

5	2	3	12	8	6
2	5	3	12	8	6
2	3	5	12	8	6
2	3	5	8	12	6
2	3	5	8	6	12
2	3	5	6	8	12

Insertion sort

4	8	2	7	5	11
4	8	2	7	5	11
4	8	2	7	5	11
2	4	8	7	5	11
2	4	7	8	5	11
2	4	5	7	8	11
2	4	5	7	8	11

Merge sort

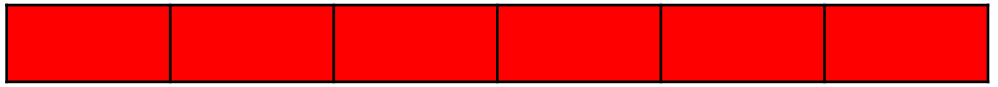
1	10	8	6	11	9
1	10	8	6	11	9
1	10	6	8	9	11
1	6	8	9	10	11

Topic 2.1.3 – Searching & sorting algorithms

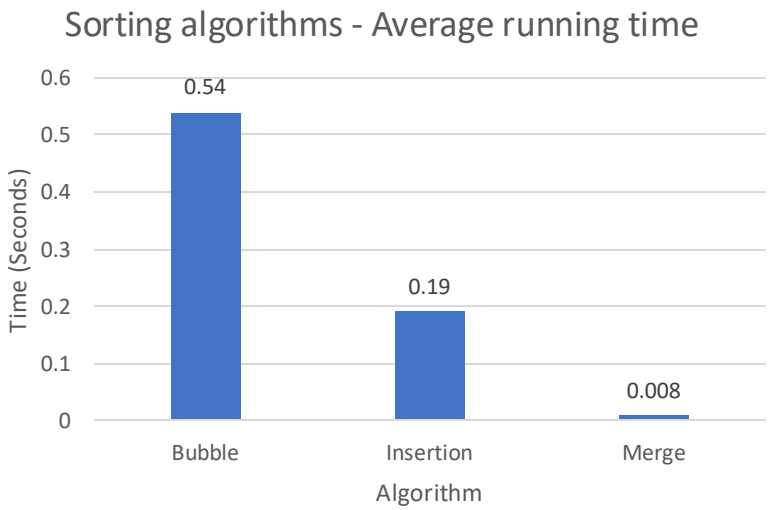
Activity 4: Comparing algorithms

Tick one or more boxes per row to identify the sorting algorithm that best fits each statement.

Difficulty level:

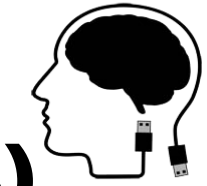


Algorithm	Test 1	Test 2	Test 3	Average
Bubble	0.44	0.69	0.50	0.54
Insertion	0.21	0.16	0.21	0.19
Merge	0.01	0.006	0.007	0.008



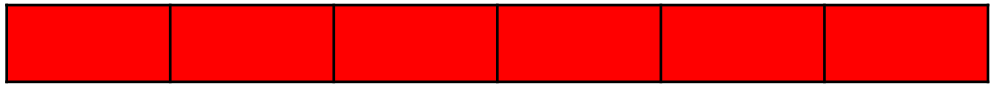
Statement	Bubble	Insertion	Merge
The easiest of the three to write and code.			
Less efficient for sorting of large data			
The fastest at sorting data.			
The most difficult to code out of the three.			
Uses more memory space to store the sub elements of the initial split list.			
The slowest because it may need to perform multiple iterations (i.e. go back to the beginning)			
Efficient for sorting of small data/data that is almost sorted			

Topic 2.1.3 – Searching & sorting algorithms (Answers)



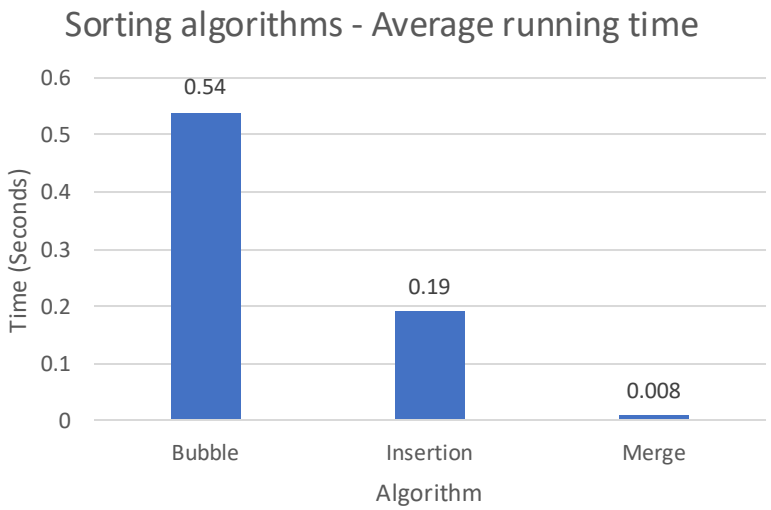
Activity 4: Comparing algorithms

Difficulty level:



Tick one or more boxes per row to identify the sorting algorithm that best fits each statement.

Algorithm	Test 1	Test 2	Test 3	Average
Bubble	0.44	0.69	0.50	0.54
Insertion	0.21	0.16	0.21	0.19
Merge	0.01	0.006	0.007	0.008



Statement	Bubble	Insertion	Merge
The easiest of the three to write and code.	✓		
Less efficient for sorting of large data	✓	✓	
The fastest at sorting data.			✓
The most difficult to code out of the three.			✓
Uses more memory space to store the sub elements of the initial split list.			✓
The slowest because it may need to perform multiple iterations (i.e. go back to the beginning)	✓		
Efficient for sorting of small data/data that is almost sorted		✓	

Topic 2.2.1 – Programming fundamentals

Activity 1: Arithmetic operators

Difficulty level:

Identify the symbols used to represent each operation shown in the table below. The options have been provided. Some can include more than one answer.

Operator	Symbol
Addition	
Subtraction	
Multiplication	
Integer division	
Float division	
Modulus	
Exponentiation	

+	**	//
*	%	-
MOD	DIV	/
		^

Topic 2.2.1 – Programming fundamentals (Answers)

Activity 1: Adapting code

Difficulty level:

Identify the symbols used to represent each operation shown in the table below. The options have been provided. Some can include more than one answer.

Operator	Symbol
Addition	+
Subtraction	-
Multiplication	*
Integer division	DIV or //
Float division	/
Modulus	% or MOD
Exponentiation	** or ^

+	**	//
*	%	-
MOD	DIV	/
		^

Topic 2.2.1 – Programming fundamentals

Activity 2: Variables and constants

Answer the questions shown below.

- a = 5
- b = int(input("Enter a number"))
- c = int(input("Enter another number"))
- total = (a+b) - c
- print (total)

Difficulty level:

Fill in the remaining gaps to describe what is meant by a variable and a constant.

A variable is a _____ storage location that stores a value that _____ change at any point.

A constant is a type of _____ that stores a value that _____ change later on.

Using the algorithm shown on the left. Identify **two** variables that have been used**. (2 marks)**

Guidance:

Use your description of variable above to help you.

Topic 2.2.1 – Programming fundamentals (Answers)

Activity 2: Variables and constants

Answer the questions shown below.

- a = 5
- b = int(input("Enter a number"))
- c = int(input("Enter another number"))
- total = (a+b) - c
- print (total)

Difficulty level:

Fill in the remaining gaps to describe what is meant by a variable and a constant.

A variable is a **named** storage location that stores a value that **can** change at any point.

A constant is a type of **variable** that stores a value that **cannot** change later on.

Using the algorithm shown on the left. Identify **two** variables that have been used**. (2 marks)**

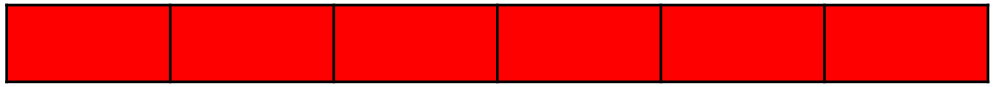
Guidance: Use your description of variable above to help you.	b c Total

Topic 2.2.1 – Programming fundamentals

Activity 3: Writing algorithms

Complete the two algorithms below.

Difficulty level:



Richard is writing a program to keep of athletes running times.

Write an algorithm that will calculate how many hours and minutes are contained in a time and entered as minutes. For example, 150 minutes is 2 hours and 30 minutes.

Guidance:

- Input a number (BP1)
- Calculate the number of hours (BP2)
- Calculate the number minutes (BP3)
- Output using an appropriate message (BP4)

Write an algorithm that will allow the user to input final velocity in m/s (v), initial velocity in m/s (u) and time in seconds (t).

