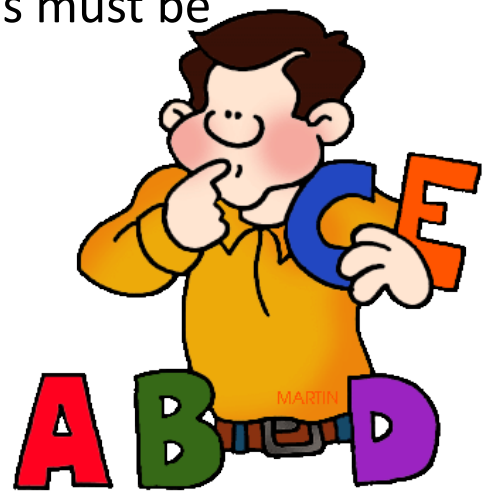
Sequence

To develop knowledge by identifying sequence, selection and iteration ***in algorithms***

Sequence:
an action, or event, leads to the next ordered action in a predetermined order.
- WJEC

*In other words….the order in which **instructions** occur and are processed*

- Algorithms consist of a series of instructions in a specific order.
- This is the order or **sequence** in which the instructions must be carried out for the algorithm to work.
- A computer can only follow instructions in the order they are given.
- If the sequence is not right the computer will still follow the order in which the instructions are given.



Selection

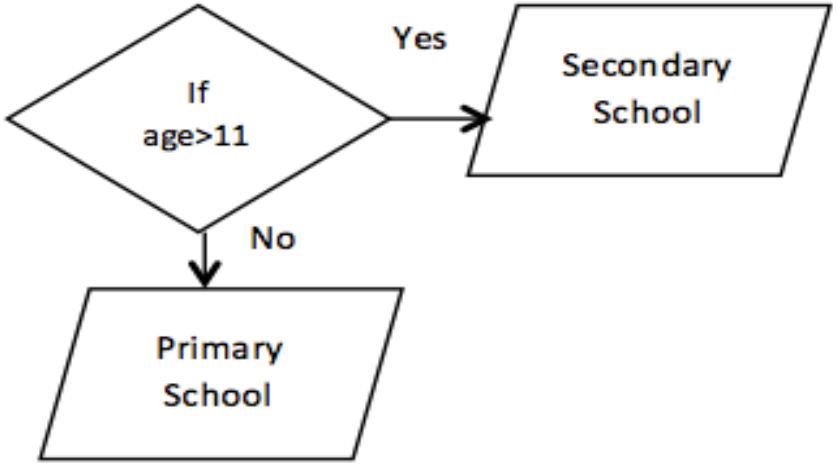
To develop knowledge by identifying sequence, selection and iteration ***in algorithms***

Selection:
In selection, a question is asked, and depending on the answer, the program takes one of two courses of action.

- A **selection** instruction is one where a decision must be made. There are times when an instruction in an algorithm may give different options

- In python you have used IF ELIF and ELSE

```
2 if age > 11 then
3     print "Secondary school
4 else
5     print "Primary School"
6     end if
7
```



Iteration

To develop knowledge by identifying sequence, selection and iteration ***in algorithms***

Iteration:
An iteration is a single pass through a set of instructions.
Most programs contain loops of instructions that are executed over and over again. The computer repeatedly executes the loop, iterating through the loop.

Sometimes an algorithm will require a set of steps to be carried out more than once or many times. This is called ***iteration*** *and often referred to as a loop in the program.*

Example of iteration in an algorithm:

```
for count = 1 to 10
        print count * 10
next count
```

Count controlled iteration

If we want a program to repeat a set of two instructions four times we can use a 'for…next' loop.

```
1
2 for i = 1 to 4
3     draw line
4     turn 90
5 next i
```

Learning Intention

To develop knowledge by identifying basic constructs in programs and algorithms

To secure understanding by Explaining use and giving examples of basic constructs in algorithms

Condition controlled iteration

condition-controlled iteration - repeatedly executes a section of code until a **condition** is met or no longer met (example: WHILE LOOP)

```
set count = 0
while count < 6
        print "Coding is cool"
        count = count + 1
repeat
```

