Computing @ turton

YEAR 9

Theme 2

Algorithms & Programming

HOMEWORK BOOKLET

Name

Form

CS

Introduction

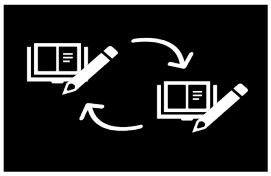
During theme 2, we will continue through the journey of Algorithms and Programming. We will begin by revisiting the basics of an effective algorithm and use this to develop our pseudo code, flowchart and programming skills, focussing on the three constructs of:

1. *Sequence*
2. *Selection*
3. *Iteration*

See the key below to find out what the icons below mean:

Self Assessment: You will mark your work at the start of next lesson.
ENSURE YOU COMPELTE HOMEWORK AS MARKS WILL BE COLLECTED IN!

Edmodo quiz at the start of next lesson based on your homework.
SO MAKE SURE YOU REVISE!



Peer Assessment: Homework marked by your class mate at the start of next lesson.
MAKE SURE YOU HAVE YOUR HOMEWORK DONE SO YOU CAN SWAP WITH ANOTHER PUPIL!

Stuck? Got a question? Email your teacher.

Mr Rifai *(Head of Computing)*	**rifaim@turton.uk.com**
Miss Davison	davisone@turton.uk.com
Miss Pascoe	pascoej@turton.uk.com

Help Tools for Theme 2

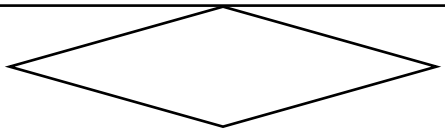
Pseudo Code Tools

Outputs	**print (“ “)**
Variables	**variablename = value**
Inputs	**variablename = input()**

Selection	**If variablename *operator* condition then**
	Else if ‘’ ‘’ ‘’ then
	Else
	Endif

Definite Iteration	**For variablename = range** **endfor**
Indefinite Iteration	**while variablename *operator* condition** **endwhile**

Flowchart Symbols

	**Terminator (Starts / Ends)**
	**Process**
	**Input / Output**
	**Decision (Selection / Iteration)**
	**Connector**

H/W 1: Algorithms

Due Date:

Using pseudo code, create an algorithm for logging onto Edmodo. You must include inputs/outputs and processes.
Challenge – Can you include iteration?

	WWW:	***EBI:***	***Total:*** ____ / 54

H/W 2 : *Flowcharts*

Due Date:

Using last week's homework, turn your pseudo code algorithm into a working flowchart.

Challenge – Can you include iteration?

WWW:	***EBI:***	***Total:*** ____ / 30

H/W 3 : Converting Algorithms

Due Date:

Convert the following python program into a flowchart.

```
print("What is your name?")
name = input()
if name == "Miss Davison":
    print("Teacher of Computer Science")
elif name = "Miss Pascoe"
    print("Teacher of Maths & Computer Science")
```

Challenge – Can you include a third selection statement for Mr Rifai?

	WWW:	***EBI:***	***Total:*** ____ / 20

H/W 4 : Problem Solving Algorithms

Due Date:

A maths teachers wants to create an algorithm which converts inches to centimetres (* 2.54).

Create an algorithm that:

1. **Asks the user to enter the number of inches they want to convert.**
2. **Converts from inches to centimetres.**
3. **Outputs the result.**

Challenge – Reverse the algorithm to convert centimetres to inches.

	WWW:	***EBI:***	***Total:*** ____ / 20

H/W 5 : Problem Solving Algorithms

Due Date:

Create a logging in algorithm (pseudo code or flowchart) which:

1. **Asks the user to input their password once.**
2. **Asks the user to input their password again.**
3. **While the passwords don't match, ask the user to re enter their details.**

Challenge – Can you repeat this for a username algorithm?

WWW:	***EBI:***	***Total:*** ____ / 20

Computing @ turton

H/W 6 : Problem Solving

Due Date:

A baker needs to calculate her ingredients depending on the product she makes. Create an algorithm (pseudo or code flowchart) that:

1. **Creates 3 variables: eggs, flour, sugar (all set to 0).**
2. **Asks the user to input the product they are baking.**
3. **If the product is "cake": flour = 175, sugar = 175, eggs = 4**
4. **Else if the product is "brownie": flour = 80, sugar = 275, eggs = 3**
5. **Outputs these ingredients.**

Challenge – Use iteration to *prevent the program from continuing if they enter anything other than brownies or cakes.*

WWW:	***EBI:***	***Total:*** ____ / 20

Theme 2

Keywords and Concepts

Computing@turton

Algorithm - An algorithm is a sequence of steps that can be followed to complete a task. *Be aware that a computer program is an implementation of an algorithm and that an algorithm is not a computer program.*

Decomposition - Decomposition means breaking a problem into a number of sub-problems, so that each sub- problem accomplishes an identifiable task, which might itself be further subdivided.