A formula should then be used to calculate acceleration and then output into the main program.
The formula that calculates acceleration is: **a = (v-u) ÷ t.**

Guidance:

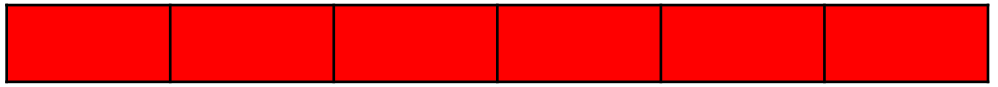
- Input three numbers (BP1)
- Calculate the acceleration using the formula (BP2)
- Output using an appropriate message (BP3)

Topic 2.2.1 – Programming fundamentals (Answers)

Activity 3: Writing algorithms

Complete the two algorithms below.

Difficulty level:



Richard is writing a program to keep of athletes running times.

Write an algorithm that will calculate how many hours and minutes are contained in a time and entered as minutes. For example, 150 minutes is 2 hours and 30 minutes.

Guidance:	
• Input a number (BP1) • Calculate the number of hours (BP2) • Calculate the number minutes (BP3) • Output using an appropriate message (BP4)	• Num = int(input("Enter a number") • Hours = Num // 60 • Minutes = Num % 60 • print(Hours,"hours and", Minutes, "minutes") Accept use of DIV and MOD.

Write an algorithm that will allow the user to input final velocity in m/s (v), initial velocity in m/s (u) and time in seconds (t).

A formula should then be used to calculate acceleration and then output into the main program.
The formula that calculates acceleration is: **a = (v-u) ÷ t.**

Guidance:	
• Input three numbers (BP1) • Calculate the acceleration using the formula (BP2) • Output using an appropriate message (BP3)	• v = int(input("Input final velocity")) • u = int(input("Input initial velocity")) • t = int(input("Input time in seconds")) • a = (v – u) //t • print ("The acceleration is", a)

Topic 2.2.1 – Selection

Activity 1: Comparison operators

Identify the symbols used to represent each operation shown in the table below. The options have been provided.

Difficulty level:

Operator	Symbol
Equal to	
Not equal to	
Greater than	
Less than	
Greater or equal to	
Less or equal to	

!= > <

== >= <=

Topic 2.2.1 – Selection (Answers)

Activity 1: Comparison operators

Identify the symbols used to represent each operation shown in the table below. The options have been provided.

Difficulty level:

Operator	Symbol
Equal to	==
Not equal to	!=
Greater than	>
Less than	<
Greater or equal to	>=
Less or equal to	<=

!= > <

== >= <=

Topic 2.2.1 – Selection

Activity 2: Completing and refining algorithms

Answer both questions below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:

Complete

Fill in the missing gaps to complete the algorithm below that checks if a number entered by the user is greater than 10.

Guidance:

- Input a number (BP1)
- Use a selection statement (BP2)
- Correctly compare the number for all three outcomes (BP3, BP4, BP5)

```
Num = int(input("Enter a number"))
..... Num ..... 10:
    print("Greater than 10")
.....
    print("Less than 10.")
```

Refine

The algorithm on the left will not work in the way it's intended.

Refine the algorithm so it works the way it's intended to.

Guidance:

- Input a number (BP1)
- Use a selection statement (BP2)
- Correctly compare the number for all three outcomes (BP3, BP4, BP5)

Topic 2.2.1 – Selection (Answers)

Activity 2: Completing and refining algorithms

Answer both questions below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:

Complete

Fill in the missing gaps to complete the algorithm below that checks if a number entered by the user is greater than 10.

Guidance:

- Input a number (BP1)
- Use a selection statement (BP2)
- Correctly compare the number for all three outcomes (BP3, BP4, BP5)

```
Num = int(input("Enter a number"))
if Num > 10:
    print("Greater than 10")
else:
    print("Less than 10.")
```

Refine

The algorithm on the left will not work in the way it's intended.

Refine the algorithm so it works the way it's intended to.

Guidance:

- Input a number (BP1)
- Use a selection statement (BP2)
- Correctly compare the number for all three outcomes (BP3, BP4, BP5)

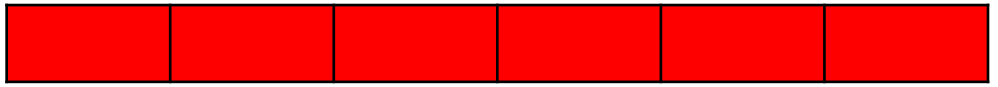
```
Num = int(input("Enter a number"))
if Num > 10:
    print("Greater than 10")
elif Num < 10:
    print("Less than 10")
else:
    print("Number is exactly 10")
```

Topic 2.2.1 – Selection

Activity 3: Writing algorithms

Answer both questions below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



Write an algorithm that will:

- allow the user to enter a number.
- check whether the number is less than 200.
- output a message if the number is equal to 150, 100, 50 or less than 50.
- output a message if the value is equal or greater than 200.

Guidance:

- Input a number (BP1)
- Use a selection to statement to check value is less than 200 (BP2)
- Use selection to check value is equal to 150, 100, 50 and less than 50 (BP3)
    - Output the appropriate message for each (BP4)
- Check value is equal or greater than 200 (BP5)

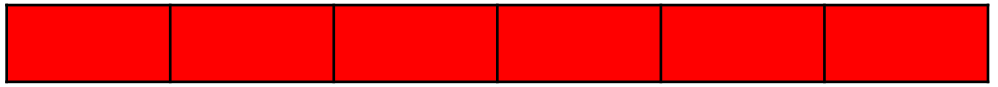
Answer:

Topic 2.2.1 – Selection (Answers)

Activity 3: Writing algorithms

Answer both questions below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



Write an algorithm that will:

- allow the user to enter a number.
- check whether the number is less than 200.
- output a message if the number is equal to 150, 100, 50 or less than 50.
- output a message if the value is equal or greater than 200.

Guidance:

- Input a number (BP1)
- Use a selection to statement to check value is less than 200 (BP2)
- Use selection to check value is equal to 150, 100, 50 and less than 50 (BP3)
  - Output the appropriate message for each (BP4)
- Check value is equal or greater than 200 (BP5)

Answer:

```
Num = int(input("Enter a number"))
if Num < 200:
    print ("Expression value is less than 200")
    if Num == 150:
        print ("Which is 150")
    elif Num == 100:
        print ("Which is 100")
    elif Num == 50:
        print ("Which is 50")
    elif Num < 50:
        print ("Expression value is less than 50")
else:
    print ("Could not find true expression")
```

Topic 2.2.1 – Counter-controlled iteration

Activity 1: Adapting code

Difficulty level:

Complete the table below. The first one has been done for you.

Q	Code	Description
1	• for x in range (1,5): • print (x)	Print the value of x from 1 to 4.
2	• for x in range (_____): • print (x)	Print the value of x from 3 to 12.
3	• for x in range (1,14, ____): • print (x)	Print the values in twos from 1 to 13.
4		Print the values of x from 10 to 1.
5		Print the values of x from 1 to whichever number is inputted by the user.

Topic 2.2.1 – Counter-controlled iteration (Answers)

Activity 1: Adapting code

Difficulty level:

Complete the table below. The first one has been done for you.

Q	Code	Description
1	• for x in range (1,5): • print (x)	Print the value of x from 1 to 4.
2	• for x in range (**3, 13**): • print (x)	Print the value of x from 3 to 12.
3	• for x in range (1,14, **2**): • print (x)	Print the values in twos from 1 to 13.
4	• **for in range (10, 0, -1):** • **print (x)**	Print the values of x from 10 to 1.

5	• **num=int(input("Enter a number: "))**
	  • **for x in range (0, num):**
	  • **print (x)**
		Print the values of x from 1 to whichever number is inputted by the user.

Topic 2.2.1 – Counter-controlled iteration

Activity 2: Tracing algorithms

Difficulty level:

Complete a trace table for each of the algorithms shown below.

Algorithm 1:

```
y = 2
for x in range (1,7):
    y =y + x
print (y)
```

x	y	Output

Algorithm 2:

```
a = 5
b = 1
for x in range (1,7):
    a = a – 1
    b = (a+b)*2
print (b)
```

a	b	Output

Topic 2.2.1 – Counter-controlled iteration (Answers)

Activity 2: Tracing algorithms

Difficulty level:

Complete a trace table for each of the algorithms shown below.

Algorithm 1:

```
y = 2
for x in range (1,7):
    y =y + x
print (y)
```

x	y	Output
1	3	
2	5	
3	8	
4	12	
5	17	
6	23	23

Algorithm 2:

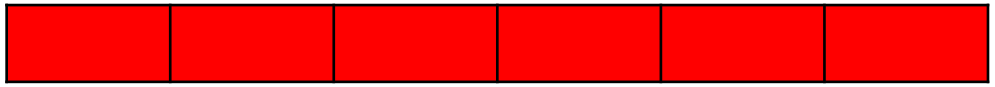
```
a = 5
b = 1
for x in range (1,6):
    a = a – 1
    b = (a+b) - 2
print (b)
```

a	b	Output
4	3	
3	4	
2	4	
1	3	
0	1	1

Topic 2.2.1 – Counter-controlled iteration

Activity 3: Creating and Refining algorithms.
Answer both questions below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



Refine

Look at the program written below:

```
print("1")
print("2")
print("3")
print("4")
print("5")
print("Time is up")
```

Refine the program to be more efficient. Write the refined version of the algorithm.