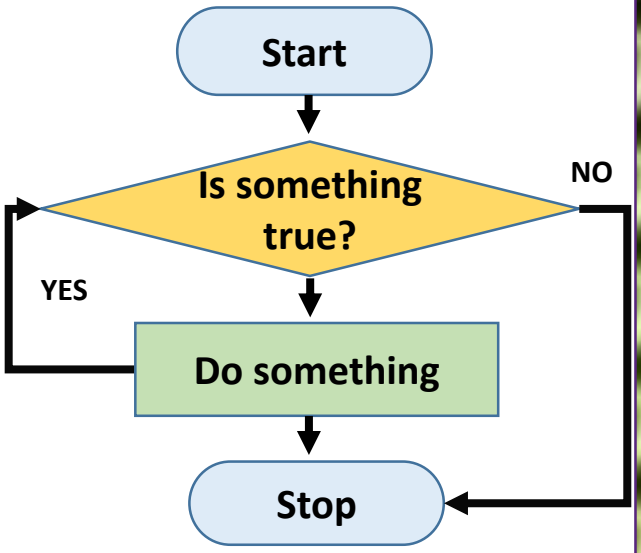
To secure understanding by Explaining use and giving examples of basic constructs in algorithms

A **variable**, (count in above example) , is used for the condition.

The while statement tests the condition,
(tests if count is less than 6 in example above)

If the result is TRUE, the code within the loop is executed

The program then loops back to the condition, which is tested again.



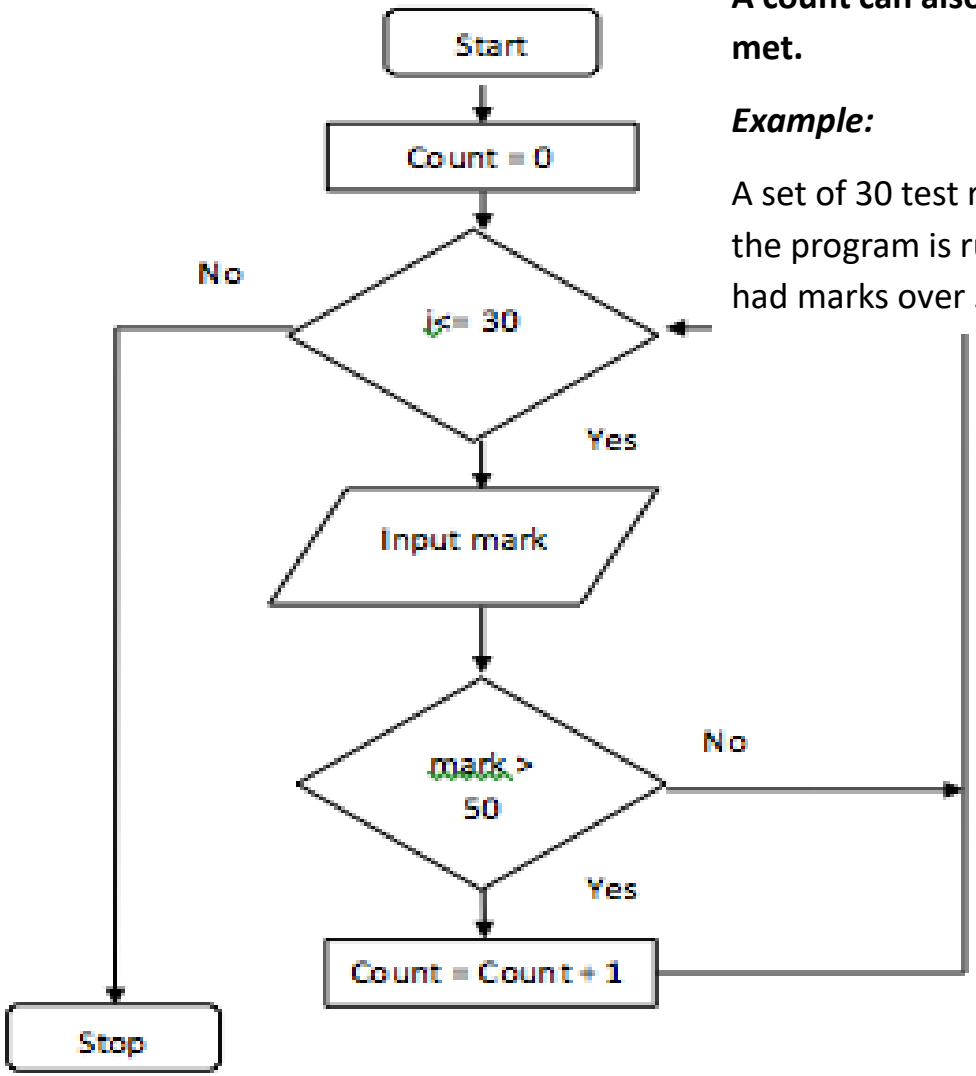
Using count and rogue values with loops

- Eventually **all loops must be terminated**.
- Sometimes we will not know how many times we will need the instructions in the loop to be used.
- If this is the case **we can control the program by using a count or a rouge value.**
- **Count will record the number of times a process is carried out**. *When the count reaches the required number the loop will terminate*

```
2  count = 0
3
4  repeat
5      input data
6      count = count + 1
7  until count = 10
8
9
```