Abstraction - The process of removing unnecessary detail from a problem. E.g. The London tube map is a form of abstraction. The map tells you what line each station is on and which other lines are connected. Very useful for a person travelling. Not useful to an engineer who is planning where to dig tunnels for a new line.

3 Programming Constructs

1 Sequence
In a sequence structure, an action or event leads to the next in a predetermined order.

```
qty = input()
total = qty * price
print(total)
```

2 Selection
A question is asked, depending on the answer the program takes one, two or more courses of action.

```
x = input()
if x > 5 then
   print("too big")
else
   print("just right!")
endif
```

3 Iteration
A process wherein a set of instructions or structures are repeated in a sequence a set number of times or until a condition is met.

```
for count = 1 to 10
   print("ROVERS!")
next count
```

flowcharts

Computing@turton

Variables and constants

Variable – Sometimes we need computers to remember the information we give it. A variable can be thought of as a box (memory location) that the computer can use to store a value. The value held in the box may change or vary. A program can use as many variables as it needs.

A variable is made up of three parts:
- A name (identifier)
- A type **(data type – see below)**
- A value (what you are storing)

name = "Mr rifai"
*The variable is called **name**, its data type is a **string**, and its value is **Mr Rifai***

Assignment - In order to change the data value stored in a variable, you use an operation called assignment. Different values may be assigned to a variable at different times during the execution of a program.

```
x = 5    #here we are assigning 5 to the variable x
name = input()  #here whatever the user types
                in will be assigned to the
                variable name.
```

Scope – The scope of a variable can be **local** or **global**.
- ***local*** variables only work in the procedure or loop they are created in.
- ***global*** variables can be accessed from any point in a program.

Declaration – Declaring a name for a variable is saying what the data type will be and where it will be stored in memory.
E.g. Dim name as String

Data types	
String	Combination of characters that appear on the keyboard (alphanumeric)
Integer	A whole number
Real	A decimal/fractional number
Boolean	True/False or Yes/No
Character/Char	Used for single letters

Example:

String	Float or Real	Integer	Boolean
Title	**Rating**	**Times Viewed**	**Favourite**
Zombie Attack	9.5	83	True
True Love	8.0	5	True
Mission: Pluto	2.5	1	False

Knowledge Organiser

Selection continued...

For **selection** in programming you can use ***if ...else***

```
age = int(input("How old are you?"))
if age >= 70:
  print("You are aged to perfection!")
else:
  print("You are a spring chicken!")
End if
```

You can use ***else if*** to provide more choices.

```
age = int(input("How old are you?"))
if age >= 70 then:
  print("You are aged to perfection!")
elseif age == 50 then:
  print("Wow, you are half a century old!")
else:
  print("You are a spring chicken!")
```

Iteration

The third programming construct is ***iteration***. Means repetition, so ***iterative*** statements always involve performing a loop in the program to ***repeat*** a number of statements.

There are 2 types of iteration:

1. **Indefinite** – iteration continues until some specified condition is met.
   e.g. WHILE...END WHILE and REPEAT...UNTIL
2. **Definite** – Iteration is carried out a set number of times and is decided in advance.
   e.g. FOR....NEXT loops in programming.

Indefinite = Condition-controlled loop
Definite = Counter-controlled loop

WHILE ...END WHILE loop
The condition is tested **before each iteration.**
And the statements in the loop will be **executed** if the **condition** is **true.**
The statements in the loop may not be executed (if the condition is initially false)

```
num = input()
WHILE num > 0
   total = total + num
   num = input()
END WHILE
print total
```

WHILE Loops are used when the number of repetitions is **NOT** known in advance
WHILE Loops are known as **condition-controlled**, as the loop ends when a **condition** is met.

REPEAT ...UNTIL loop
Similar to the **WHILE** loop.
Difference being that the Boolean expression is tested at the **end** of the loop!
This means the loop is **always performed** at least **once**!

```
num = input()
REPEAT
  total = total + num
  num = input()
UNTIL num = 0
print total
```

In the above code, when num = 0, the loop will stop.
The condition is tested **AT THE END** of the loop – hence the instructions within the loop **GET EXECUTED AT LEAST ONCE**
Also **condition-controlled** and used when repetitions **NOT known** in advance.

FOR ...NEXT loop
Useful when you **know in advance** the number of iterations you wish to perform.
Uses a counter variable.

```
FOR i = 1 to 5
   print ("ROVERS")
NEXT
```

The above code will iterate 5 times and print *ROVERS* five times. The counter variable i starts at 1 and ends at 5 and **jumps** out of the loop.
Counter-controlled as the Counter variable is used to stop the Loop.
Used when the number of repetitions are **known in advance.**
(Finite number of Loops)

```
x = 1
WHILE x < 6
   print x
   x = x + 1
END WHILE
```

```
x = 1
REPEAT
   print x
   x = x + 1
UNTIL x > 5
```

```
FOR x = 1 TO 5
   print x
NEXT
```