Guidance:

- Use a FOR loop (BP1)
- Correct range output 1 to 5 (BP2)
- Output "Time is up" (BP3)

Create

Write an algorithm that asks the user to input a number between 1 and 12.

The program outputs the times table of that number between 1 and 12 in the following format.

5 x 5 = 25

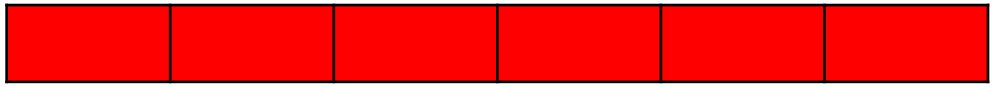
Guidance:

- Input a number (BP1)
- Use a FOR loop (BP2)
- Set correct range (BP3)
- Output answer (BP4)
- Use an appropriate message (BP5)

Topic 2.2.1 – Counter-controlled iteration (Answers)

Activity 3: Creating and Refining algorithms.
Answer both questions below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



Refine

Look at the program written below:

```
print("1")
print("2")
print("3")
print("4")
print("5")
print("Time is up")
```

Refine the program to be more efficient. Write the refined version of the algorithm.

Guidance:

- Use a FOR loop (BP1)
- Correct range output 1 to 5 (BP2)
- Output "Time is up" (BP3)

- For x in range (5):
  - print (x)
- print ("Time is up")

Create

Write an algorithm that asks the user to input a number between 1 and 12.

The program outputs the times table of that number between 1 and 12 in the following format.

5 x 5 = 25

Guidance:

- Input a number (BP1)
- Use a FOR loop (BP2)
- Set correct range (BP3)
- Output answer (BP4)
- Use an appropriate message (BP5)

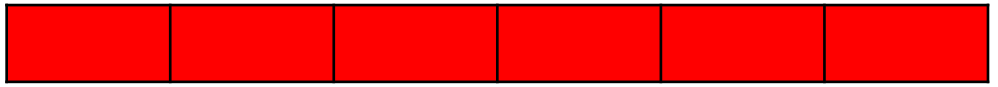
- Num1 = int(input("Enter a number")
- For x in range (1,13):
  - print (x,"*",x,"="x*Num1)

Topic 2.2.1 – Counter-controlled iteration

Activity 4: Interpreting a trace table.

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



i	j	Total
1	1	1
1	2	3
1	3	6
2	1	8
2	2	12
2	3	18
3	1	21
3	2	27
3	3	36

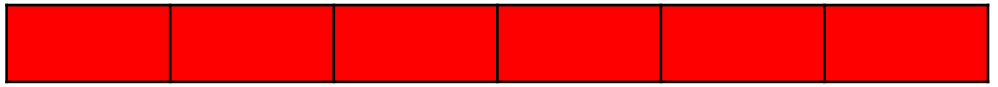
Answer:

Topic 2.2.1 – Counter-controlled iteration (Answer)

Activity 4: Interpreting a trace table.

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



i	j	Total
1	1	1
1	2	3
1	3	6
2	1	8
2	2	12
2	3	18
3	1	21
3	2	27
3	3	36

Answer:

```
Num = 0
Total = 0
for i in range (1,4):
    for j in range (1,4):
        Num = i * j
        Total += Num
        print(Total)
```

Topic 2.2.1 – Condition-controlled iteration

Activity 1: Programming constructs

Difficulty level:

Complete the questions below.

Match up each programming construct to it's correct description.

1. Sequence	A. A process of repeating steps.
2. Selection	B. A list of instructions that follow a logical order.
3. Iteration	C. An outcome that depends on whether a certain condition is met.

Tick one box per row to identify which programming constructs have been used in the algorithms below.

```
y = 2
for x in range (1,7):
    y =y + x
print (y)
```

Sequence	
Selection	
Iteration	

```
Num = int(input("Enter a number")
if Num > 10:
    print("Greater than 10")
else:
    print("Less than 10.")
```

Sequence	
Selection	
Iteration	

Topic 2.2.1 – Condition-controlled iteration (Answers)

Activity 1: Programming constructs

Difficulty level:

Complete the questions below.

Match up each programming construct to it's correct description.

1. Sequence	A. A process of repeating steps.
2. Selection	B. A list of instructions that follow a logical order.
3. Iteration	C. An outcome that depends on whether a certain condition is met.

Tick one box per row to identify which programming constructs have been used in the algorithms below.

```
y = 2
for x in range (1,7):
    y =y + x
print (y)
```

Sequence	✓
Selection	
Iteration	✓

```
Num = int(input("Enter a number")
if Num > 10:
    print("Greater than 10")
else:
    print("Less than 10.")
```

Sequence	✓
Selection	✓
Iteration	

Topic 2.2.1 – Condition-controlled iteration

Activity 2: Refining algorithms

Difficulty level:

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

A teacher has developed a program that will allow her students three attempts to login to their account. After three unsuccessful attempts, they will be locked out for two hours.

- name = input (“Enter name”)
- attempts = 0
- password = “hello”
- if password != name
  - attempts = attempts + 1
  - name = input (“Enter name”)
- if password != name
  - attempts = attempts + 1
  - name = input (“Enter name”)
- if password != name
  - attempts = attempts + 1
- print(“locked out of account)

Refine the program to be more efficient. Write the refined version of the algorithm. **(6 marks)**

Guidance:

- Use of a loop (BP1)
- …checking whether password matches the input (BP2)
- …increase attempts counter (BP3)
- Check on how many attempts the user has made. (BP4)
- …appropriate output for attempts that are less or equal to 3. (BP5)

Topic 2.2.1 – Condition-controlled iteration (Answer)

Activity 2: Refining algorithms

Difficulty level:

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

A teacher has developed a program that will allow her students three attempts to login to their account. After three unsuccessful attempts, they will be locked out for two hours.

- name = input (“Enter name”)
- attempts = 0
- password = “hello”
- if password != name
  - attempts = attempts + 1
  - name = input (“Enter name”)
- if password != name
  - attempts = attempts + 1
  - name = input (“Enter name”)
- if password != name
  - attempts = attempts + 1
- print(“locked out of account)

Refine the program to be more efficient. Write the refined version of the algorithm. **(6 marks)**

Guidance:

- Use of a loop (BP1)
- …checking whether password matches the input (BP2)
- …increase attempts counter (BP3)
- Check on how many attempts the user has made. (BP4)
- …appropriate output for attempts that are less or equal to 3. (BP5)

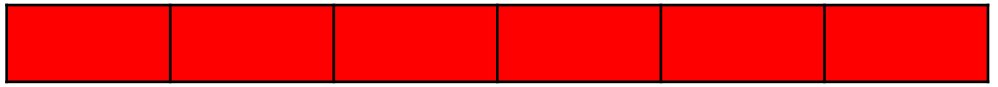
```
attempts = 1
password = "Hello"
entry = input("Enter password")
while entry != password:
    if attempts < 3:
        attempts +=1
        entry = input("Enter password")
    else:
        print("Locked out of account")
```

Topic 2.2.1 – Condition-controlled iteration

Activity 3: Writing an algorithm

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



Write an algorithm that will:

- allow the user to input a number
- add all the numbers entered by the user together
- Repeat bullet points 1 and 2 until the user enters the number 0.

Guidance:

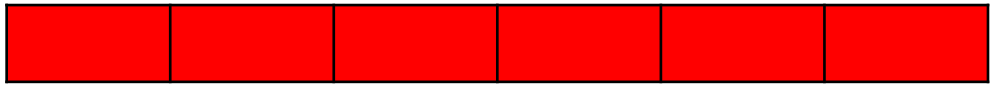
- Set up a variable to store the total (BP1)
- User to input a number (BP2)
- Use a loop to check number entered is not equal to 0 (BP3)
    - Increment entered values (BP4)
- Output the total (BP5)

Topic 2.2.1 – Condition-controlled iteration (Answer)

Activity 3: Writing an algorithm

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



Write an algorithm that will:

- allow the user to input a number
- add all the numbers entered by the user together
- Repeat bullet points 1 and 2 until the user enters the number 0.

Guidance:

- Set up a variable to store the total (BP1)
- User to input a number (BP2)
- Use a loop to check number entered is not equal to 0 (BP3)
  - Increment entered values (BP4)
- Output the total (BP5)

```
Total = 0
Num = int(input("Enter a number") )
while Num != 0:
    Num = int(input("Enter a number") )
    Total += Num
print(Total)
```

Topic 2.2.2 – Data types

Activity 1: Data types

Difficulty level:

Match up each data type with the correct description.