To achieve excellence
Identify, explain and use counts and rogue values in algorithms and programs.

Using count to stop a loop



A count can also be used to check whether a condition has been met.

Example:

A set of 30 test results is to be entered into a program. Each time the program is run the teacher wants to know how many pupils had marks over 50.

```
1
2  count = 0
3
4  for i = 1 to 30
5      input mark
6      if mark > 50 then
7          count = count + 1
8      end if
9  next if
10
```

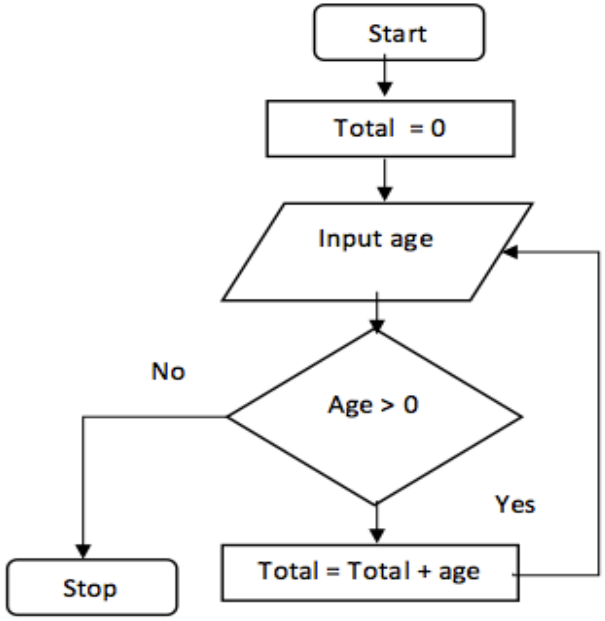
To achieve excellence

Explain more complex constructs and string handling.

Using count and rouge values with loops

A rogue value is a value that falls outside the range of possible values for the data that is being processed

If we were calculating the average age of a class of children, we could set a rogue value of -1 to stop the loop as no child can have a negative age.




```
1
2  total = 0
3
4  repeat
5      input age
6      if age > 0 then
7          total = total + age
8      end if
9  until age = -1
10
11
```

Once a **rogue value** outside of the condition has been found it will terminate the loop

Another example:
A program asks the user to enter a number between 1 and 10.
If they enter numbers outside this range, it is a rogue number

String Handling

Creating strings

To create a string we need to define a variable as a string and assign a value to the variable.

```
greeting is String
greeting = “Hello from me!”
print greeting
Hello from me!
```

Measuring the length of a string

To find out how many characters are in a string we used the **‘len’** command. Any keyboard stroke is counted as a character so spaces and special characters like an exclamation mark are counted as well as letters and numbers.

```
greeting is String
greeting = “Hello from me!”
length = len(greeting)
print length
13
```

To secure understanding by Explaining use and giving examples of basic constructs in algorithms

To achieve excellence Explain more complex constructs and string handling.

String Handling

Replace part of a string

To replace part of a string you need to use the 'replace' command.

```
txt is String
message is String
txt = "Have a happy birthday"
message = replace(txt, "happy", "fantastic")
print message
```

Have a fantastic birthday

To secure understanding by Explaining use and giving examples of basic constructs in algorithms

To achieve excellence Explain more complex constructs and string handling.

Task 2 : String handling

S.P.I.R.I.T
✓ Independence

Task

1. **Glue in string handling table**
2. **Explain and give example of the following:**

- **Creating a string**
- **Measuring a string**
- **Replacing a string**

Excellence:
Use WJEC INFO
Add

- **Joining strings**
- **Comparing strings**

Learning Intention

To develop knowledge by identifying basic constructs in programs and algorithms

To secure understanding by Explaining use and giving examples of basic constructs in algorithms

To achieve excellence
Explain more complex constructs and string handling.