1. Boolean	
2. Character	
3. Float	
4. Integer	
5. Real	
6. String	

A. Used to represent a whole number.
B. Used to represent a number that includes a decimal value.
C. Used to represent one of two outcomes.
D. Used to represent a single character within a string.
E. Used to represent a set of characters enclosed in quotation marks.

Topic 2.2.2 – Data types (Answers)

Activity 1: Data types

Difficulty level:

Match up each data type with the correct description.

1. Boolean	A. Used to represent a whole number.
2. Character	B. Used to represent a number that includes a decimal value.
3. Float	C. Used to represent one of two outcomes.
4. Integer	D. Used to represent a single character within a string.
5. Real	E. Used to represent a set of characters enclosed in quotation marks.
6. String	

Topic 2.2.2 – Data types

Activity 2: Use of data types in existing systems.

Use the image of the sat nav screen to label and describe the data types present.

Difficulty level:

Data type:

Justification:

Data type:

Justification:

Data type:

Justification:

Data type:

Justification:

Topic 2.2.2 – Data types (Answers)

Activity 2: Use of data types in existing systems.

Difficulty level:

Use the image of the sat nav screen to label and describe the data types present.

Data type:
Float/Real

Justification:
Specific distance in miles.

Data type:
String

Justification:
mph for miles per hour.

Data type:
Character

Justification:
The letter P for parking.

Data type:
Integer

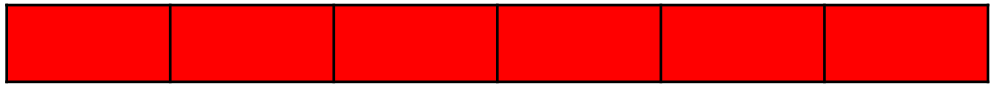
Justification:
The speed represented as a whole number.

Topic 2.2.2 – Data types

Activity 3: Writing an algorithm

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



Code:

```
Num1 = input("Enter a number")
Num2 = input("Enter another number")
print(Num1+Num2)
```

Output:

```
Enter a number: 4
Enter another number: 4
44
```

The program on the left contains a logic error. Casting is required for the two inputs.

Define the term 'casting' and identify how it can be used to remove the logic error in this program. (2 marks)

Guidance:

- Definition of casting (BP1)
- Removing the logic error with casting by rewriting the lines in question. (BP2)

Topic 2.2.2 – Data types (Answers)

Activity 3: Writing an algorithm

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

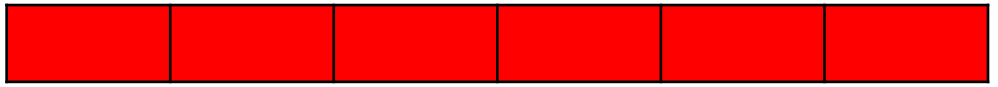
Code:

```
Num1 = input("Enter a number")
Num2 = input("Enter another number")
print(Num1+Num2)
```

Output:

```
Enter a number: 4
Enter another number: 4
44
```

Difficulty level:



The program on the left contains a logic error. Casting is required for the two inputs.

Define the term 'casting' and identify how it can be used to remove the logic error in this program. (2 marks)

Guidance:

- Definition of casting (BP1)
- Removing the logic error with casting by rewriting the lines in question. (BP2)

Casting is when you convert data from one type to another.

In this program the two lines need to be casted to integers:

```
Num1 = int(input("Enter a number"))
Num2 = int(input("Enter another number))
```

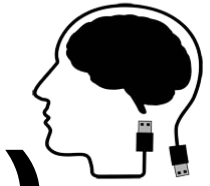
Topic 2.2.3 – String manipulation

Activity 1: Basic string manipulation

Difficulty level:

Complete the table below to identify the output from the algorithm provided.

Code	Output	Explanation
Fname = “Joe” Sname = “Bloggs” print(Fname[:1]) print(Sname[:2])		
Fname = "Joe" Sname = "Bloggs" print(Fname.upper()) print(Sname[-2:])		
Fname = "Joe" Sname = "Bloggs" print(len(Fname))+(len(Sname))		



Topic 2.2.3 – String manipulation (Answers)

Activity 1: Basic string manipulation

Difficulty level:

Complete the table below to identify the output from the algorithm provided.

Code	Output	Explanation
Fname = “Joe” Sname = “Bloggs” print(Fname[:1]) print(Sname[:2])	**J** **Bl**	**Extracts characters from the left of the string.**
Fname = "Joe" Sname = "Bloggs" print(Fname.upper()) print(Sname[-2:])	**OE** **gs**	**The upper function converts a string to uppercase. The surname has used a technique that extracts characters from the right of the string.**
Fname = "Joe" Sname = "Bloggs" print(len(Fname))+(len(Sname))	**9**	**Len checks the length of a string. Joe contains 3 and Bloggs contains 6 which makes 9.**

Topic 2.2.3 – String manipulation

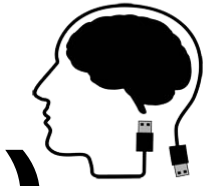
Activity 2: Concatenation and Slicing

Difficulty level:

Using the two snippets below, identify where concatenation and slicing has been used with an explanation to what they mean.

```
person1_fname = "David"
person1_sname = "Bowen"
person2_fname = "Daisy"
person2_sname = "Duffy"

print(person1_fname[:2]+person1_sname[2:5])
print(person2_fname[-2:]+person2_sname[0:3])
```

Slicing	Concatenation
Definition/Meaning:	Definition/Meaning:
Example from the code:	Example from the code:



Topic 2.2.3 – String manipulation (Answers)

Activity 2: Concatenation and Slicing

Difficulty level:

Using the two snippets below, identify where concatenation and slicing has been used with an explanation to what they mean.

```
person1_fname = "David"
person1_sname = "Bowen"
person2_fname = "Daisy"
person2_sname = "Duffy"

print(person1_fname[:2]+person1_sname[2:5])
print(person2_fname[-2:]+person2_sname[0:3])
```

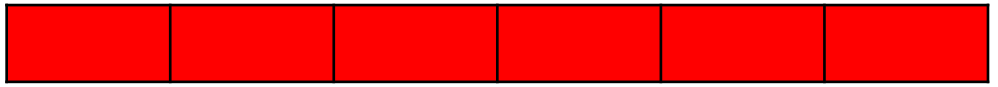
Slicing	Concatenation
Definition/Meaning: **It allows you to select specific parts of a string.**	Definition/Meaning: **The joining of two strings.**
Example from the code: **sname [2:5]**	Example from the code: **The use of the + operator**

Topic 2.2.3 – String manipulation

Activity 3: Writing an algorithm

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



Write an algorithm that will generate a school user name based on four inputs: First Name, Middle Name, Surname and Year started at the high school.

For example, Bob Eric Smith started in 2015 and his username is 15smithBE **(6 marks)**

Guidance:

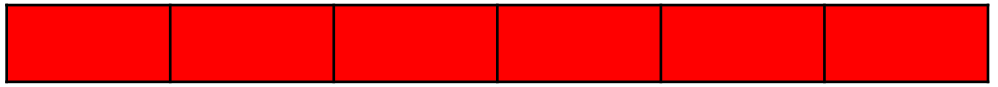
- User inputs (BP1)
- Correct format of first name (BP2)
- Correct format of middle name (BP3)
- Correct format of surname (BP4)
- Correct format of year (BP5)
- Output using an appropriate message (BP6)

Topic 2.2.3 – String manipulation (Answer)

Activity 3: Writing an algorithm

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



Write an algorithm that will generate a school user name based on four inputs: First Name, Middle Name, Surname and Year started at the high school.

For example, Bob Eric Smith started in 2015 and his username is 15smithBE **(6 marks)**

Guidance:

- User inputs (BP1)
- Correct format of first name (BP2)
- Correct format of middle name (BP3)
- Correct format of surname (BP4)
- Correct format of year (BP5)
- Output using an appropriate message (BP6)

```
Fn = input("Enter first name")
Mn = input("Enter middle name")
Sn = input("Enter surname")
Year = input("What year did you start at the high school?")
print(Year[-2:]+Sn.lower()+Fn[:1].upper()+Mn[:1].upper())
```

Topic 2.2.3 – File handling

Activity 1: File handling commands

Difficulty level:

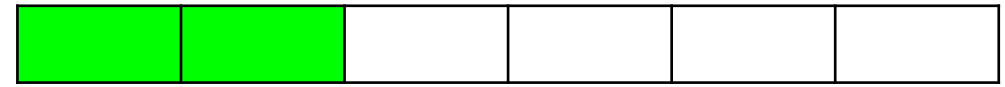
Identify the purpose of each file handling command shown below.

r	w	a	close	open

Topic 2.2.3 – File handling (Answers)

Activity 1: File handling commands

Difficulty level:



Identify the purpose of each file handling command shown below.

r	**Read contents of a text file**
w	**Write contents into a text file**
a	**Update contents of a text file.**
close	**Close the text file so it can't be read or written to.**
open	**Open the contents of a text file.**

Topic 2.2.3 – File handling

Activity 2: Manipulating text files.

Difficulty level:

Fill in the remaining gaps to complete each file handling algorithm.

Read contents of the text file

- F = open (“Trains.txt,”……”)
- print(F……..())
- F……..()

Read a specific line in a text file.

- F = open (“Trains.txt,”……”)
- print(F……..())[….]
- F……..()

Write the contents of a list to a text file.

- F= open("Trains2.txt",“…….")
- stations = ["Selly Oak", "Solihull", "Tyseley", "Yardley Green"]
- for i in ……..:
- F…….(stations[…….]+ " ")
- F……..()

Read a line in a text file

- F = open (“Trains.txt,”……”)
- print(F……..())
- F……..()

Write a line to a text file.

- F = open (“Trains.txt,”……”)
- print(F……..(“Lye”
- F……..()

Topic 2.2.3 – File handling (Answers)

Activity 2: Manipulating text files.

Difficulty level:

Fill in the remaining gaps to complete each file handling algorithm.

Read contents of the text file

- F = open (“Trains.txt,”**r**”)
- print(F.**read**())
- F.**close**()

Read a line in a text file

- F = open (“Trains.txt,”**r**”)
- print(F.**readline**())
- F.**close**()

Read a specific line in a text file.

- F = open (“Trains.txt,”**r**”)
- print(F.**readlines**.())[**0**]
- F……..()

Write a line to a text file.

- F = open (“Trains.txt,”**w**”)
- print(F.**write**(“Lye”
- F.**close**()

Write the contents of a list to a text file.

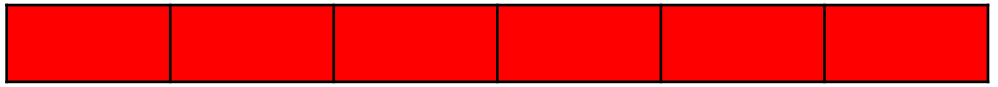
- F= open("Trains2.txt",“**w**")
- stations = ["Selly Oak", "Solihull", "Tyseley", "Yardley Green"]
- for i in **stations**:
- F.**write**(stations[**i**]+ " ")
- F.**close**()

Topic 2.2.3 – File handling

Activity 3: Writing an algorithm

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



Some new train stations have been added to a text file and a user wants to find out whether a particular station exists. **(5 marks)**

Write an algorithm that will:

- allow the user to enter the name of a train station.
- check if the station exists in the text file called Trains2.txt.
- repeat bullet point 2 until it reaches the end of the text file.

Guidance:

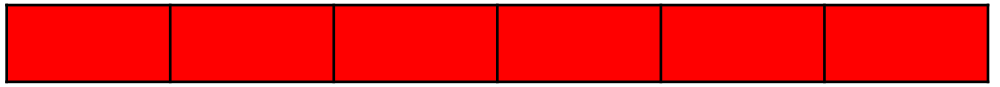
- User inputs (BP1)
- Open the correct text file (BP2)
- Use iteration to check the text file (BP3)
- Use selection to check if the station exists in the text file. (BP4)
- Output the appropriate message for both outcomes (BP5)

Topic 2.2.3 – File handling (Answers)

Activity 3: Writing an algorithm

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Difficulty level:



Some new train stations have been added to a text file and a user wants to find out whether a particular station exists. **(5 marks)**

Write an algorithm that will:

- allow the user to enter the name of a train station.
- check if the station exists in the text file called Trains2.txt.
- repeat bullet point 2 until it reaches the end of the text file.

Guidance:

- User inputs (BP1)
- Open the correct text file (BP2)
- Use iteration to check the text file (BP3)
- Use selection to check if the station exists in the text file. (BP4)
- Output the appropriate message for both outcomes (BP5)

```
T = input("Input station")
Line = open ("Trains2.txt","r")
for x in Line:
    if T in x:
        print("Station present")
    else:
        print("Station not present")
```

Topic 2.2.3 – SQL & Data Structures

Activity 1: Introduction to databases

Difficulty level:

Identify the two components of a database and the meaning of each SQL clause.

Product No.	Registration	Make	Year	Mileage	Price
0001	AV60 HES	Peugeot	2010	33156	£5,500
0002	GF56 RTE	Toyota	2006	26875	£8,500
0003	FD02 YOU	Hyundai	2002	85300	£3,499
0004	AD62 HGF	Peugeot	2012	50887	£7,649
0005	AF63 THE	Peugeot	2013	45860	£6,780
0006	GF64 NGB	Renault	2014	38665	£6,199
0007	GR11 JUL	Renault	2011	90760	£2,999

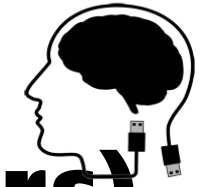
Query:

```
SELECT Product No FROM Cars WHERE Make = Toyota
```

Result:

0002

Clause	Purpose
SELECT	
FROM	
WHERE	



Topic 2.2.3 – SQL & Data Structures (Answers)

Activity 1: Introduction to databases

Difficulty level:

Identify the two components of a database and the meaning of each SQL clause.

Product No.	Registration	Make	Year	Mileage	Price
0001	AV60 HES	Peugeot	2010	33156	£5,500
0002	GF56 RTE	Toyota	2006	26875	£8,500
0003	FD02 YOU	Hyundai	2002	85300	£3,499
0004	AD62 HGF	Peugeot	2012	50887	£7,649
0005	AF63 THE	Peugeot	2013	45860	£6,780
0006	GF64 NGB	Renault	2014	38665	£6,199
0007	GR11 JUL	Renault	2011	90760	£2,999

Field

Record(s)

Query:

SELECT Product No FROM Cars WHERE Make = Toyota

Result:

0002

Clause	Purpose
SELECT	Fields that will be visible in the query.
FROM	Source of the data.
WHERE	Used to filter records.

Topic 2.2.3 – SQL & Data Structures

Activity 2: The use of SQL to search for data.

Difficulty level:

Identify the output for each of these SQL queries based on the database called cars.

Product No.	Registration	Make	Year	Mileage	Price
0001	AV60 HES	Peugeot	2010	33156	£5,500
0002	GF56 RTE	Toyota	2006	26875	£8,500
0003	FD02 YOU	Hyundai	2002	85300	£3,499
0004	AD62 HGF	Peugeot	2012	50887	£7,649
0005	AF63 THE	Peugeot	2013	45860	£6,780
0006	GF64 NGB	Renault	2014	38665	£6,199
0007	GR11 JUL	Renault	2011	90760	£2,999

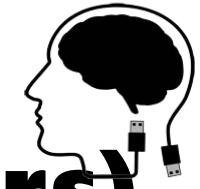
Query:

SELECT Product No, FROM Cars WHERE Mileage > 40000

Query:

SELECT Registration, Price, FROM Cars WHERE Make = Peugeot AND Price <7000

Which character is used to select all fields?



Topic 2.2.3 – SQL & Data Structures (Answers)

Activity 2: The use of SQL to search for data.

Difficulty level:

Identify the output for each of these SQL queries based on the database called cars.

Product No.	Registration	Make	Year	Mileage	Price
0001	AV60 HES	Peugeot	2010	33156	£5,500
0002	GF56 RTE	Toyota	2006	26875	£8,500
0003	FD02 YOU	Hyundai	2002	85300	£3,499
0004	AD62 HGF	Peugeot	2012	50887	£7,649
0005	AF63 THE	Peugeot	2013	45860	£6,780
0006	GF64 NGB	Renault	2014	38665	£6,199
0007	GR11 JUL	Renault	2011	90760	£2,999

Which character is used to select all fields?	*

Query:

SELECT Product No, FROM Cars WHERE Mileage > 40000

0003, 0004, 0005, 0007

Query:

SELECT Registration, Price, FROM Cars WHERE Make = Peugeot AND Price <7000

AV60 HES 5500
AF63 THE 6780

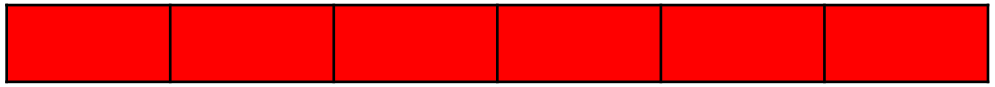
Topic 2.2.3 – SQL & Data Structures

Activity 3: The use of SQL to search for data.

Write a SQL statement that will output the following data from a database named products.

Product Code	ProductName	Price	QtyInStock	OnOrder
T4578	Baked beans	£0.50	288	N
T5632	Tomato soup	£0.63	170	N
D1144	Butter	£1.56	17	Y
B8443	Tea bags	£1.75	580	N
B7761	Instant Coffee	£2.56	22	N
D5229	Greek Yoghurt	£0.85	28	Y
D1258	Pizza	£1.25	57	N

Difficulty level:

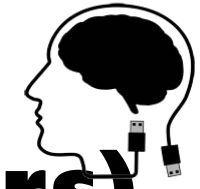


Query:

Display all fields, where quantity of stock is less than 30.

Query:

The product name and code of all products priced between 60p and £2.00.



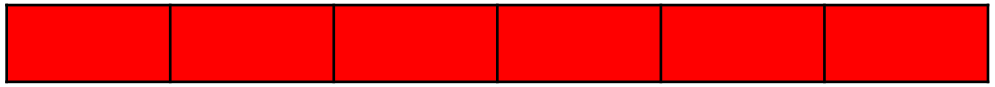
Topic 2.2.3 – SQL & Data Structures (Answers)

Activity 3: The use of SQL to search for data.

Write a SQL statement that will output the following data from a database named products.

Product Code	ProductName	Price	QtyInStock	OnOrder
T4578	Baked beans	£0.50	288	N
T5632	Tomato soup	£0.63	170	N
D1144	Butter	£1.56	17	Y
B8443	Tea bags	£1.75	580	N
B7761	Instant Coffee	£2.56	22	N
D5229	Greek Yoghurt	£0.85	28	Y
D1258	Pizza	£1.25	57	N

Difficulty level:



Query:

SELECT * FROM products WHERE QtyInStock < 30

Display all fields, where quantity of stock is less than 30.

Query:

SELECT ProductName, Product Code FROM products WHERE Price >0.59 and <2.01

The product name and code of all products priced between 60p and £2.00.

Topic 2.2.3 – Subprograms

Activity 1: Understanding subprograms

Difficulty level:

Correctly annotate the subprogram using the options provided and provide a description to each.

```
def add (Num1,Num2):
    a = 10
    return (Num1+Num2)/a

b = int(input("Input a number"))
c = int(input("Input a second number"))
print(add(b,c))
```

Keywords
Call
Parameters
Local variable
Define
Return
Global variable

Topic 2.2.3 – Subprograms (Answers)

Activity 1: Understanding subprograms

Difficulty level:

Correctly annotate the subprogram using the options provided and provide a description to each.

Define function
This will enables you to create a subprogram.

Parameters
Variables that stores values that can be passed through the subprogram.

```
def add (Num1,Num2):
    a = 10
    return (Num1+Num2)/a

b = int(input("Input a number"))
c = int(input("Input a second number"))
print(add(b,c))
```

Local variable
A variable that can only be used with the subprogram it's assigned to.

Global variable
A variable that can be accessed anywhere in the program.

Call
This allows you to call the subprogram so it can be used.

Return
This allows you return a value. Used in functions.

Keywords
Call
Parameters
Local variable
Define
Return
Global variable

Topic 2.2.3 – Subprograms

Activity 2: Functions v Procedures

Answer the following questions.

Difficulty level:

Function:

```
def add (num1, num2):
    return num1 + num2

    a = int(input("Enter first number: "))
    b = int(input("Enter second number: "))
    print(add(a,b))
```

Procedure:

```
def message():
    print("Hello world")
    print("Welcome")


message()
```

Describe the purpose of a subprogram. **(2 marks)**	
Guidance: • What is a subprogram • Why are they used?	

State the difference between a function and a procedure. **(1 mark)**	
Guidance: • How do they handle data differently?	

Identify **three** reasons why programmers use subprograms **(3 marks)**	
Guidance: • Testing • Efficiency • Error checking • Decomposition	

Topic 2.2.3 – Subprograms (Answers)

Activity 2: Functions v Procedures

Answer the following questions.

Difficulty level:

Function:

```
def add (num1, num2):
    return num1 + num2

    a = int(input("Enter first number: "))
    b = int(input("Enter second number: "))
    print(add(a,b))
```

Procedure:

```
def message():
    print("Hello world")
    print("Welcome")


message()
```

Describe the purpose of a subprogram. **(2 marks)**	
Guidance: • What is a subprogram • Why are they used?	An independent block of code that performs a specific task and can be re-used at any point during the program.

State the difference between a function and a procedure. **(1 mark)**	
Guidance: • How do they handle data differently?	Functions return a result whereas a procedure doesn't.

Identify **three** reasons why programmers use subprograms **(3 marks)**	
Guidance: • Testing • Efficiency • Error checking • Decomposition	• Easier to test as it's modular. • More efficient because less lines of code. • Easier to find errors if they're contained within subprograms.

Topic 2.2.3 – Subprograms

Activity 3: Writing algorithms that include subprograms.

Difficulty level:

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Kim is creating a game in which users have to throw two dice and add the results together to find out the overall score.

This has to happen many times in the game, therefore it's recommended that Kim uses a subprogram in her code.

Write an algorithm, using a subprogram that will add together the values of two dice together for Kim's game. **(4 marks)**

Guidance:

- Set up random integers (BP1)
- Defined the function (BP2)
- Set the correct parameters (BP3)
- Returned the correct value (BP4)

Topic 2.2.3 – Subprograms (Answers)

Activity 3: Writing algorithms that include subprograms.

Difficulty level:

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Kim is creating a game in which users have to throw two dice and add the results together to find out the overall score.

This has to happen many times in the game, therefore it's recommended that Kim uses a subprogram in her code.

Write an algorithm, using a subprogram that will add together the values of two dice together for Kim's game. **(4 marks)**

Guidance:

- Set up random integers (BP1)
- Defined the function (BP2)
- Set the correct parameters (BP3)
- Returned the correct value (BP4)

- Import random
- Roll1 = random.randint(1,6)
- Roll2 = random.randint(1,6)
- def dice(roll1, roll2):
  - Score = roll1 + roll2
  - return Score

Topic 2.2.3 – Arrays

Activity 1: Extracting data from arrays.

Difficulty level:

Complete the tables below to identify which item(s) will be printed from each array.

1D Array

0	1	2	3	4	5
Darren	Mike	Chloe	Katie	Mary	Steve

Code:

```
Names = [“Darren”, “Mike”, “Chloe”, “Katie”, “Mary”, “Steve”]
```

print(Names[0]	
print(Names[0:2])	

2D Array

	0	1	2	3	4	5
0	Darren	Mike	Chloe	Katie	Mary	Steve
1	M	M	F	F	F	M

Code:

```
Names = [[“Darren”,”M”],[ “Mike”,”M”], “[Chloe”,”F”,][
“Katie”,”F”,[ “Mary”,”F”,][“Steve”,”M”]]
```

print(Names[0]	
print(Names[1][1])	

Topic 2.2.3 – Arrays (Answers)

Activity 1: Extracting data from arrays.

Difficulty level:

Complete the tables below to identify which item(s) will be printed from each array.

1D Array

0	1	2	3	4	5
Darren	Mike	Chloe	Katie	Mary	Steve

Code:

```
Names = [“Darren”, “Mike”, “Chloe”, “Katie”, “Mary”, “Steve”]
```

print(Names[0]	**Darren**
print(Names[0:2])	**Darren, Mike**

2D Array

	0	1	2	3	4	5
0	Darren	Mike	Chloe	Katie	Mary	Steve
1	M	M	F	F	F	M

Code:

```
Names = [[“Darren”,”M”],[ “Mike”,”M”], “[Chloe”,”F”,][
“Katie”,”F”,[ “Mary”,”F”,][“Steve”,”M”]]
```

print(Names[0]	**Darren, M**
print(Names[1][1])	**M**

Topic 2.2.3 – Arrays

Activity 2: Completing algorithms

Difficulty level:

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Complete

The following algorithm will check through a list of numbers and output whether it's an even or odd number.

Fill in the gaps to complete the algorithm.

Guidance:

- Set up the array (BP1)
- Iterate through the array (BP2/BP3)
- Use selection to check for any remainders (BP4)
- Output for even and for odd (BP5)

```
Numbers = [3,6,8,11,14]
for x in ……..:
    if x[…….] % 2 == ……..
        print("Even")
    else:
        print("Odd")
```

Complete

The following algorithm will check through the list of names and their gender from the previous slide.

Fill in the gaps to complete the algorithm.

Guidance:

- Iterate through the array (BP1)
- Selection statement (BP2)
- Comparison of correct index position (BP3)
- Output message with correct index position. (BP4)

```
Names = [["Darren","M"],[
"Mike","M"], "[Chloe","F",][
"Katie","F",[
"Mary","F",]["Steve","M"]]
for x in ……..:
    if ……. == "M":
        print(….," is Male")
    else:
        print(……," is
        Female")
```

Topic 2.2.3 – Arrays (Answers)

Activity 2: Completing algorithms

Difficulty level:

Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

Complete

The following algorithm will check through a list of numbers and output whether it's an even or odd number.

Fill in the gaps to complete the algorithm.

Guidance:

- Set up the array (BP1)
- Iterate through the array (BP2/BP3)
- Use selection to check for any remainders (BP4)
- Output for even and for odd (BP5)

```
Numbers = [3,6,8,11,14]
for x in numbers:
    if x[0] % 2 == 0
        print("Even")
    else:
        print("Odd")
```

Complete

The following algorithm will check through the list of names and their gender from the previous slide.

Fill in the gaps to complete the algorithm.

Guidance:

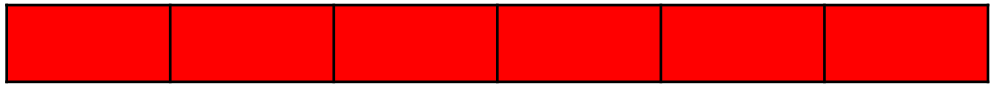
- Iterate through the array (BP1)
- Selection statement (BP2)
- Comparison of correct index position (BP3)
- Output message with correct index position. (BP4)

```
Names = [["Darren","M"],["Mike","M"], "[Chloe","F",]["Katie","F",["Mary","F",]["Steve","M"]]
for x in Names:
    if x[1] == "M":
        print(x[0]," is Male")
    else:
        print(x[0]," is Female")
```

Topic 2.2.3 – Arrays

Activity 3: Writing algorithms that include subprograms.

Difficulty level:



Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

A supermarket is placing a bulk order on soft drinks. Each soft drink comes in different sizes and their value (in pounds) are stored in a two-dimensional (2D array) with the identifier prices. 2D array shown on the right.

A function called checkprice() searches through the 2D array and applies the correct price. The quantity ordered and price are passed in as parameters and the appropriate result is returned.

Guidance:

- Define the function (BP1)
- Set parameters (BP2)
- Iterate through the list of prices (BP3)
- Selection statement that checks the correct index position (BP4/BP5)
- Call the function (BP6)

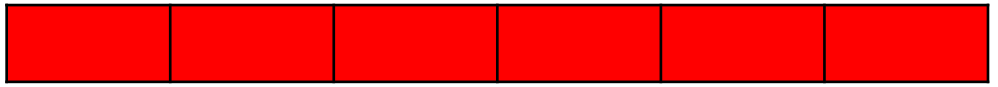
2D Array:

330	0.60
500	1.00
2000	1.50

Topic 2.2.3 – Arrays

Activity 3: Writing algorithms that include subprograms.

Difficulty level:



Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

A supermarket is placing a bulk order on soft drinks. Each soft drink comes in different sizes and their value (in pounds) are stored in a two-dimensional (2D array) with the identifier prices. 2D array shown on the right.

A function called checkprice() searches through the 2D array and applies the correct price. The quantity ordered and price are passed in as parameters and the appropriate result is returned.

2D Array:

330	0.60
500	1.00
2000	1.50

Guidance:

- Define the function (BP1)
- Set parameters (BP2)
- Iterate through the list of prices (BP3)
- Selection statement that checks the correct index position (BP4/BP5)
- Call the function (BP6)

```
prices = [["300",0.60],["500",1.00],["2000",1.50]]
def checkprice(quantity,size):
    for x in prices:
        if x[0] == size:
            return quantity * x[1]
        elif x[0] == size:
            return quantity * x[1]
        elif x[0] == size:
            return quantity * x[1]
        else:
            print("Not recognised")

quantity = int(input("Enter quantity"))
size = input(input("Enter size"))
print(checkprice(quantity,size))
```

Topic 2.3.1 – Defensive design

Activity 1: Authentication

Difficulty level:

Using the description cards below, identify what these different methods of authentication are.

A type of challenge–response test used in computing to determine whether the user is human. It provide challenges that are difficult for computers to perform but relatively easy for humans.	A method of authentication that adds extra layers of security. It strengthens access security by requiring two methods to verify your identity.
Answer:	Answer:
A backup measure used to authenticate the user of a website or an application in the event that they have forgotten their user name and/or password. This is normally in the form of a question that only the user should know.	This is a passcode that is valid for only one login session or transaction, on a computer system or other digital device. This could be a six-digit numerical code sent in real time as SMS while performing the transaction
Answer:	Answer:

Topic 2.3.1 – Defensive design (Answers)

Activity 1: Authentication

Difficulty level:

Using the description cards below, identify what these different methods of authentication are.

A type of challenge–response test used in computing to determine whether the user is human. It provide challenges that are difficult for computers to perform but relatively easy for humans.	A method of authentication that adds extra layers of security. It strengthens access security by requiring two methods to verify your identity.
Answer: **CAPTCHA**	Answer: **Two-factor authentication (2FA)**
A backup measure used to authenticate the user of a website or an application in the event that they have forgotten their user name and/or password. This is normally in the form of a question that only the user should know.	This is a passcode that is valid for only one login session or transaction, on a computer system or other digital device. This could be a six-digit numerical code sent in real time as SMS while performing the transaction
Answer: **Security question**	Answer: **Passcode/One-time Password (OTP)**

Topic 2.3.1 – Defensive design

Activity 2: Input validation

Using the snippets of code provided. Identify which validation method is being used from the list provided.

Difficulty level:

```
while True:
    password = input("Enter a new password: ")
    if len(password) < 10:
        print("Password must have more than ten characters")
    else:
        print("Password length correct")
```

Answer:

```
while True:
    option = int(input("Option: "))
    if option >= 1 and option <=4:
        print("You have chosen option",option)
        break
```

Answer:

Check digit
Format check
Lookup table
Range check
Presence check
Length check

```
while True:
    number = input("Enter an integer")
    if number.isdigit():
        number = int(number)
        break
    else:
        print("That's not an integer")
```

Answer:

```
Name = ""
print ("Please enter your Name")
Name = input()
while Name == "":
  print ("Sorry, your name must be entered, try again ")
  Name = input()
print ("Thank you " + Name + " Your comment has been posted")
```

Answer:

Topic 2.3.1 – Defensive design (Answers)

Activity 2: Input validation

Difficulty level:

Using the snippets of code provided. Identify which validation method is being used from the list provided.

```
while True:
    password = input("Enter a new password: ")
    if len(password) < 10:
        print("Password must have more than ten characters")
    else:
        print("Password length correct")
```

Answer: **Length check**

```
while True:
    option = int(input("Option: "))
    if option >= 1 and option <=4:
        print("You have chosen option",option)
        break
```

Answer: **Range check**

Check digit
Format check
Lookup table
Range check
Presence check
Length check

```
while True:
    number = input("Enter an integer")
    if number.isdigit():
        number = int(number)
        break
    else:
        print("That's not an integer")
```

Answer: **Format check**

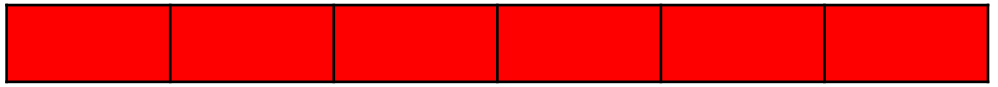
```
Name = ""
print ("Please enter your Name")
Name = input()
while Name == "":
  print ("Sorry, your name must be entered, try again ")
  Name = input()
print ("Thank you " + Name + " Your comment has been posted")
```

Answer: **Presence check**

Topic 2.3.1 – Defensive design

Activity 3: Maintainability

Difficulty level:



Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

```
a = int(input("Enter first number"))
b = int(input("Enter second number"))
if a > b
print(a,"is bigger than",b)
elif a < b
print(b,"is bigger than",a)
else
print("The values are the same")

a = int(input("Enter first number"))
b = int(input("Enter second number"))
if a > b
print(a,"is bigger than",b)
elif a < b
print(b,"is bigger than",a)
else
print("The values are the same")

a = int(input("Enter first number"))
b = int(input("Enter second number"))
if a > b
print(a,"is bigger than",b)
elif a < b
print(b,"is bigger than",a)
else
print("The values are the same")
```

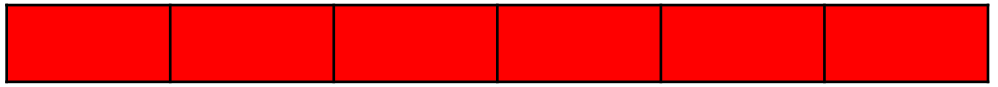
Suggest four ways this program could be maintained.

In the box below, re-write the improved version of this program.

Topic 2.3.1 – Defensive design (Answers)

Activity 3: Maintainability

Difficulty level:



Answer the question below using OCR Exam Reference Language or a high-level programming language that you have studied.

```
a = int(input("Enter first number"))
b = int(input("Enter second number"))
if a > b
print(a,"is bigger than",b)
elif a < b
print(b,"is bigger than",a)
else
print("The values are the same")

a = int(input("Enter first number"))
b = int(input("Enter second number"))
if a > b
print(a,"is bigger than",b)
elif a < b
print(b,"is bigger than",a)
else
print("The values are the same")

a = int(input("Enter first number"))
b = int(input("Enter second number"))
if a > b
print(a,"is bigger than",b)
elif a < b
print(b,"is bigger than",a)
else
print("The values are the same")
```

Suggest four ways this program could be maintained.

Appropriately named variables	Indentation	Comments	Subprograms

In the box below, re-write the improved version of this program.

```
def numbers (): #Procedure to store the program.
    a = int(input("Enter first number")) #User to enter two numbers
    b = int(input("Enter second number"))
    if a > b: #Selection statements to compare values
        print(a, "is bigger than", b)
    elif a < b:
        print(b,"is bigger than", a)
    else:
        print("The values are the same")

numbers() #Call the procedure
```

Topic 2.3.2 - Testing

Activity 1: The purpose of testing

Fill in the missing gaps to each description below.

Difficulty level:

The purpose of testing is to find _____ in the program so that they can be _____ before use and ensure the program is _____ as intended.

_____ testing is done _____ the development of the program.

_____ testing is done when the program is complete.

Selecting and using suitable test data can done using the following methods.

1. _____
2. _____
3. _____
4. _____

Topic 2.3.2 – Testing (Answers)

Activity 1: The purpose of testing

Fill in the missing gaps to each description below.

Difficulty level:

The purpose of testing is to find **errors/defects** in the program so that they can be **fixed** before use and ensure the program is **working** as intended.

Iterative testing is done **during** the development of the program.

Final testing is done when the program is complete.

Selecting and using suitable test data can done using the following methods.

1. **Valid**
2. **Invalid**
3. **Boundary**
4. **Erroneous**

Topic 2.3.2 - Testing

Activity 2: Test data

Match up type of test data to the correct description.

Difficulty level:

1. Valid	A. Data of the correct data type which should be rejected by a computer system.
2. Invalid	B. Data which should be accepted by a program without causing errors.
3. Boundary	C. Data of the correct type which is on the very edge of being valid.
4. Erroneous	D. Data of the incorrect data type which should be rejected by a computer system.

Topic 2.3.2 – Testing (Answers)

Activity 2: Test data

Match up type of test data to the correct description.

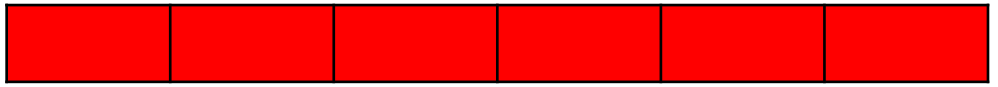
Difficulty level:

1. Valid	A. Data of the correct data type which should be rejected by a computer system.
2. Invalid	B. Data which should be accepted by a program without causing errors.
3. Boundary	C. Data of the correct type which is on the very edge of being valid.
4. Erroneous	D. Data of the incorrect data type which should be rejected by a computer system.

Topic 2.3.2 - Testing

Activity 3: Completing a test plan

Difficulty level:



Use the code above to identify different types of test data that can be used add them to the test plan.

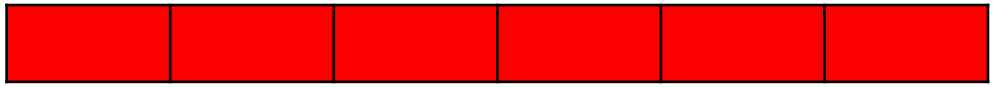
```
#Guess the number challenge
import random

Answer = random.randint(1,100)#Random number between 1 and a 100.

score = 1 # Record number of guesses
guess = int(input("Enter a number between 1 and a 100"))

while guess != Answer: #While user guesses incorrectly.
    if guess < Answer: #Indicates whether they are too high or low.
        print("Too low")
    elif guess > Answer:
        print("Too high")
    elif guess == Answer:
        print("Correct")
    else:
        print("Out of range")
    score = score + 1 #Adds to guesses
    guess = int(input("Please enter a number between 1 and 100"))

print("Well done, it took you",score,"guesses") #Prints out the total
```

No.	Test data	Type	Expected outcome	Actual outcome
1				
2				
3				
4				

Topic 2.3.2 – Testing (Answers)

Activity 3: Completing a test plan

Difficulty level:



Use the code above to identify different types of test data that can be used add them to the test plan.

```
#Guess the number challenge
import random

Answer = random.randint(1,100)#Random number between 1 and a 100.

score = 1 # Record number of guesses
guess = int(input("Enter a number between 1 and a 100"))

while guess != Answer: #While user guesses incorrectly.
    if guess < Answer: #Indicates whether they are too high or low.
        print("Too low")
    elif guess > Answer:
        print("Too high")
    elif guess == Answer:
        print("Correct")
    else:
        print("Out of range")
    score = score + 1 #Adds to guesses
    guess = int(input("Please enter a number between 1 and 100"))

print("Well done, it took you",score,"guesses") #Prints out the total
```

No.	Test data	Type	Expected outcome	Actual outcome
1	**54**	**Valid**	**Pass**	**Pass**
2	**132**	**Invalid**	**Fail**	**Fail**
3	**101**	**Boundary**	**Fail**	**Fail**
4	**“Apple”**	**Erroneous**	**Fail**	**Fail**

Topic 2.5.1 – Programming languages

Activity 1: High-level v Low-level languages

Answer the following questions.

Difficulty level:

In the table below tick one box per row to identify which language the syntax best applies to.

Syntax	High-level language	Low-level language
ADD 4		
print(“Hello world”)		
STA 6		
00010011		
Name = input(“Enter name”)		

Fill in the gaps

_____ languages are human-friendly. They are very easy to understand and learn by any programmer

_____ languages are machine-friendly. They are very difficult to understand and learn by any human.

Topic 2.5.1 – Programming languages (Answers)

Activity 1: High-level v Low-level languages

Answer the following questions.

Difficulty level:

In the table below tick one box per row to identify which language the syntax best applies to.

Syntax	High-level language	Low-level language
ADD 4		✓
print(“Hello world”)	✓	
STA 6		✓
00010011		✓
Name = input(“Enter name”)	✓	

Fill in the gaps

High-level languages are human-friendly. They are very easy to understand and learn by any programmer

Low-level languages are machine-friendly. They are very difficult to understand and learn by any human.

Topic 2.5.1 – Programming languages

Activity 2: Compiler v Interpreter

Difficulty level:

Tick one or more boxes per row to identify the statements that best match up with a compiler or an interpreter.

Statement	**Compiler**	**Interpreter**
Execute one line of code at a time.		
A high-level language translator.		
Generate a list of errors at the end of execution.		
Executes all the code in one go.		
Will stop at the first error.		
Must be translated each time it's run.		
Packaged into a machine code file.		
Will run faster.		

Topic 2.5.1 – Programming languages (Answers)

Activity 2: Compiler v Interpreter

Difficulty level:

Tick one or more boxes per row to identify the statements that best match up with a compiler or an interpreter.

Statement	**Compiler**	**Interpreter**
Execute one line of code at a time.		✓
A high-level language translator.	✓	✓
Generate a list of errors at the end of execution.	✓	
Executes all the code in one go.	✓	
Will stop at the first error.		✓
Must be translated each time it's run.		✓
Packaged into a machine code file.	✓	
Will run faster.	✓	

Topic 2.5.1 – Programming languages

Activity 3: IDE Features

Answer the following question.

Difficulty level:

Source code:

```
password = ""
while password != "secret":
    password = input("Please enter the password: ")
    if password == "secret":
        print("Thank you. You have entered the correct password")
    else:
        print("Sorry the value entered in incorrect - try again")
```

Translator:

```
Traceback (most recent call last):
  File "\\dshs-05\CorbettDC\Desktop\Python\password.py", line 2, in <module>
    while passwords != "secret":
NameError: name 'passwords' is not defined
```

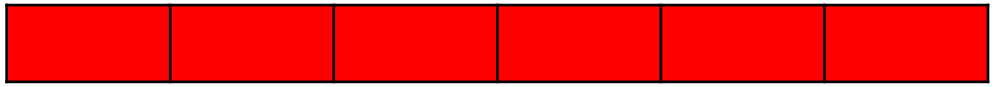
Translator
Syntax highlighter
Run-time environment
Bracket matching
Auto completion
Auto indentation
Debugging tool

Topic 2.5.1 – Programming languages (Answers)

Activity 3: IDE Features

Answer the following question.

Difficulty level:



Source code:

```
password = ""
while password != "secret":
    password = input("Please enter the password: ")
    if password == "secret":
        print("Thank you. You have entered the correct password")
    else:
        print("Sorry the value entered in incorrect - try again")
```

Syntax highlighter
Displays source code in different colours so certain commands in orange, functions in purple etc..

Bracket matching
It highlights matching sets to identify whether you've used the correct amount of brackets.

Auto indentation
Automatically indents the next line if required.

Translator:

```
Traceback (most recent call last):
  File "\\dshs-05\CorbettDC\Desktop\Python\password.py", line 2, in <module>
    while passwords != "secret":
NameError: name 'passwords' is not defined
```

Debugging tool
Allows code to be inspected for errors with suggestions on where the problem lies.

Translator
Syntax highlighter
Run-time environment
Bracket matching
Auto completion
Auto indentation
Debugging tool